FORMAL VERIFICATION OF PROBABILISTIC SYSTEMS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Luca de Alfaro
December 1997

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Zohar Manna
(Principal Adviser)

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

John C. Mitchell

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Rob J. van Glabbeek

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Ugo Montanari

Approved for the University Committee on Graduate Studies:

Abstract

Methods for the formal verification and specification of systems are a critical tool for the development of correct systems, and they have been applied to the design of hardware, software and control systems. While some system properties can be studied in a non-probabilistic setting, others, such as system performance and reliability, require a probabilistic characterization of the system. The formal analysis of these properties is especially necessary in the case of safety-critical systems, which must be designed to meet specific reliability and performance guarantees.

This dissertation focuses on methods for the formal modeling and specification of probabilistic systems, and on algorithms for the automated verification of these systems. Our system models describe the behavior of a system in terms of probability, nondeterminism, fairness and time.

The formal specification languages we consider are based on extensions of branching-time temporal logics, and enable the expression of *single-event* and *long-run average* system properties. This latter class of properties, not expressible with previous formal languages, includes most of the performance properties studied in the field of performance evaluation, such as system throughput and average response time. We also introduce a classification of probabilistic properties, which explains the relationship between the different classes of properties expressible in the proposed languages.

Our choice of system models and specification languages has been guided by the goal of providing efficient verification algorithms. The algorithms rely on the theory of *Markov decision processes,* and exploit a connection between the graph-theoretical and probabilistic properties of these processes. This connection also leads to new results about classical problems, such as an extension to the solvable cases of the *stochastic shortest path* problem, an improved algorithm for the computation of reachability probabilities, and new results on the *average reward* problem for semi-Markov decision processes.

Acknowledgements

First, I would like to thank Zohar Manna, my advisor during the six years I spent at Stanford. Under his direction, I learnt everything I know about formal methods and theoretical computer science. Without his generosity, unconditional support and encouragement, this dissertation would not have been possible. I would also like to thank him for creating such a friendly and stimulating atmosphere in the research group.

I would then like to thank the members of my readings and orals committee, Rob van Glabbeek, Tom Henzinger, Grigori Mints, John Mitchell, Ugo Montanari and Rajeev Motwani, for their time and for their valuable comments.

Next, I would like to express my gratitude to Andrea Bianco, who brought to my attention the topic of formal verification of probabilistic systems, and with whom I wrote my first paper on the subject. Working with him was an extremely enjoyable experience.

I would also like to thank Professor Arthur Veinott, who gave me very valuable advice in my study of the theory of Markov decision processes, and who never tired of answering my questions about references in the literature.

Many thanks also go to the other members of my research group: Nikolaj Bjørner, Anca Browne, Michael Colón, Arjun Kapur, Henny Sipma and Tomás Uribe. Phyllis Winkler, the secretary of the group, pulled me out of several inextricable bureaucratic problems with her efficiency and zeal.

My gratitude also goes to Angelo Raffaele Meo, my undergraduate advisor, who first nurtured my research activities, and who greatly helped me to come to Stanford.

I am also very grateful to the school teachers that had the most influence on me; first among them Adriano Attanasio, who turned me from scoundrel to attentive pupil; Giuseppe Gallione, who taught me the beauty of elementary mathematics and axiomatic geometry, Elena Micheletti, who taught me calculus and physics with passion and joy, and Carla Origlia, who taught me the love for natural sciences. I am also very indebted to Gillian Sortino, from whom I learnt the language in which I am now writing.

I would also like to thank my family for their love and encouragement, for having taught me to love mathematics and physics, and for teaching me how to think (mostly) rationally. Last, but not least, I would like to thank my wife Françoise Beaufays for her incredible patience, support and understanding, as well as for other reasons that are beyond the scope of this dissertation.

Luca de Alfaro
Stanford, California
December 1997

Contents

Chapter 1

Introduction

Some systems can be designed in such a way that they are guaranteed to behave correctly under all circumstances, short of a total breakdown of the system or of its physical environment. For example, it is possible to design integrated circuits or write some computer programs in such a way that they will perform exactly as intended, if the power supply is present and the computer is not physically damaged.

On the other hand, for many types of systems this guarantee of correctness cannot be achieved, either because it would be impractically expensive, or because the system includes intrinsically unreliable components. Examples of these systems include telecommunication systems and computer networks, distributed systems built over these networks, and complex software-controlled physical systems such as industrial plants, electrical power stations and transportation systems.

When a system cannot be guaranteed to exhibit the desired behavior under all circumstances, it becomes important to characterize the likelihood of undesirable behaviors. For these systems, the concept of unconditional correctness is substituted by bounds on the probability that certain behaviors occur.

Even for systems whose correctness can be unconditionally guaranteed, the study of system performance relies on a probabilistic model of the system and its load. In fact, the term *system performance* commonly refers to the long-run average time required by a system to perform given tasks, such as message delivery or response to a query. These long-run averages can be computed on the basis of the probabilistic models.

This dissertation presents languages for the formal specification of performance and reliability properties of probabilistic systems, and algorithms for the automated verification of these properties on finite-state systems. What distinguishes this dissertation from previous work in the area is the broader range of systems and properties covered. We consider timed systems whose behavior is characterized by probability, nondeterminism, fairness and time; among them are systems whose execution model extends that of generalized stochastic Petri nets with the introduction of nondeterminism and transitions with unspecified delay distributions.

The languages for formal specification that we present extend the expressive capabilities of existing languages by allowing the expression of *long-run average* properties of systems. This class of properties includes many basic performance measures considered in the field of performance modeling and evaluation, such as system throughput and average response time. Along with this extension, we present a classification of probabilistic properties of systems, which explains the relationship between the different types of properties discussed in the dissertation.

The algorithms for the automated verification of finite-state systems rest on the theory of dynamic programming and optimal control, as well as on results from automata theory. In particular, the theory of Markov decision processes played a central role in the development of the algorithms. Several algorithms depend on a new approach to the study of Markov decision processes, based on a joint analysis that combines graph-theoretical and probabilistic methods. Aside from the development of the algorithms, this approach leads to the solution of problems on Markov decision processes that had resisted previous analysis, such as the non-negative and the non-positive cases of the *stochastic shortest path* problem, and the characterization of the relation between two optimization criteria for semi-Markov decision processes.

1.1 The Benefits of Formal Methods

The study of system performance and reliability has so far been carried out mostly without the use of formal methods. The motivations for the application of formal methods to these problems can be summarized as follows.

Precise specification languages. A formal approach offers a language in which one can express with precision complex system specifications. Most system properties relating to reliability and performance have so far been expressed using a mixture of natural language and numbers. While this is adequate for simple properties, the precise specification of different scenarios of malfunctioning can become rather complex, and the precision of a formal language is thus beneficial. Similarly, while an informal approach is adequate for the specification of simple performance indices, a formal language makes it possible to define in a precise way complex performance criteria and measures.

Complete coverage. While simulation methods can produce accurate estimates of reliability and performance indices for many instances of purely probabilistic systems, they fare less well for systems in which nondeterminism is present. In fact, for purely probabilistic systems it is possible to use the probabilistic structure of the system to estimate the accuracy of performance indices computed by repeated simulation. In presence of nondeterminism, however, these accuracy estimates are substituted by the weaker notion of coverage, similarly to the case of hardware and software testing. Without using formal methods, complete coverage is hard or impossible to achieve for systems in which probability and nondeterminism coexist, and incomplete coverage does not provide a reliable measure of confidence towards the satisfaction of design requirements.

Common framework for the study of correctness, reliability and performance properties. The study of reliability and performance properties often relies on a prior analysis of the correctness of the system under consideration. The use of formal methods offers a common framework in which one can study system correctness, performance and reliability, facilitating the interchange of information between the different types of properties.

1.2 System Models

In this dissertation, we model the behavior of systems in terms of probability, nondeterminism, fairness and time. Probability enables the modeling of system components with unreliable or probabilistic behavior, and it is an essential element in the study of reliability and performance properties. Nondeterminism enables the modeling of unknown scheduling mechanisms, and it can be used as a replacement for more accurate probabilistic models in the construction of system approximations. Fairness, jointly with nondeterminism, enables the modeling of unknown (but non-zero) probabilities, and it is a widely used ingredient in the modeling of reactive systems. Time is used to specify waiting-time distributions for the system transitions, and it is an essential element for the evaluation of system performance.

We present two different models for probabilistic systems: a high-level model that enables a concise and readable description of systems, and a low-level model to which the verification algorithms can be applied.

The high-level model is that of *stochastic transition systems* (STSs). Stochastic transition systems are a probabilistic extension of first-order transition systems. The transitions can either be immediate, in which case they are taken as soon as they become possible, or they can have an associated probability distribution of waiting times. This probability distribution can either be exponential or unspecified. The choice of transition is also subject to fairness requirements. Once a transition is taken, its effects are described in terms of both probability and nondeterminism. The execution model of stochastic transition systems thus extends that of generalized stochastic Petri nets with the introduction of nondeterminism and unspecified delay distributions.

Our low-level computational model is that of *timed probabilistic systems* (TPSs), and is based on *Markov decision processes.* Markov decision processes generalize Markov chains by associating to each state a set of possible actions. Each state-action pair is associated with a probability distribution for the successor state. The behavior of a Markov decision process consists in an infinite alternation of states and actions: at each state, an action is selected nondeterministically; the corresponding probability distribution is then used to choose the following state.

A TPS is a Markov decision process in which to each state-action pair is associated the expected time elapsed at the state when the action is chosen. Again, it is possible to specify fairness requirements for the choice of the actions.

The relation between STSs and TPSs is similar to the relation between a first-order transition system and its underlying state-transition graph: the first provide a concise and compositional language for system modeling, while the second provides the structure to which the model-checking algorithms can be applied. TPSs are a more general computational model than STSs, and in fact it is possible to automatically translate an STS into its corresponding TPS. This translation is part of our verification methodology, since the verification algorithms we present can be applied only to TPSs.

1.3 Specification Languages

The languages for the formal specification of probabilistic systems that we present in this dissertation are extensions of the branching-time temporal logics CTL and CTL*. The languages extend the expressive power of CTL and CTL* by enabling the expression of two types of probabilistic properties: *single-event properties* and *long-run average properties.*

1.3.1 Single-Event Properties

Single-event properties refer to the probability with which a system behavior satisfies a temporal logic formula, or to the expected time with which it reaches a specified set of states. These properties owe their name to the fact that they involve the occurrence of a single event (satisfying a formula, or reaching a set of states). In a Markov chain, the verification of these properties involves the computation of the reachability probability or the expected first-passage time for particular subsets of states of the chain.

The specification of single-event properties relies on two logical operators: an operator P that enables to express bounds on the probability of satisfying temporal formulas, and an operator D that enables to express bounds on the expected time to reach specified sets of states. The class of properties expressible with these two operators includes several reliability properties, and some performance ones.

1.3.2 Long-Run Average Properties

Long-run average properties refer to the long-run average outcome or duration of specific patterns that are repeated infinitely often along a behavior. In a Markov chain, the verification of these properties involves the computation of the steady-state distribution of the closed recurrent classes of the chain.

The specification of long-run average properties relies on *experiments,* which are related to the tests of process algebra. Experiments are small labeled graphs that can be used to specify behavior patterns of interest, such as the request for a resource followed by either a grant or a refusal. Unlike tests, experiments are meant to be repeated infinitely often along a behavior of the system. Two logical operators, $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$, enable the specification of the long-run average outcome and duration of experiments, respectively.

Long-run average properties include most classical performance measures of a system, such as its average response time and throughput, as well as several reliability measures. Unlike single-event properties, these properties could not be specified with previous formal languages, and their consideration is one of the novelty claims of this dissertation.

1.3.3 Classification of Probabilistic Properties

The study of single-event and long-run average properties leads to a classification of probabilistic properties of systems. Properties are classified along the two axes of probability vs. expected time, and single-event vs. long-run average. This classification is shown in Table 1.1, along with the logical operators used to specify each class of properties, and examples of properties belonging to each class.

The specification style we adopt is based on the specification of bounds for these performance and reliability measures: for example, we can write a formal specification requiring that the average system response time is below a specified upper bound.

1.4 Verification

Our choice of system models and specification languages has been guided by the goal of providing efficient algorithms for system verification. By efficient algorithms, we mean algorithms having polynomial time-complexity in the size of the encoding of the transition graph corresponding to the system. The algorithms provide either a positive answer to the verification problem, or information that characterizes the system behaviors that violate the specification. The algorithms can be applied to finite-state probabilistic systems: while infinite-state systems can also be analyzed through the use of abstractions and approximations, their study is beyond the scope of this dissertation.

The algorithms are based on the theory of Markov decision processes, and on results from the theory of automata on infinite words. In particular, the algorithms rest upon a connection, developed in this dissertation, between the probabilistic analysis of Markov decision processes and their graph-theoretical structure. This connection is useful not only for the design of the verification algorithms, but also in providing new results on classical problems in the theory of Markov decision processes.

Among these results are improved algorithms for the computation of reachability probabilities, as well as extensions to the solvable cases of the stochastic shortest path problem. The extension consists in algorithms for the solution of non-negative and non-positive instances of the stochastic

	PROBABILITY	**TIME**
SINGLE-EVENT	Probability of events $\mathbf{P}$ *What is the probability of reaching deadlock from an initial state?*	Expected reachability time $\mathbf{D}$ *From a "good" state, what is the expected time to failure?*
LONG-RUN AVERAGE	Long-run average outcome $\bar{\mathbf{P}}$ *How often are messages followed by acknowledge?*	Long-run average duration $\bar{\mathbf{D}}$ *What is the system's average response time?*

Table 1.1: Classification of probabilistic properties discussed in this dissertation. For each of the four classes of properties, we list also the logical operator used to specify them, as well as an informal example of a typical property belonging to the class.

shortest path problem, and has in turn led to improved algorithms for the non-negative case of the minimum expected total cost problem. Another result obtained in this dissertation is the proof of the equivalence of two alternative optimization criteria for semi-Markov decision processes, closing a problem that was open for more than two decades.

1.5 Outline of the Dissertation

The presentation of our methodology for the study of probabilistic systems begins in Chapter 2 with the introduction of *stochastic transition systems* (STSs), our high-level model for probabilistic systems. In the same chapter, we also overview the probabilistic temporal logics introduced later in the dissertation, and we present some examples of system modeling and specification.

Chapter 3 introduces our low-level system models, *timed probabilistic systems* (TPSs), closely related to Markov decision processes. The chapter develops the connection between the graph-theoretical and probabilistic properties of these models, and it describes two optimization problems: the *stochastic shortest path* problem and the *maximum reachability probability problem.*

Our simplest probabilistic temporal logics are introduced in Chapter 4, along with their model-checking algorithms. These logics are extensions of the branching-time logics CTL and CTL*, and can express single-event probabilistic properties. The specification of long-run average properties is the subject of Chapter 5. The chapter introduces *experiments,* and it presents probabilistic logics that can express bounds on the long-run average outcome or duration of experiments. The model-checking algorithms for these logics are presented in Chapter 6, along with their correctness proof. Chapter 6 also discusses optimization problems for semi-Markov decision processes.

Chapter 7 presents new results on the stochastic shortest path problem. The results are needed for subsequent model-checking algorithms, and are also of independent interest.

Chapter 8 extends our specification and verification methods to probabilistic systems with fairness. The chapter introduces our notion of fairness, called *probabilistic fairness,* and compares it with previous notions of fairness for probabilistic systems. The chapter then presents probabilistic logics and model-checking algorithms for systems with fairness.

Chapter 9 completes the presentation of our formal verification methodology by providing a translation from STSs to TPSs. The translation enables the use of the logics and model-checking algorithms of Chapters 3–8 for the specification and verification of the system models presented in Chapter 2.

1.6 How to Read This Dissertation

There are three ways of reading this dissertation, depending on whether the reader is interested primarily in formal verification of probabilistic systems, in dynamic programming and optimal control, or in both. Those who are interested in both topics can read this dissertation in linear order.

Those whose main interest is the formal verification of probabilistic systems can start with Chapters 2, 3, 4 and 5. They can then read the first section of Chapter 6, and from there continue with Chapters 8 and 9.

Those whose main interest is the theory of Markov decision processes can start from Chapter 3. From there, there are two alternatives. If they are interested in semi-Markov decision processes, they have no alternative but to proceed through Chapters 5 and 6. Otherwise, they can skip these two chapters and go directly to Chapter 7, which deals with extensions to the stochastic shortest path and expected total cost problems.

Throughout the dissertation we have denoted with (‡) the sections that contain detailed proofs and arguments that are only used in few occasions; these sections can be skipped at a first reading, if so desired.

1.7 Chronology

The results of Chapter 4 and, to some extent, Chapter 3, have been presented in Bianco and de Alfaro [BdA95] and de Alfaro [dA97]. The remainder of this dissertation consists of previously unpublished results.

1.8 Related Work

In this section we present an overview previous work in formal modeling and verification of probabilistic systems, mentioning also areas of other disciplines whose results have been used in this dissertation.

1.8.1 Markov Decision Processes and Probabilistic Automata

Our low-level computational model for probabilistic systems is provided by *Markov decision processes,* enriched with additional structure. Markov decision processes were introduced by Bellman [Bel57] and Howard [How60], and have been the subject of much research in Operations Research and Economics. Among the many books that have since been written on them, we referred most often to the ones by Derman [Der70], Puterman [Put94] and Bertsekas [Ber87, Ber95]. Other books on this topic are by Ross [Ros70a], Howard [How71], Heyman [HS82], Heyman and Sobel [HS84b], and Tijms [Tij86]. Puterman [Put94] gives an extensive bibliography on Markov decision processes. Bertsekas and Tsitsiklis [BT91] present several results on the stochastic shortest path problem. These results were mentioned in Chapter 3, and extended in Chapter 2.

The complexity of solving optimization problems on Markov decision processes has been considered in Papadimitriou and Tsitsiklis [PT87], which compare the complexities under nondeterminism alone and under nondeterminism with probability, and by Littman et al. [LDPK95]. The complexity of the policy improvement methods under various assumptions are discussed by Melekopoglou and Condon [MC94].

Markov decision processes are closely related to *probabilistic automata,* introduced by Rabin [Rab63]. By identifying the controls of Markov decision processes with the inputs of probabilistic automata, the two computational models are almost identical. They differ, however, in the way they are used: [Rab63] studies which sequences of inputs are accepted by probabilistic automata under various acceptance conditions; the work on Markov decision processes is aimed instead at identifying control policies that maximize or minimize given functionals. In this dissertation, the controls (or inputs) are used to model nondeterminism, and we study the effect of nondeterminism on quantities related to system performance and reliability: our point of view is thus closer to that of Markov decision processes.

Several related models that include both probability and nondeterminism have been presented in the computer science literature. Some of these models are the *concurrent Markov chains* of Vardi [Var85], the *probabilistic finite-state programs* of Pnueli and Zuck [PZ86], the *NP systems* of Wang and Larsen [WL92] and the *probabilistic automata* of Segala and Lynch [SL94] and Segala [Seg95a, Seg95b].

1.8.2 Logics of Programs and Probabilistic Dynamic Logics

Some of the early work in the study of probabilistic models of computation was done in the framework of logics of programs and of dynamic logics.

Kozen [Koz79] presents two alternative semantics for probabilistic programs. One of the semantics represents programs as functions on measurable spaces; the other represents programs as operators on linear Banach spaces of measures. The relationship between these semantics and the ordered domains of Scott and Strachey [SS71] is also discussed.

Ramshaw [Ram80] proposes Hoare-style rules for probabilistic programs, and applies them to the study of some algorithms.

The study of probabilistic dynamic logic began with Reif [Rei80], in which the program operators include probabilistic choice, and the tests can specify lower bounds on the probability of success. Axiomatizations for the resulting logics are discussed. Later, Feldman and Harel [FH82] present an extended probabilistic dynamic logic, and improve on the results of [Rei80] by proving decidability relative to the theory of second-order arithmetic.

Kozen [Koz83] presents a dynamic logic on probabilistic models. The logic is based on an arithmetic semantics, rather than a truth-functional one: propositions generalize measurable functions, programs correspond to real-valued functions, and the modal constructors are function transformers; the satisfiability relation returns a satisfaction probability. [Koz83] presents a small-model property theorem and a polynomial-space decision procedure for the logic. Feldman [Fel83] presents a propositional and decidable version of the logic of [FH82]; the decidability result is proved by reduction to first-order real-number theory.

Huth and Kwiatkowska [HK97] propose a probabilistic mu-calculus. The structure on which the mu-calculus is evaluated includes nondeterminism, and the logical connectives are given a fuzzy-logic semantics. For example, denoting with $\widetilde{P}(A)$ the fuzzy "probability" of event A, the rule for conjunction is $\widetilde{P}(A \wedge B) = \max\{0, \widetilde{P}(A) + \widetilde{P}(B) - 1\}$. As in [Koz83], the truth-functional semantics of formulas is replaced by an arithmetic semantics in terms of probability intervals; the choice of the fuzzy semantics leads to a simplification of the computational procedure to evaluate the intervals on a given structure.

A related line of research consists in considering programs as probabilistic predicate transformers. Hart, Sharir and Pnueli [HSP84] model programs as Markov chains, and propose a verification style for the input-output relations of programs based on the use of probabilistic intermediate assertions. Morgan, McIver and Seidel [CM96] extend the predicate-transformer approach to systems including nondeterminism, in part based on a relational model of probabilistic computation proposed by He et al. [HMS96]. The proposed specification languages have a Hoare-like syntax, extended to probability and nondeterminism.

1.8.3 Temporal Logics: Qualitative Verification

Given a probabilistic system, the term *qualitative verification* refers to the problem of determining whether formulas of a given specification language hold with probability 1 over the system. A considerable amount of work has been done on this subject.

Lehman and Shelah [LS82] present three temporal logics evaluated on Markov chains. The logics differ for the class of Markov chains on which they are evaluated: different classes of chains lead to different axiomatizations. The three classes of chains are finite chains, *bounded* chains (in which all non-zero transition probabilities are bounded from below by a non-zero constant), and infinite chains. They discuss linear-time temporal logics whose formulas hold iff they are satisfied with probability 1 by a behavior of the chain, and they provide axiomatizations for these logics.

Hart, Sharir and Pnueli [HSP82, HSP83] study systems consisting of a set of Markov chains that execute concurrently. The scheduling mechanism is assumed to be fair but is otherwise arbitrary. [HSP82] present a set of conditions that characterize when the system converges with probability 1 to a given set of states. The conditions are independent of the exact values of the transition probabilities. The relation between probabilistic and fair choice in the study of qualitative system properties is discussed for the first time. [HSP82, HSP83] also introduce *K-ergodic sets,* which are in some way similar to our end components. The conditions that ensure convergence are also related to some algorithms discussed in this dissertation.

Pnueli [Pnu83] presents a proof system that enables to verify whether a linear-time temporal logic formula holds with probability 1 over a system. Again, the system is modeled as the parallel composition of Markov chains. The concept of *extreme fairness* is introduced, as an approximation of the fairness properties resulting from choices in which each outcome has non-zero probability. A simpler concept of fairness, *state fairness,* is also described. This notion is the predecessor of the notions of fairness described in Vardi [Var85] and later in Kwiatkowska and Baier [KB96].

Hart and Sharir [HS84a] continue the line of research of [LS82] by introducing branching-time temporal logics to be interpreted on structures that are Markov chains. They distinguish two logics, depending on whether the chains are finite or bounded, and they provide decision procedures and axiom systems for the resulting logics. This work has then been extended by Hart and Sharir [HS86].

Vardi [Var85] considers system modeled as Markov chains and *concurrent Markov chains;* the latter are closely related to Markov decision processes, with the addition of *state fairness.* The paper presents algorithms for verifying whether these systems satisfy linear-time temporal logic specifications with probability 1. The proposed algorithm relies on automata-theoretic constructions, and it is perhaps the first algorithm in a line of research that led to the algorithm for the model checking of operator P described in Chapter 4.

Pnueli and Zuck [PZ86], and later [PZ93], consider the problem of verifying whether a probabilistic system satisfies with probability 1 a specification. The specification is written in the temporal logic RTL, which contains all past temporal operators, and the future temporal operators $\Box$ (*always*) and $\Diamond$ (*eventually*): the future operators $\mathcal{U}$ (*until*) is not present in RTL. The probabilistic system is described by a set of states and transitions; each transition can be taken in one of several *modes,* to which are associated mode probabilities. The system models are thus related to those of [Var85]. The algorithms presented by [PZ86, PZ93] are of single exponential time-complexity in the size of the specification, unlike those of [CY88]: the lower complexity is due to the use of RTL as specification language, instead of full temporal logic. Pnueli and Zuck [PZ93] also introduce α-fairness, a notion of fairness that accounts precisely for the properties of a probabilistic finite-state system, once the specific values of the transition probabilities are abstracted away. Thus, α-fairness is an improvement on the concept of extreme fairness of [Pnu83]. The system models proposed in [PZ86, PZ93] were influential in the development of our *stochastic transition systems.*

Courcoubetis and Yannakakis [CY88] consider systems modeled either as Markov chains or as concurrent Markov chains, and present algorithms to check whether a linear-time temporal formula holds with probability 1 over such models. The algorithm they propose for Markov chains improves on previous algorithms, and has time-complexity polynomial in the size of the system and exponential in the size of the formula. On concurrent Markov chains, the proposed algorithms have time-complexity polynomial in the size of the description of the system, and doubly exponential in the size of the temporal formula. They prove that these complexities are lower bounds, thus showing the optimality of their algorithms. The lower bound results proved in this paper are the source of all the lower bounds cited in this dissertation. An extended version of this work has been presented in [CY95].

Courcoubetis and Yannakakis [CY90] study the problem of maximizing the reward of a Markov decision process, given a reward structure specified by a set of omega-automata. This problem is related to the problem of computing the maximum and minimum probabilities with which a linear-time temporal formula holds over a system behavior. The proposed algorithm, related to those of [CY88], can therefore be used to solve the model-checking problem for operator P in our logics, in absence of fairness. In this dissertation we presented an alternative algorithm, based on the complete rather than partial determinization of an omega-automaton. This complete determinization can be performed with the algorithm described in Safra [Saf88, Saf92].

Alur, Courcoubetis and Dill [ACD91] introduce *real-time probabilistic systems,* which consist of a finite set of states and a finite set of transitions. To each transition is associated a delay distribution, which can either be geometrical, or have finite-support; the model of time is *discrete,* i.e. the domain of time is taken to be the set of non-negative integers. The paper presents algorithms for verifying that real-time probabilistic systems satisfy with probability 1 specifications written in the real-time temporal logic TCTL [ACD90]. These results have been later extended by [ACD92] to continuous-time systems and specifications encoded by timed automata [AD90]. The system models of [ACD91, ACD92] have been influential in the development of our *stochastic transition systems.*

1.8.4 Temporal Logics: Quantitative Verification

Given a probabilistic system, the term *quantitative verification* refers to the problem of determining the probability with which a system satisfies a specification, or more generally to the problem of verifying specifications that involve the numerical values of probabilities and expectations. Using temporal logics for the quantitative specification and verification of probabilistic systems is the approach taken in this dissertation. The previous work in this area has been described in Section 4.1.

1.8.5 Verification of Randomized Systems

Lynch, Saias and Segala [LSS94], as well as Pogosyants and Segala [PS95], propose methods for proving time-bound properties of randomized distributed algorithms. The verification of these properties is carried out in a natural proof style, but in fact the methods of [PS95] lend themselves to a certain degree of automation. The completeness of these methods, which are proposed for infinite-state systems, is not discussed.

Segala [Seg95b] presents the computational model of *probabilistic automata,* which are related to Markov decision processes, albeit the notation is fairly different. Real-time system are modeled by *probabilistic timed automata,* in which an additional set of actions represents the passing of time. For these systems, [Seg95b] proposes a verification approach that can be used to verify properties related to the probability of behaviors, the expected time to events, or the probability of meeting deadlines.

In the proposed approach, the proof that a system satisfies a specification is constructed in a natural proof style, in which lemmas and theorems are proved at will about the system. The methodology proposed is semi-formal, and it does not rely on a fixed set of inference rules. Facts about simulation relations, *coin lemmas* and other constructs are freely used in the course of the proof. This approach is to be contrasted with others that rely on a fixed set of inference rules and proof methods, such as the one developed by Manna and Pnueli [MP95] for fair transition systems. The lack of fixed inference rules makes for a fairly general verification framework in which complex systems can be studied. On the other hand, this freedom results in a lack of guidance in the process of proof construction, and no completeness results have been presented for this informal approach. The proposal of [Seg95b] seems thus to be better suited to the study of randomized algorithms than to the verification of reliability and performance properties of systems, not least since long-run average properties are not dealt with.

1.8.6 Process Algebras for Performance Modeling

Process algebras have been proposed as a compositional and high-level modeling language for the description of Markovian performance models.

Götz, Herzog and Rettelbach [GHR93] propose the process algebra TIPP, and discuss its use for performance modeling. Subsets of TIPP can be translated into continuous-time Markov chains, that can then be studied with the usual methods of performance evaluation.

Gilmore and Hillston [GH94] and Hillston [Hil95, Hil96] introduce the process algebra PEPA, which provides a concise and compositional modeling language for timed probabilistic systems. Systems described in PEPA can be automatically translated in continuous-time Markov chains, to which the usual performance evaluation algorithms can be applied. Donatelli, Ribaudo and Hillston [DRH95] compare this approach with that based on stochastic Petri nets, and Ribaudo [Rib95] presents a translation from PEPA into Petri Nets. The proceedings [Pro] contain further contributions to the use of process algebras as modeling languages for performance evaluation.

Bernardo, Donatiello and Gorrieri [BDG94a] introduce *Markovian Process Algebra* (MPA), a stochastic process algebra in which actions can be either immediate, or have an exponential delay distribution. The choice between immediate actions is mediated by priorities. Actions are further divided into *active* and *passive,* depending on whether they determine the timing of synchronous actions when the composition of systems is computed. Two semantics are defined for this algebra: one is an interleaving operational semantics, the other is a Markovian semantics defined by translation into continuous-time Markov chains. A third semantics, defined by translation into *Generalized Stochastic Petri Nets,* is presented in [BDG94b, BDG95]. Bernardo, Busi and Gorrieri [BBG95] introduce *stochastic contextual nets,* which extend GSPN with passive transitions and contextual arcs, and define a semantics for EMPA based on a translation into stochastic contextual nets. The advantage of this semantics, compared to the one defined by translation into GSPN, is that the resulting nets can be considerably smaller. Once the MPA description of a system is translated into a Markov chain, the performance of the system can be studied with the usual methods of performance analysis. Bernardo and Gorrieri [BG96] propose *extended Markovian process algebra* EMPA, closely related to MPA.

1.8.7 Stochastic Petri Nets

Much of the work in the performance evaluation of timed systems relies on the formalism of *stochastic Petri nets,* introduced by Natkin [Nat80], Molloy [Mol81, Mol82] and Symons [Sym80]. Balbo [Bal95] provides an overview of the origins and of the developments of this field. The proceedings of [Pet] are devoted to the use of Petri nets for the modeling and evaluation of system performance. Our *stochastic transition systems* are in part inspired by *generalized stochastic Petri nets,* introduced by Ajmone Marsan, Balbo and Conte [ABC84], and widely used in the performance evaluation of distributed computing systems. The recent book by Ajmone Marsan et al. [ABC$^+$94] is devoted to this approach to performance modeling.

Ciardo, Muppala and Trivedi [CMT91] introduce *GSPN reward models,* which are GSPN with additional labels that specify a reward structure. Specifically, a reward is associated to each marking, and an instantaneous reward is associated to each transition. A GSPN reward model can be translated into a continuous time Markov chain with an associated reward structure. Performance parameters such as the long-run average reward can be computed on this structure using the standard methods of Markov-chain analysis.

The modeling and expressive capabilities of GSPN reward models are similar to the capabilities of the methods presented in this dissertation, except that the system models we consider also include nondeterminism, fairness, and transitions with unspecified delays. Due to these extensions, our systems require radically different analysis methods: our methods trade the simplicity of Markov-chain analysis with the modeling power of nondeterminism, fairness, and unspecified delays.

1.8.8 Testing, Simulation and Process Algebras

Several notions of testing preorders, simulation and bisimulation have been introduced for probabilistic processes, inspired by the corresponding notions for non-probabilistic processes. Larsen and Skou [LS89, LS91] propose languages to encode tests for probabilistic processes, and introduce a notion of bisimulation that respects testing equivalence. They also present a probabilistic modal logic, showing that two processes are bisimilar exactly when they satisfy the same formulas of the logic, provided the transition probabilities are bounded away from zero. Van Glabbeek, Smolka, Steffen and Tofts [vGSST90, vGSST95] study the semantics of probabilistic processes, distinguishing between *reactive, stratified* and *generative* models of processes. For each of these, they present extensions of the process algebra SCCS (Milner [Mil83]) and bisimulation relations.

Jou and Smolka [JS90] present the probabilistic process algebra PCCS, based on SCCS [Mil83], and they discuss several notions of equivalence between PCCS processes, including the bisimulation defined in [LS89]. Christoff and Christoff [CC91] present algorithms for checking testing equivalences of purely probabilistic systems. The algorithms have polynomial time-complexity in the size of the processes being compared.

Jonsson and Larsen [JL91] present a novel framework, in which specifications are represented by systems whose transitions are labeled by intervals of probability. They define a new type of simulation relation between systems and specifications, which checks that the transition probabilities of the system belong to the intervals of the specification. They also present methods for checking the existence of simulation relations between system and specification, and between specifications. The complexity of these methods is not discussed.

Baeten, Bergstra and Smolka [BBS92] present a complete axiomatization for a probabilistic process algebra based on ACP [BK84]. Larsen and Skou [LS92] present compositional verification methods for purely probabilistic processes. The methods are based on process algebra for the modeling of probabilistic systems, and on temporal logic for their specification.

Cleaveland, Smolka and Zwarico [CSZ92] present testing preorders for probabilistic processes based on the probability with which a process passes a test. The preorders are extended also to substochastic processes by attributing the "missing probability" to undefined behavior (rather than to termination, as in classical substochastic process theory).

Wang and Larsen [WL92] present an extension of CCS (see De Nicola and Hennessy [DNH84]) which introduces a choice operator parameterized by a probability. The processes thus contain a combination of probability and nondeterminism. They present testing preorders for processes described in this calculus. Jonsson, Ho-Stuart and Wang [JHW94] present a denotational characterization of the preorders of [WL92], and introduce the notion of *reward testing.* This work has been extended by Jonsson and Yi [JW95], which consider both must- and may-testing, and obtain compositional preorders. The *experiments* introduced in this dissertation are somewhat reminiscent of the tests of [WL92, JHW94], but are meant to be repeated an infinite number of times, and it is their long-run average outcome that is evaluated.

Wu, Smolka and Stark [WSS94] introduce probabilistic I/O automata, derived from the I/O automata of Lynch and Tuttle [LT87]. They presents fully abstract and compositional characterizations of these automata with respect to a simple class of tests, which are encoded as probabilistic I/O automata with one distinguished output action.

Yuen, Cleaveland, Dayar and Smolka [YCDS94] present alternative characterizations of the testing preorders of [CSZ92], and they show that the characterization is fully abstract with respect to the testing preorder. Other testing preorders have been studied by Segala [Seg96], which uses tests that rely on multiple success actions. The testing preorders are characterized in terms of the probability distribution over failures.

Segala and Lynch [SL94, SL95] and Segala [Seg95b] discuss simulation and bisimulation relation for systems that include both probability and nondeterminism; [SL94] studies the preservation of properties expressed in probabilistic temporal logics under these relations. Segala [Seg95a] discusses the compositional semantics of *probabilistic automata,* and introduces preorders based on *trace distribution inclusions.* A trace distribution is related to the stochastic processes obtained by fixing a policy in a Markov decision process.

Algorithms for Simulations and Bisimulations

Even though many notions of probabilistic simulation, bisimulation and testing equivalence have been proposed, the number of algorithms for the checking of these relations is comparatively limited; moreover, the known algorithms do not cover the full spectrum of system types and relations. Algorithms for checking testing equivalences, simulations and bisimulations have been proposed by various authors. Christoff and Christoff [CC91] present algorithms for checking testing equivalences between purely probabilistic systems with silent actions. Baier [Bai96] gives algorithms for checking strong bisimulations between probabilistic automata, and Baier and Hermanns [BH97] present algorithms for checking the weak simulation and bisimulation of purely probabilistic processes.

Chapter 2

Stochastic Transition Systems

Many results presented in this dissertation depend on detailed definitions and lengthy arguments: reading them requires motivation. To provide this motivation, we begin this dissertation from the end, so to say, and we describe at once the formal verification framework that will be possible once all the results of the dissertation will be available.

We start this overview with the definition of *stochastic transition systems,* our high-level model for probabilistic systems, followed by examples of formal specification of probabilistic systems using extensions of temporal logic. While we are unable at this point to define the precise semantics of these models and logics, this chapter will provide insight into the types of systems and properties that we intend to study, and it will offer a preview of many concepts that will be introduced in the course of the dissertation. The rest of this dissertation is essentially devoted to building the set of definitions and techniques needed to turn these ideas into practice.

2.1 Stochastic Transition Systems

Stochastic transition systems are our high-level model for probabilistic real-time systems. They are derived from the *fair transition systems* of Manna and Pnueli [MP91, MP95], augmented with information about the probability and expected waiting times of transitions. They are also related to the *probabilistic finite-state programs* of Pnueli and Zuck [PZ86] and to *real-time probabilistic processes* of Alur, Courcoubetis and Dill [ACD91, ACD92], with the addition of nondeterminism, fairness and unspecified delay distributions.

Stochastic transition systems provide a concise and readable way to describe real-time probabilistic systems in terms of probability, waiting-time distributions, nondeterminism and fairness. The definition of stochastic transition systems represents a compromise between expressive power and complexity of the verification algorithms. While they cannot model all the subtleties of real physical systems, they can model a large class of systems, which includes (and extends) the class that can be modeled by generalized stochastic Petri nets, and they are well suited for the automated verification of probabilistic specifications.

Stochastic transition systems are not our basic computational model for probabilistic systems. The basic model, used to develop the specification languages and the verification algorithms, is that of *timed probabilistic systems* (TPSs), and it will be introduced in the next chapter. Essentially, there is the same relation between a stochastic transition system and its corresponding TPS as between a fair transition system and its representation in terms of a state transition graph. Stochastic transition systems, like fair transition systems, are first-order structures providing a high-level description of a system; TPSs provide the underlying computational model to which model-checking algorithms can be applied.

A stochastic transition system is assigned a semantics by translating it into its corresponding TPS. The translation will be presented in Chapter 9: here, we will rely on intuition, aided by examples, to understand their semantics.

2.1.1 Definition of Stochastic Transition Systems

Before we introduce the definition of stochastic transition systems, we need some notation regarding variables and formulas. Throughout the dissertation, we assume that we have an underlying set of interpreted function and predicate symbols which can be used to form logic formulas. These functions and predicates might include, for example, those of standard integer arithmetic. Constants are simply considered as 0-argument functions.

Given a set $\mathcal{V}$ of typed variables, we denote by $\mathit{Form}(\mathcal{V})$ the set of well-formed first-order logic formulas built from $\mathcal{V}$ using the available function and predicate symbols. Moreover, we let $\mathcal{V}' = \{x' \mid x \in \mathcal{V}\}$ be the set of variables obtained by priming each variable of $\mathcal{V}$.

With this notation, we can define stochastic transition systems as follows.

Definition 2.1 (stochastic transition system) A *stochastic transition system* (STS) is a triple $\mathcal{S} = (\mathcal{V}, \Theta, \mathcal{T})$, where:

- $\mathcal{V}$ is a finite set of typed variables, called the *state variables.* The state space S consists of all type-consistent interpretations of the variables in $\mathcal{V}$; we denote by $s[\![x]\!]$ the value at state $s \in S$ of variable $x \in \mathcal{V}$.
- $\Theta \in \mathit{Form}(\mathcal{V})$ is the *initial condition*, which specifies the set of initial states of the STS.
- $\mathcal{T}$ is the *set of transitions.*

With each transition $\tau \in \mathcal{T}$ are associated the following quantities:

- An *enabling condition* $\mathcal{E}_\tau \in \mathit{Form}(\mathcal{V})$, which specifies the set of states from which the transition can be taken.

- A number m_τ of *transition modes.* Each $1 \le i \le m_\tau$ represents a *transition mode,* and corresponds to a different possible outcome of the transition. For each $1 \le i \le m_\tau$ is specified:
  - A *mode transition formula* $\rho_i^\tau \in \mathit{Form}(\mathcal{V}, \mathcal{V}')$, which denotes the transition relation $\{(s, s') \mid (s, s') \models \rho_i^\tau\}$ corresponding to the mode, where (s, s') interprets $x \in \mathcal{V}$ as $s[\![x]\!]$ and $x' \in \mathcal{V}'$ as $s'[\![x]\!]$. Throughout the dissertation, we use the convention of omitting from the transition formula the variables that are not modified by the transition. If $s \models \mathcal{E}_\tau$, we assume that for all $1 \le i \le m_\tau$ it is $\{t \in S \mid (s, t) \models \rho_i^\tau\} \neq \emptyset$: if the transition is enabled, every transition mode can lead to at least one successor state.
  - A *mode probability* $p_i^\tau \in [0, 1]$, indicating the probability with which the mode is chosen. These probabilities are such that $\sum_{i=1}^{m_\tau} p_i^\tau = 1$.

The set $\mathcal{T}$ of transitions is partitioned in two subsets $\mathcal{T}_i$ and $\mathcal{T}_d$ of *immediate* and *delayed* transitions. Immediate transitions must be taken as soon as they are enabled. A subset $\mathcal{T}_f \subseteq \mathcal{T}_i$ indicates the set of *fair* transitions . The set $\mathcal{T}_d$ of delayed transitions is partitioned in the sets $\mathcal{T}_e$ and $\mathcal{T}_u$, where:

- $\mathcal{T}_e$ is the set of *exponential-delay transitions*. To each $\tau \in \mathcal{T}_e$ is associated a *transition rate* $\gamma_\tau > 0$, which indicates the instantaneous probability with which τ is taken. Precisely, if $F_\tau(t) = \Pr(\tau$ *not taken before delay* $t)$ is the cumulative delay distribution of τ, it is $\mathrm{d}F_\tau(t)/\mathrm{d}t = -\gamma_\tau F_\tau(t)$, or $F_\tau(t) = \exp(-\gamma_\tau t)$. Thus, if τ is the only enabled transition, γ_τ^{-1} is the expected time for which τ is enabled before being taken.
- $\mathcal{T}_u$ is the set of transitions with *unspecified delay distributions.* These transitions are taken with some delay, but the probability distribution of this delay, and the possible dependencies between this distribution and the past system history or present system state are not known. ∎

Given a state $s \in S$, we indicate by $\mathcal{T}(s) = \{\tau \in \mathcal{T} \mid s \models \mathcal{E}_\tau\}$ the set of enabled transitions at s. To define the semantics of an STS, it is convenient to assume that $\mathcal{T}(s) \neq \emptyset$ for all states $s \in S$. For this reason, we implicitly add to every STS an *idle transition* τ_{idle} defined as follows.

Definition 2.2 (idle transition) The *idle transition* is a transition $\tau_{idle} \in \mathcal{T}_e$ with the following parameters:

$$\mathcal{E}_{\tau_{idle}} = \mathit{true} \qquad m_{\tau_{idle}} = 1 \qquad \rho_1^{\tau_{idle}} = \bigwedge_{x \in \mathcal{V}} x' = x \qquad p_1^{\tau_{idle}} = 1 \qquad \gamma_{\tau_{idle}} = 1\,. \qquad ∎$$

Thus, the idle transition is always enabled, and when taken it does not change the value of the state variables. The choice of an unitary transition rate is arbitrary: any positive transition rate would do.

2.1.2 Informal Semantics of Stochastic Transition Systems

As mentioned earlier, the semantics of an STS is defined by translation into *timed probabilistic systems,* and it will be presented in Chapter 9. Here, we provide an informal semantics, which will suffice to understand the examples that will be presented. This informal semantics will also be used in Chapter 9 to justify the translation process. In this informal semantics, the temporal evolution of the system state is represented by a *timed trace.*

Definition 2.3 (timed trace) A *timed trace* is an infinite sequence $(s_0, I_0), (s_1, I_1), \ldots$ of pairs, where $I_k \subseteq \mathbb{R}^+$ is a closed interval, and s_k is a system state, for $k \geq 0$. The intervals must be contiguous, i.e. $\max I_k = \min I_{k+1}$ for all $k \geq 0$, and the first interval must begin at 0, i.e. $\min I_0 = 0$. ∎

A pair (s_k, I_k) in a timed trace indicates that during the interval of time I_k the system is in state s_k. The choice of considering only closed intervals is arbitrary: the specification languages we consider will not be able to distinguish between open and closed intervals. Note, however, that interval endpoints overlap, so that a timed trace is not equivalent to a function $\mathbb{R}^+ \mapsto S$ mapping each instant of time to a system state. In particular, zero-duration intervals are permitted, and they represent transitory states in which an immediate transition is taken before time advances. In this respect, the execution model is very similar to that of *generalized stochastic Petri nets* (GSPN) introduced by Ajmone Marsan, Balbo and Conte [ABC84]: the transitory states of a timed trace correspond to the *vanishing markings* of a GSPN. The notion of timed trace is also closely related to the *super-dense semantics* for hybrid systems described in Maler, Manna and Pnueli [MMP92].

The initial state of a timed trace must satisfy the initial condition: $s_0 \models \Theta$. For $k \geq 0$, state s_k determines the expected duration of interval I_k and the next state s_{k+1} as follows.

Expected Duration of I_k

The set of transitions enabled at s_k is given by $\mathcal{T}(s_k) \subseteq \mathcal{T}$. Depending on the type of transitions in $\mathcal{T}(s_k)$, there are two cases:

- $\mathcal{T}(s_k) \cap \mathcal{T}_i \neq \emptyset$. If there is an immediate transition enabled at s_k, the expected duration of I_k is 0. Thus, immediate transitions take precedence over delayed ones.

- $\mathcal{T}(s_k) \subseteq \mathcal{T}_d$. Let $\mathcal{T}_e(s_k) = \mathcal{T}(s_k) \cap \mathcal{T}_e$ be the enabled transitions with exponential distribution of waiting times, and $\mathcal{T}_u(s_k) = \mathcal{T}(s_k) \cap \mathcal{T}_u$ be the enabled transitions with unspecified distribution of waiting times. Note that $\mathcal{T}_e(s_k) \neq \emptyset$, due to the presence of the idling transition. The expected duration of I_k is given by

$$\Big(\sum_{\tau \in \mathcal{T}(s_k)} \gamma_\tau \Big)^{-1}, \tag{2.1}$$

where the rates $\gamma_\tau > 0$ for $\tau \in \mathcal{T}_u(s_k)$ are selected nondeterministically. This expected duration is derived from the hypothesis that the waiting times of each $\tau \in \mathcal{T}(s_k)$ are distributed exponentially, with rate γ_τ. Note that if $\mathcal{T}_u(s_k) = \emptyset$ the definition is exactly as in a GSPN.

Successor State s_{k+1}

The successor state is chosen in three steps: first, an enabled transition is chosen; second, a transition mode is chosen; and third, the successor state is chosen among the states that satisfy the mode's transition formula. These three choices are made as follows.

Choice of transition. Similarly to the case of expected duration, there are two cases, depending on the type of transitions enabled at s_k.

- $\mathcal{T}(s_k) \cap \mathcal{T}_i \neq \emptyset$. One of the transitions in $\mathcal{T}(s_k) \cap \mathcal{T}_i$ is chosen nondeterministically. This nondeterministic choice is subject to fairness requirements: if $\tau \in \mathcal{T}(s_k) \cap \mathcal{T}_f$, the criterion that governs the nondeterministic choice must select τ with non-zero probability. The subject of fairness is discussed in Chapter 8.

- $\mathcal{T}(s_k) \subseteq \mathcal{T}_d$. Let $\mathcal{T}_e(s_k) = \mathcal{T}(s_k) \cap \mathcal{T}_e$ and $\mathcal{T}_u(s_k) = \mathcal{T}(s_k) \cap \mathcal{T}_u$ as before. Transition $\tau \in \mathcal{T}(s)$ is chosen with probability
$$\gamma_\tau \Big/ \Big(\sum_{\tau' \in \mathcal{T}(s_k)} \gamma_{\tau'} \Big) , \tag{2.2}$$
where for $\tau \in \mathcal{T}_u(s_k)$, the rate γ_τ is the *same* that was nondeterministically selected for (2.1).

Choice of transition mode. Once transition τ has been chosen as described above, transition mode j is chosen with probability p_j^τ, for all $1 \leq j \leq m_\tau$.

Choice of successor state. Once both transition τ and mode $j : 1 \leq j \leq m_\tau$ have been chosen, the successor state s_{k+1} of s_k is chosen nondeterministically from the set $\{s' \in S \mid (s_k, s') \models \rho_j^\tau\}$.

Time Divergence

In our definition of timed trace, we have not excluded the traces in which time does not diverge. These traces can arise, since the time intervals in the trace can be point intervals, or can be arbitrarily small. In this chapter, we will ignore the issue of time divergence, and we will present systems in which the probability of non-time-divergent traces is 0. In the following chapters, we provide two solutions to the problem of time divergence.

In Chapter 3, we call *non-Zeno* the systems in which time-divergent traces occur always with probability 1. We provide algorithms to check whether systems are non-Zeno, and we limit ourselves to the study of non-Zeno systems.

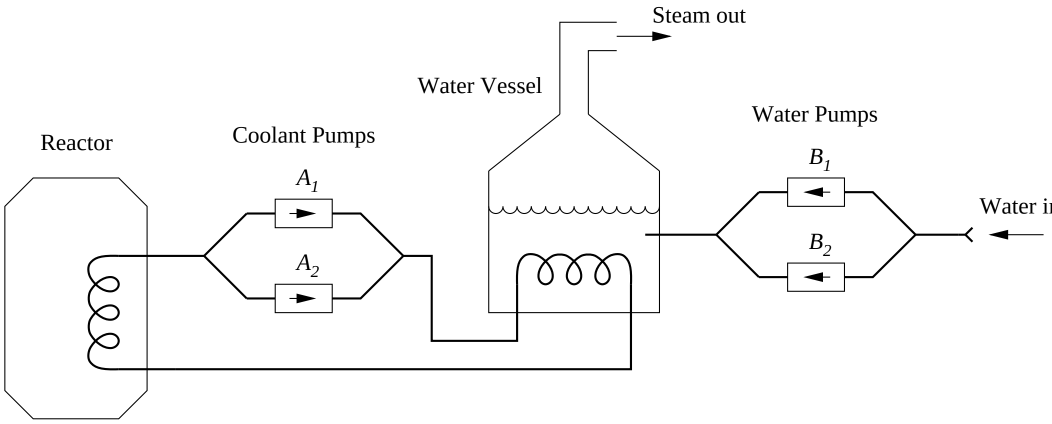

Figure 2.1: Reactor cooling system. Pipes are marked in boldface.

In Chapter 8, instead of requiring that all nondeterministic behaviors lead with probability 1 to time-divergent traces, we adopt the approach of Segala [Seg95b], and we consider only nondeterministic behaviors under which time-divergent traces have probability 1.

2.2 Application 1: Cooling a Reactor

To illustrate the use of STS, we apply them to the modeling and verification of the cooling system for a reactor. Our model will be rather simplistic, with the intent of presenting an introductory example: the model will certainly not be an accurate representation of a real cooling system.

The system, depicted in Figure 2.1, consists of a reactor R, cooled by a primary coolant circulating in pipe 1. In turn, the primary coolant is used to heat a water vessel, producing steam. Pumps A_1 and A_2 are used to circulate the primary coolant, and pumps B_1 and B_2 are used to replenish the water vessel. In this example, we will analyze the system reliability as a function of the failures and repair rates of these pumps.

To keep track of the state of the pumps, we use four variables a_1, a_2, b_1, b_2: each variable assumes value 1 if the pump is working, and 0 if it is broken. One additional variable c keeps track of the state of the system as a whole: it assumes value 1 normally, and 0 if the reactor has overheated severely, causing a shutdown. Thus, the set of variables of the STS is $\mathcal{V} = \{a_1, a_2, b_1, b_2, c\}$. Initially, the system is in operating conditions, so that the initial condition is

$$\Theta : a_1 = 1 \wedge a_2 = 1 \wedge b_1 = 1 \wedge b_2 = 1 \wedge c = 1 \ .$$

The behavior of pump x, for $x \in \{A_1, A_2, B_1, B_2\}$, is described by three transitions. The first one, *x-norm-break,* corresponds to the pump breaking under normal working conditions, i.e. when

Name	$\mathcal{E}_\tau$	Type	m_τ	γ_τ	ρ_1^τ
A_1*-norm-break*	$a_1 = 1 \wedge a_2 = 1 \wedge c = 1$	$\in \mathcal{T}_d$	1	2	$a_1' = 0$
A_1*-strain-break*	$a_1 = 1 \wedge a_2 = 0 \wedge c = 1$	$\in \mathcal{T}_d$	1	5	$a_1' = 0$
A_1*-repaired*	$a_1 = 0 \wedge c = 1$	$\in \mathcal{T}_d$	1	20	$a_1' = 1$
...	...	...	...	...	...
B_2*-repaired*	$b_2 = 0 \wedge c = 1$	$\in \mathcal{T}_d$	1	10	$b_2' = 1$
prim-shut	$a_1 = 0 \wedge a_2 = 0 \wedge c = 1$	$\in \mathcal{T}_d$	1	7	$c' = 0$
water-shut	$b_1 = 0 \wedge b_2 = 0 \wedge c = 1$	$\in \mathcal{T}_d$	1	3	$c' = 0$
restart	$c = 0$	$\in \mathcal{T}_u$	1	—	$a_1' = 1 \wedge a_2' = 1 \wedge b_1' = 1$ $\wedge b_2' = 1 \wedge c' = 1$

Table 2.1: Transitions for the reactor cooling example. To reduce clutter, we have presented only some of the pump transitions. Since transitions have only one mode, the mode probabilities have not been indicated. We recall that unprimed variables refer to the system state before a transition, and primed variables to the state immediately following the transition. Transition rates are expressed in arbitrary units.

the other parallel pump is functioning. The second transition, x*-strain-break,* corresponds to the pump breaking under the doubled workload that occurs when the parallel pump is broken. These two transitions occur at a different rate, to model the fact that the *mean time before failures* (MTBF) of a pump depends on its workload. The third transition, x*-repaired,* corresponds to the pump being repaired.

The shutdown of the system is controlled by three transitions, *prim-shut* and *water-shut.* Transition *prim-shut* corresponds to the overheating of the reactor occurring when the primary coolant is not circulating, and transition *water-shut* corresponds to the overheating of the primary coolant and reactor when the water vessel is empty. The rate for this second transition is lower, since a larger part of the system has to overheat for it to occur. Once the system is shut down, the necessary repairs are performed, and the system is restarted with transition *restart.* The delay distribution of this last transition is left unspecified, since it is not of interest in the model, and its rate may not be known with precision.

The transitions are summarized by Table 2.1. In this table, we use the previously mentioned convention of omitting from the transition formulas all variables that are not modified by a transition. Thus, the formula $a_1' = 0$ is an abbreviation for $a_1' = 0 \wedge a_2' = a_2 \wedge b_1' = b_1 \wedge b_2' = b_2 \wedge c' = c$.

Specification

We consider two properties of this reactor cooling system: a reliability property and a performance property.

A Reliability Property

The system is considered in its normal operating condition when all four pumps are working correctly; and we define "failure" as $c = 0$, i.e. when the system has been shut down. In this system, there is always a possibility of shutdown: more precisely, the shutdown state is reachable with non-zero probability from any other state. Thus, the eventual occurrence of shutdowns is inevitable: in fact, every timed trace will contain with probability 1 infinitely many alternations $c = 0, 1$.

For this reason, it is not interesting to specify a property such as "the probability of shutdowns is less than or equal to q", for some threshold q: the only value of q for which such a property holds is $q = 1$. We will instead specify that, when the system is in normal operating conditions, the expected time to a shutdown should be greater than T_0. To this end, define the abbreviation $\phi_{norm} : \ c = 1 \wedge a_1 = 1 \wedge a_2 = 1 \wedge b_1 = 1 \wedge b_2 = 1$. The specification is then expressed by the following formula:

$$A\Box\Big[\phi_{norm} \rightarrow \mathrm{D}_{>T_0}(c = 0)\Big] . \tag{2.3}$$

This formula can be read as follows. The prefix "A$\Box$" is a branching-time temporal logic construct whose meaning is "for any *reachable* system state". The *reachable* system states are the states that can occur along a timed trace: for example, the reader can verify that the state

$$c = 1,\ a_1 = 1,\ a_2 = 1,\ b_1 = 0,\ b_2 = 1$$

is reachable in our system, whereas the state

$$c = 0,\ a_1 = 1,\ a_2 = 1,\ b_1 = 0,\ b_2 = 1$$

is not. The formula $\phi_{norm} \rightarrow \mathrm{D}_{>T_0}(c = 0)$ can be read as: "if ϕ_{norm} holds, then the expected time before $c = 0$ holds is greater than T_0". The operator D is thus used to express bounds on the expected time to given system states.

A Performance Property

Another property that might be of interest is how much time the system spends overheating, on average, before a shutdown. The system overheats when $c = 1$ and either $a_1 = a_2 = 0$ or $b_1 = b_2 = 0$; not every instance of overheating leads to a shutdown, since a pump may be repaired on time. Knowing how long it overheats before a shutdown may be useful for maintenance purposes, since overheating might cause cumulative part damage.

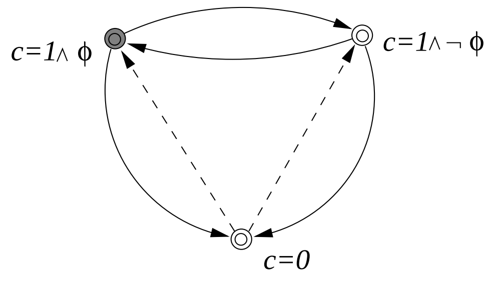

Figure 2.2: Experiment Ψ for the specification of expected overheating time during reactor cycle. Formula ϕ stands for $(a_1 = 0 \wedge a_2 = 0) \vee (b_1 = 0 \wedge b_2 = 0)$.

We call a *reactor cycle* the period that goes from one shutdown to the next. In words, the property we specify is as follows:

> *On the long run, the average time spent by the system in an overheating state during a reactor cycle is less than T_1.*

For example, this property can be used to determine that it suffices to replace certain parts only during shutoff periods, since the parts spend only a short amount of time overheating during each cycle.

To express this property, we use a special type of graph, called an *experiment.* The experiment Ψ for this property is depicted in Figure 2.2. An experiment is a graph whose vertices are labeled by formulas. The experiment is meant to be composed with the system: whenever the STS changes state, the experiment follows an edge to a vertex labeled by a formula that holds for the new state.

Every experiment vertex has an implicit self-loop. Thus, if the state of the STS does not change, the experiment remains at the same vertex. The transition rule encoded by the experiment edges and by the vertex labels must be deterministic, so that to each state transition of the STS corresponds exactly one vertex transition of the experiment.

The experiment has a distinguished set of *initial vertices,* drawn as double circles, that are taken as the starting point of the experiment. The labels of these vertices must be mutually exclusive, and their disjunction must be equivalent to *true.*

The experiment has a distinguished set of *reset* edges, drawn as dashed lines. Each time a reset edge is traversed, we say that an *experiment instance ends.* Each time an experiment instance ends, another one is immediately begun: experiments are meant to be performed an infinite number of times. When experiment Ψ of Figure 2.2 is composed with our reactor system, an instance of Ψ is ended and another one is begun each time a new reactor cycle begins.

There are two types of experiments: *P-experiments* and *D-experiments.* D-experiments are used to measure long-run average durations: Ψ is a D-experiment. The vertices of a D-experiment are

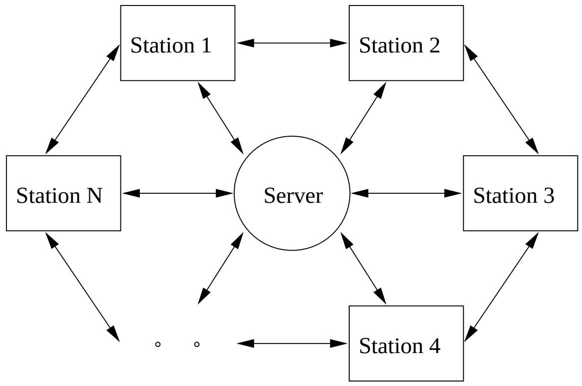

Figure 2.3: A token ring system, consisting of N stations that can access a shared server.

divided into *timed vertices,* drawn as filled circles, and *untimed vertices,* drawn as empty circles. A D-experiment measures the long-run average time spent by the system at timed vertices during an experiment instance.

In our example, experiment Ψ measures the long-run average amount of time spent overheating during a reactor cycle. With the help of experiment Ψ, our specification can be written simply as

$$\bar{\mathrm{D}}_{<T_1}(\Psi) \,, \tag{2.4}$$

and states that the long-run average amount of overheating time per reactor cycle should be less than T_1. More precisely, specification (2.4) has the following meaning:

> *Under any nondeterministic behavior of the system, with probability 1 a timed trace will exhibit a long-run average overheating time per reactor cycle less than T_1.*

The precise definition and semantics of experiments will be the subject of Chapter 5.

2.3 Application 2: A Token Ring Network

As a second example of probabilistic modeling and specification, we present a more complex application: the modeling and specification of a token-ring communication network. The network, depicted in Figure 2.3, is composed by $N > 0$ stations that can access a shared server. At most one station should be able to access to the server at any given time. To enforce mutual exclusion, a token circulates along the ring: a station can access the server only when in possession of the token.

Additionally, both server and stations can fail. Once they have failed, they stop taking part in the protocol until they start functioning again. If a station fails, it stops accessing the server; if a server fails, no station can access it. If a station fails while in possession of the token or while accessing the server, the token is lost. When the protocol senses that no token is present, it activates a leader-election algorithm to create a new token. The algorithm first checks that at most one station is down (so that the ring is not partitioned), and then creates a new token.

We present this system for a generic number $N > 0$ of stations; however, the verification methods presented in this dissertation can be applied only to instances of the system corresponding to fixed values of N.

2.3.1 STS for Token Ring

The status of each station i is described by the following variables, for $1 \leq i \leq N$:

- Variable l_i assumes value 0 if the station is idle, value 1 if it is waiting to access the server, and 2 if it is accessing the server.
- Variable t_i assumes value 0 if the station does not have the token, and 1 if it does.
- Variable f_i assumes value 0 if the station is functioning, and 1 if it has failed.

The status of the server is described by the following variables:

- Variable a assumes value 0 if the server is idle, and 1 if it is being accessed.
- Variable d assumes value 0 if the server is functioning, and 1 if it has failed.

The set of variables of the STS is thus $\mathcal{V} = \{a, d\} \cup \bigcup_{i=1}^{N} \{l_i, t_i, f_i\}$. The initial condition is

$$\Theta : d = 0 \wedge a = 0 \wedge \left(\bigwedge_{i=1}^{N} l_i = 0 \right) \wedge \left(\bigwedge_{i=1}^{N} f_i = 0 \right) \wedge \sum_{i=1}^{N} t_i = 1 \ ,$$

so that the initial position of the token is not specified.

The transitions of the STS can be divided into three categories: the transitions that describe the normal operation of the ring, the transitions that describe the failure of the server or stations, and the transitions that describe the recovery from failure. There are no fair transitions in this STS: $\mathcal{T}_f = \emptyset$. Moreover, all transitions have a single mode, i.e. $m_\tau = 1$ for all τ. To avoid clutter, when describing transition τ we write $\mathcal{E}$ instead of $\mathcal{E}_1^\tau$, and similarly for the other components. Moreover, we do not specify the rates of transitions with exponential delay distributions, since we are discussing a generic token-ring example.

Normal ring operation. The transitions that describe the normal ring operation are as follows.

- For $1 \leq i \leq N$, transition $\tau_i^1 \in \mathcal{T}_e$ is taken when a station goes from idle to requesting access to the server. The transition has $\mathcal{E} : f_i = 0 \wedge l_i = 0$ and $\rho : l_i' = 1$.

- For $1 \leq i \leq N$, transition $\tau_i^2 \in \mathcal{T}_e$ is taken when the token is passed from station i to station $i+1$. It has

$$\mathcal{E} : t_i = 1 \wedge f_i = 0 \wedge f_{(i \bmod N)+1} = 0 \qquad \rho : t_i' = 0 \wedge t'_{(i \bmod N)+1} = 1 \ .$$

  Note that this transition is enabled even when station i is requesting access to the server.

- For $1 \leq i \leq N$, transition $\tau_i^3 \in \mathcal{T}_i$ is taken when station i, having received the token, starts accessing the server. It has

$$\mathcal{E} : t_i = 1 \wedge l_i = 1 \wedge f_i = 0 \wedge d = 0 \wedge a = 0 \qquad \rho : t_i' = 0 \wedge l_i' = 2 \wedge a' = 1 \ .$$

  Note that if this transition is enabled, it is taken immediately, preventing the previous transition from passing the token to the next station.

- For $1 \leq i \leq N$, transition $\tau_i^4 \in \mathcal{T}_e$ is taken when station i terminates the access to the server. It has

$$\mathcal{E} : l_i = 2 \wedge f_i = 0 \wedge d = 0 \wedge a = 1 \qquad \rho : l_i' = 0 \wedge a' = 0 \wedge t_i' = 1 \ .$$

Server and station failures. The transitions that describe the server and station failures are as follows.

- For $1 \leq i \leq N$, transition $\tau_i^5 \in \mathcal{T}_e$ is taken when station i fails. It has $\mathcal{E} : f_i = 0$ and $\rho : f_i' = 1$.

- Transition $\tau^6 \in \mathcal{T}_e$ is taken when the server fails. It has $\mathcal{E} : d = 0$ and $\rho : d' = 1$.

- For $1 \leq i \leq N$, transition $\tau_i^7 \in \mathcal{T}_i$ is taken when station i detects a server failure while accessing it. Station i then goes back to the state in which it is requesting access, and lets the token circulate. It has

$$\mathcal{E} : l_i = 2 \wedge f_i = 0 \wedge d = 1 \qquad \rho : l_i' = 1 \wedge t_i' = 1 \ .$$

- Transition $\tau^8 \in \mathcal{T}_i$ is taken when the server detects that the station accessing it has failed. It has

$$\mathcal{E} : l_i = 2 \wedge f_i = 1 \wedge a = 1 \qquad \rho : a' = 0 \ .$$

Failure recovery. The transitions that describe recovery from failure are as follows.

- For $1 \leq i \leq N$, transition $\tau^9 \in \mathcal{T}_u$ is taken when station i recovers from failure. It is a transition with unspecified delay distribution, since we do not wish to make assumptions about a station's recovery time. It has

$$\mathcal{E} : f_i = 1 \qquad \rho : f_i' = 0 \wedge t_i' = 0 \wedge (l_i > 0 \rightarrow l_i' = 1) \wedge (l_i = 0 \rightarrow l_i' = 0) \; .$$

- Transition $\tau^{10} \in \mathcal{T}_u$ is taken when the server recovers from failure. Again, it has unspecified delay distribution, since we do not wish to make assumptions about a server's recovery time. It has
$$\mathcal{E} : d = 1 \qquad \rho : d' = 0 \wedge a' = 0 \; .$$

- Transition $\tau^{11} \in \mathcal{T}_e$ is taken when the functioning stations detect that no token is circulating, and that no more than one station has failed. This last requirement ensures that the ring is not partitioned. Under these circumstances, the functioning stations execute a leader-election algorithm that creates a new token. Since we do not wish to make assumptions on which specific leader-election algorithm is used, we write a nondeterministic transition relation ρ, so that the station at which the new token is created is selected nondeterministically. This transition has
$$\mathcal{E} : \bigwedge_{i=1}^{N} (f_i = 0 \rightarrow t_i = 0) \wedge 1 \geq \sum_{i=1}^{N} f_i$$
$$\rho : 1 = \sum_{i=1}^{N} [t_i'(1 - f_i')] \wedge \bigwedge_{i=1}^{N} (f_i' = f_i) \; .$$

2.3.2 Specification of Token Ring

A Performance Requirement

The first property we consider is a performance requirement, which can be informally stated as follows:

> *When station 1 requests access to the resource, it is granted access in average time less than T_0, barring system failures.*

We construct a formal specification for this requirement by stages. Our first attempt is to write the following formula, in analogy with (2.3):

$$\mathrm{A}\Box\Big[l_1 = 1 \rightarrow \mathrm{D}_{<T_0} l_1 = 2\Big] \; . \tag{2.5}$$

This formula can be read as follows: "every reachable state in which $l_1 = 1$ is followed by a state in which $l_1 = 2$ in average time less than T_0". The problem with this specification is that there are many states at which $l_1 = 1$: in some of them, the token is already at 1 and both the server and the stations are functional; in others, the server or some station might have failed, or the token might be

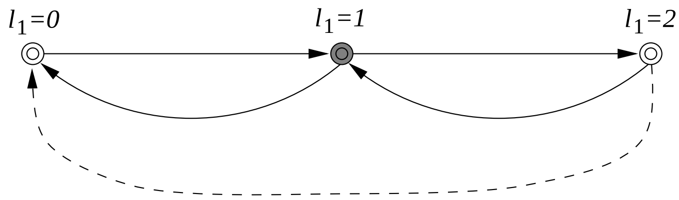

Figure 2.4: Experiment Ψ_0 for the token ring system. In this and in the following experiment, we omit edges that cannot be followed when the composition between the experiment and the token ring STS is computed.

very far away from station 1. Formula (2.5) is true if *all* these states are followed in average time less than T_0 by a state in which $l_1 = 2$: it thus takes a worst-case approach. Our informal specification refers instead to the expected time required to access the server, averaged on many request-access cycles.

As we have seen in the previous example, experiments can be used to express specifications that refer to the long-run average behavior of systems. Our second attempt to the specification of this property uses the experiment Ψ_0 depicted in Figure 2.4. This experiment measures the long-run average time elapsing between the transition $l_1 : 0 \to 1$ and $l_1 : 1 \to 2$. With the help of this experiment, the specification can be expressed by

$$\bar{\mathrm{D}}_{<T_0}(\Psi_0) \ . \tag{2.6}$$

This specification is almost a faithful encoding of our informal requirement, except that it does not take into account the "barring system failures" provision: experiment Ψ_0 measures the time required to access the server even when the server or some station has failed. Since the recovery time of server and stations is not specified, specification (2.6) is not appropriate. Our final specification uses the experiment Ψ_1 depicted in Figure 2.5. This experiment measures the time from $l_1 : 0 \to 1$ to $l_1 : 1 \to 2$ only when the server and all stations are functioning. The specification is then expressed by

$$\bar{\mathrm{D}}_{<T_0}(\Psi_1) \ . \tag{2.7}$$

For any specific value of $N > 0$, this specification can be checked with the methods developed in this dissertation.

A Reliability Requirement

Our first requirement for the token-ring system regarded the average time to gain access to the server, without counting the time elapsed during failures. Our second requirement concerns the failure rate of the system, and can be informally phrased as follows:

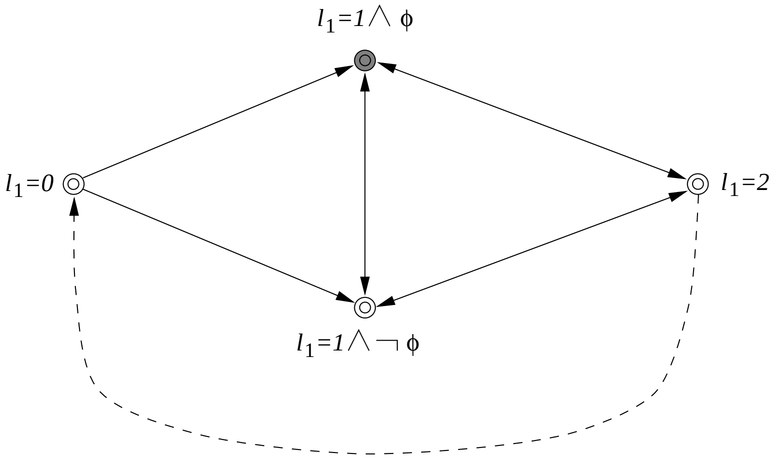

Figure 2.5: Experiment Ψ_1 for token-ring system. Formula ϕ is an abbreviation for $d + \sum_{i=1}^{N} f_i = 0$, and indicates that the server and all stations are functioning. Note the use of bi-directional edges, as a shorthand for two uni-directional ones.

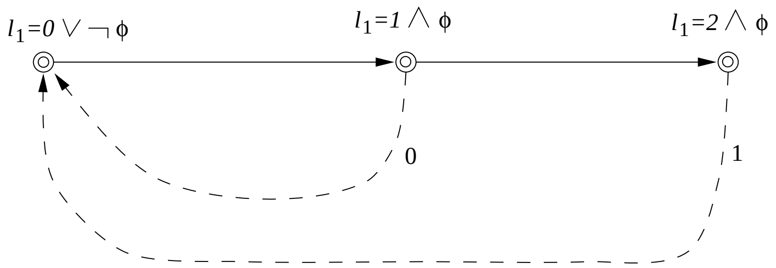

Figure 2.6: Experiment Ψ_2 for token-ring system. Formula ϕ is an abbreviation for $d + \sum_{i=1}^{N} f_i = 0$, and indicates that the server and all stations are functioning. The reset edges are labeled with their outcome, which in this case is either 0 or 1.

> *If station 1 requests access to the server when the system is fully functioning, on the long run it will gain access to the server without intervening system failures with probability at least p_0.*

We specify this requirement with the help of the experiment Ψ_2 depicted in Figure 2.6. This experiment, unlike the previous ones, is a *P-experiment.* In a P-experiment, to each reset edge is associated an *outcome:* a real number representing the success or failure of the experiment. A P-experiment is used to measure the long-run average outcome received during the system's behavior. As we can see from Figure 2.6, experiment Ψ_2 "starts" when there is a transition $l_1 : 0 \rightarrow 1$ while the token-ring system is fully functioning. The experiment yields outcome 0 if there is a system failure while l_1 is still equal to 1, and yields outcome 1 if l_1 reaches value 2 before a failure. Thus, the long-run average outcome of the experiment corresponds to the fraction of access requests from station 1 that are followed by a server access without an intervening server or station failure. We can then specify our requirement with the formula $\bar{\mathrm{P}}_{\geq p_0}(\Psi_2)$. Again, the precise semantics of P-experiments will be discussed in Chapter 5.

2.4 Concluding Remarks

Through these examples, we have seen that STS can be used to model probabilistic systems for both reliability and performance analysis, and that the logics presented in this dissertation can encode several properties of interest. To conclude this chapter, we present some general observations about probabilistic verification that can be made at this point.

Long-run average properties vs. single event properties. Many interesting properties of probabilistic systems, including performance and reliability properties that have long been studied outside of the field of formal verification, can be captured by *long-run average* probabilistic properties, rather than *single-event* ones. One of the main points of this dissertation is to make the specification and verification of these long-run average properties possible.

Use of nondeterminism. Nondeterminism can be used to represent lack of knowledge, or lack of assumptions, about some system characteristics. However, excessive use of nondeterminism can prevent desired system properties from holding. When the verification of a probabilistic specification fails, it is often necessary to analyze the system behaviors that violate the property, to determine whether excessive use of nondeterminism, incorrect specification or real system malfunction is at the root of the problem.

Writing meaningful specifications. Writing meaningful probabilistic specifications is tricky, more so than writing non-probabilistic specifications. The specification languages presented in this dissertation have been designed to facilitate the writing of meaningful specifications.

Chapter 3

Timed Probabilistic Systems

While stochastic transition systems provide the high-level language in which probabilistic systems are described, the logics and model-checking algorithms are defined with respect to the low-level model of *timed probabilistic systems* (TPSs). Essentially, the relation between a stochastic transition system and its corresponding TPS is similar to the relation between a fair transition system and its representation in terms of state transition graph. Stochastic transition systems, like fair transition systems, are first-order structures providing a high-level, compositional description of a system; TPSs provide the underlying computational model. TPSs are also in many ways more general than stochastic transition systems, so that verification techniques that can be applied directly to TPSs are of independent interest. The translation from stochastic transition systems to TPSs will be presented later in the dissertation, since it relies on many advanced concepts, such as probabilistic fairness, that will not be available until then.

The definition of TPSs is based on *Markov decision processes.* These models were introduced by Bellman [Bel57] and Howard [How60] to represent stochastic processes with control inputs. Traditionally, the inputs represent controls applied to the stochastic systems, or decisions that influence the system's behavior. In our application, the controls will represent the nondeterminism, which is superimposed to the underlying probabilistic behavior of the system. Our basic system models, *Timed Probabilistic Systems,* consists simply in Markov decision processes with additional information to represent the timing behavior.

After introducing Markov decision processes and timed probabilistic systems, we develop a connection between the graph-theoretical properties of Markov decision processes and their stochastic behavior. This connection, based on the concept of *end component,* is central to the development of much of the later material of the dissertation, including the model-checking algorithms for the logics that will be presented.

We conclude the chapter by describing two optimization problems on Markov decision processes that will be used as components of the model-checking algorithms throughout the dissertation:

the *stochastic shortest path* problem and the *maximum reachability probability* problem. We will provide a concise summary of results for these two problems, as well as an improved algorithm for the computation of maximal reachability probabilities.

3.1 Markov Decision Processes and Policies

A *Markov decision process* (MDP) is a generalization of a Markov chain in which a set of possible actions is associated to each state. To each state-action pair corresponds a probability distribution on the states, which is used to select the successor state [Der70, Ber95]. A Markov chain corresponds thus to a Markov decision process in which there is exactly one action associated with each state. Throughout the dissertation, we assume the existence of a fixed set *Acts* of actions, containing all the actions of interest. The definition of Markov decision processes is as follows.

Definition 3.1 (Markov decision process) A *Markov decision process* (MDP) (S, A, p) consists of a finite state S of states, and two components A, p that specify the transition structure:

- For each $s \in S$, $A(s) \subseteq Acts$ is the non-empty finite set of *actions* available at s.
- For each $s, t \in S$ and $a \in A(s)$, $p_{st}(a)$ is the probability of a transition from s to t when action a is selected. For every $s, t \in S$ and $a \in A(s)$, it is $0 \leq p_{st}(a) \leq 1$ and $\sum_{t \in S} p_{st}(a) = 1$. ∎

Throughout this dissertation, we will consider only the verification of *finite* Markov decision processes, that is, Markov decision processes having finite sets of states and actions. All the results we state, in particular those concerning the model-checking algorithms, are claimed to be valid for finite Markov decision processes only, unless otherwise noted.

We will often associate with a Markov decision process additional labelings, to represent various quantities of interest, such as the time required to complete each state transition. These additional labelings will be simply added to the list of components of a Markov decision process.

3.1.1 Behaviors

A *behavior* of a Markov decision process is an infinite sequence of alternating states and actions, constructed by iterating a two-phase selection process. First, given the current state s, an action $a \in A(s)$ is selected nondeterministically; second, the successor state t of s is chosen according to the probability distribution $\Pr(t \mid s, a) = p_{st}(a)$. The formal definition of behavior is as follows.

Definition 3.2 (behaviors of MDP) A *behavior* of an MDP Π is an infinite sequence $\omega : s_0 a_0 s_1 a_1 \cdots$ such that $s_i \in S$, $a_i \in A(s_i)$ and $p_{s_i, s_{i+1}}(a_i) > 0$ for all $i \geq 0$. Given a state s, we indicate by Ω_s the set of behaviors originating from s. We also let X_i, Y_i be the random variables representing the i-th state and the i-th action along a behavior, respectively. Formally, X_i and Y_i are variables that assume the value s_i, a_i on the behavior $\omega : s_0 a_0 s_1 a_1 \cdots$. ∎

A *state-action pair* is a pair composed of a state and of an action associated with the state.

Definition 3.3 (state-action pairs, Succ**)** A *state-action pair* is a pair s, a such that $s \in S$, $a \in A(s)$. We denote by $\chi_\Pi = \{(s,a) \mid s \in S \wedge a \in A(s)\}$ the set of state-action pairs of an MDP Π. For any state-action pair s, a, we define $\text{Succ}(s,a) = \{t \mid p_{st}(a) > 0\}$ to be the set of possible successors of s when a is chosen. ∎

Measurable sets of behaviors. For every state $s \in S$, we let $\mathcal{B}_s \subseteq 2^{\Omega_s}$ be the smallest algebra of subsets of Ω_s that contains all the *basic cylinder sets*

$$\{\omega \in \Omega_s \mid X_0 = s_0 \wedge Y_0 = a_0 \wedge \cdots \wedge X_n = s_n \wedge Y_n = a_n\}$$

for all $n \geq 0$, $s_0, \ldots, s_n \in S$, $a_0 \in A(s_0), \ldots, a_n \in A(s_n)$, and that is closed under complement and countable unions and intersections. This algebra is called the *Borel σ-algebra* of basic cylinder sets, and its elements are the *measurable* sets of behaviors, to which it will be possible to assign a probability [KSK66, Wil91].

3.1.2 Policies and Probabilities of Behaviors

To be able to talk about the probability of behaviors, we would like to associate to each $\Delta \in \mathcal{B}_s$ its probability measure $\Pr(\Delta)$. However, this measure is not well-defined, since the probability that a behavior $\omega \in \Omega_s$ belongs to Δ depends on how the actions along the behavior have been nondeterministically chosen.

To represent these choice criteria, we use the concept of *policy* (see Derman [Der70]). Policies are closely related to the adversaries of Segala and Lynch [SL94] and Segala [Seg95b], and to the schedulers of Lehman and Rabin [LR81], Vardi [Var85] and Pnueli and Zuck [PZ86]. A policy η is a set of conditional probabilities

$$Q_\eta(a \mid s_0 s_1 \cdots s_n) , \tag{3.1}$$

for all $n \geq 0$, all possible sequences of states $s_0 \cdots s_n$, and all $a \in A(s_n)$. It must be

$$0 \leq Q_\eta(a \mid s_0 s_1 \cdots s_n) \leq 1 \qquad \sum_{a \in A(s_n)} Q_\eta(a \mid s_0 s_1 \cdots s_n) = 1$$

for all $n \geq 0$, all sequences $s_0 \cdots s_n$ and all $a \in A(s_n)$. A policy dictates the probabilities with which the actions are chosen: according to policy η, after the finite sequence of states $s_0 s_1 \cdots s_n$ starting at the root $s = s_0$ of Ω_s, action $a \in A(s_n)$ is chosen with probability $Q_\eta(a \mid s_0 s_1 \cdots s_n)$.

Hence, under policy η the probability of a direct transition to $t \in S$ after $s_0 \cdots s_n$, denoted by

$\mathrm{Pr}^{\eta}_{s_0}(t \mid s_0 \cdots s_n)$, is given by

$$\sum_{a \in A(s_n)} p_{s_n,t}(a)\, Q_\eta(a \mid s_0 s_1 \cdots s_n)\ .$$

The probability of following a finite behavior prefix $s_0 a_0 s_1 a_1 \ldots s_n$ of states starting at the root $s = s_0$ of Ω_s under policy η is therefore equal to

$$\prod_{i=0}^{n-1} p_{s_i,s_{i+1}}(a_i)\, Q_\eta(a_i \mid s_0 \cdots s_i)\ .$$

These probabilities for the finite sequences give rise to a unique probability measure Pr^{η}_{s} on $\mathcal{B}_s$ that associates with each $\Delta \in \mathcal{B}_s$ its probability $\mathrm{Pr}^{\eta}_{s}(\Delta)$ (see Doob [Doo94], Kemeny, Snell and Knapp [KSK66], or Williams [Wil91] for more details on the extension of probability measures from cylinder sets to σ-algebras).

Recall that an *event* $\mathcal{A}$ is a measurable set of behaviors $\mathcal{A} \in \mathcal{B}_s$ for some s, and a *random variable* is a measurable function [Wil91]. Following the usual notation, we denote by $\mathrm{Pr}^{\eta}_{s}(\mathcal{A})$ the probability of event $\mathcal{A}$ under policy η starting from state s, and we denote with $\mathrm{E}^{\eta}_{s}\{X\}$ the expected value of the random variable X under policy η starting from state s; conditional probabilities and expectations are denoted in the usual way.

Markovian and deterministic policies. As usual, we say that a policy is *Markovian* if its choice of actions at a state does not depend on the portion of behavior before the state is reached. These policies are also called *memoryless,* to emphasize the lack of dependence from the past portion of the behavior.

Definition 3.4 (Markovian and deterministic policies) We say that a policy is *Markovian* if

$$Q_\eta(a \mid s_0 \ldots s_n) = Q_\eta(a \mid s_n)$$

for all $n \geq 0$ and all sequences $s_0 \ldots s_n$ of states of S. We denote by $\boldsymbol{\eta}$ the set of all policies, and by $\boldsymbol{\eta}_M \subseteq \boldsymbol{\eta}$ the set of all Markovian ones.

We say that a policy η is *deterministic* if η is Markovian, and if for each s there is an action $a \in A(s)$ such that $Q_\eta(a \mid s) = 1$. We denote with $\boldsymbol{\eta}_D$ the set of all deterministic policies. Note that if the sets of states and actions of the MDP are both finite, the set $\boldsymbol{\eta}_D$ is also finite. ∎

3.1.3 On the Notion of Policy (‡)

According the definition of policy we have adopted, the choice of action at a state can depend on the past sequence of states, but not on the past sequence of actions, as indicated by (3.1). Instead of definition (3.1), we could have adopted the more general form

$$Q_\eta(a \mid s_0 a_0 s_1 a_1 \cdots s_n) \tag{3.2}$$

for $n \geq 0$, which would enable the choice of action to depend on the past sequence of actions as well.

The motivation for our choice is essentially pragmatic. The form (3.1) for a policy is commonly used in the literature on Markov decision processes (see, for example, Derman [Der70] and Puterman [Put94]). By adopting it, we will be able to use results from the classical theory of Markov decision processes without having to provide new or revised proofs. In any case, most of the results that will be presented in this dissertation, and in particular all the logics and model-checking algorithms, are not affected by whether form (3.1) or (3.2) is chosen to define the policies. The reader can verify that all the proofs of the statements presented need only minor syntactic corrections when one form is replaced by the other.

Policies and decision rules. In the literature on Markov decision processes, another definition of policy is common. This definition relies on the concept of *decision rule:* a decision rule δ is a function that assigns to each state s a probability distribution $\{\delta_s(a)\}_{a \in A(s)}$ on the actions associated with s, such that $\sum_{a \in A(s)} \delta_s(a) = 1$. A policy η is then defined as an infinite sequence $\delta_0, \delta_1, \delta_2, \ldots$ of decision rules: after following the sequence of states $s_0, s_1, \ldots, s_n$, decision rule δ_n is used to select the next action. This is the approach adopted, for example, in Bertsekas [Ber95].

This notion of policy is more restrictive than the one adopted in this dissertation: the choice of action cannot depend on the past sequence of states, but only on the length of such a sequence. The main appeal of the definition of policies in terms of decision rules is the notational simplicity, evident especially in a matrix-theoretic treatment of the theory of Markov decision processes. Unlike (3.2), the adoption of this definition of policy in the dissertation would require changes in the statements and proofs of several results. Due to the lesser generality of this notion of policy and to these technical difficulties, we have not followed this approach.

3.1.4 Drawing Markov Decision Processes

To depict an MDP $\Pi = (S, A, p)$, we draw a graph that has the set S of states as vertices. For every $s \in S$ and $a \in A(s)$, we draw a *bundle* of edges to each $t \in S$ such that $p_{st}(a) > 0$. The edges belonging to the same bundle are grouped together by a small circle sector (similarly to the notation used to depict and-or graphs), and each bundle is labeled by the corresponding action. The transition probability $p_{st}(a)$, if indicated, labels the edge from s to t belonging to bundle a. Additional state or action labels, if present, can be represented next to the components they label.

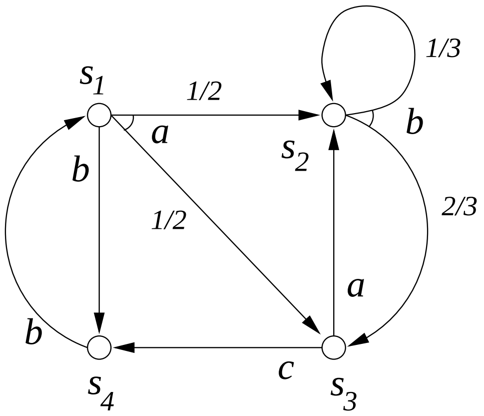

Figure 3.1: Graphical representation of Markov decision processes.

	$t =$	s_1	s_2	s_3	s_4
$p_{s_1,t}(a) =$		0	1/2	1/2	0
$p_{s_1,t}(b) =$		0	0	0	1
$p_{s_2,t}(b) =$		0	1/3	2/3	0
$p_{s_3,t}(a) =$		0	1	0	0
$p_{s_3,t}(c) =$		0	0	0	1
$p_{s_4,t}(b) =$		1	0	0	0

Table 3.1: Transition probabilities of the MDP represented in Figure 3.1.

Example 3.1 (drawing an MDP) Figure 3.1 depicts an MDP $\Pi = (S, A, p)$ having $S = \{s_1, s_2, s_3, s_4\}$, $A(s_1) = \{a, b\}$, $A(s_2) = \{c\}$, $A(s_3) = \{a, c\}$, $A(s_4) = \{b\}$. The transition probabilities are given in Table 3.1. Note that deterministic probability distributions are usually not indicated in the diagrams. ∎

3.1.5 Notation: some Events and Random Functions

It is convenient to introduce some abbreviations for events and random functions that will be often used throughout the dissertation. First, we define reachability.

Definition 3.5 (reachability in MDP) Given an MDP (S, A, p), we say that a state t is reachable from a state s if there is a path from s to t in the graph (S, ρ_S). We say that a subset $T \subseteq S$ is reachable from a subset $U \subseteq S$ if there are two states $t \in T$, $u \in U$ such that t is reachable from u. Reachability of a state from a set of states, and of a set of states from a state, are similarly defined.

We say that the MDP is *strongly connected* if every state of S is reachable from every other state of S. This is equivalent to the fact that the graph (S, ρ_S) is strongly connected. ∎

Next, we introduce a shorthand for the event of reaching a specified subset of states.

Definition 3.6 (event *reach*) Given a subset T of states, we define the event $reach(T)$ to be equivalent to $\exists k \,.\, X_k \in T$. Thus, $reach(T)$ is true of a behavior iff the behavior reaches the subset T of states. ∎

We also introduce a shorthand for the set of state-action pairs that occur infinitely often along a behavior.

Definition 3.7 (the random function *inft*) Given a behavior ω indicate by

$$inft(\omega) = \{(s, a) \mid \overset{\infty}{\exists}\, k \,.\, X_k = s \wedge Y_k = a\} ,$$

the state-action pairs that occur infinitely often along it, where $\overset{\infty}{\exists}$ is a quantifier with the meaning "there are infinitely many". ∎

Last, we define a shorthand for the event of following a behavior prefix.

Definition 3.8 (prefix following event) Let $\sigma = s_0 a_0 s_1 \cdots s_n a_n$, for $n \geq 0$, be a finite prefix of behavior. We use σ also to denote the event

$$\bigwedge_{k=0}^{n} (X_k = s_k \wedge Y_k = a_k)$$

of following σ during the first k steps of a behavior, and we write $\Pr^{\eta}(\sigma)$ for the corresponding probability. Note that it is not necessary to specify the initial state, since it is implicit in σ. We introduce an analogous notation for the case in which σ terminates with an action. ∎

3.2 Timed Probabilistic Systems

A *timed probabilistic system* (TPS) is a Markov decision process with three additional labelings that describe the set of initial states, the timing properties of the system, and the values of a set of state variables at all system states. For simplicity, when discussing TPSs we will assume a fixed set $\mathcal{V}$ of state variables.

Definition 3.9 (TPS) A TPS $\Pi = (S, A, p, S_{in}, \mathit{time}, \mathcal{I})$ is an MDP with three additional components:

- A subset $S_{in} \subseteq S$ of initial states.
- A labeling *time*, which associates to each $s \in S$ and $a \in A(s)$ the *expected* amount of time $\mathit{time}(s, a) \in \mathbb{R}^+$ spent at s when action a is selected.
- A labeling $\mathcal{I}$, which associates to each state $s \in S$ and variable $x \in \mathcal{V}$ the value $\mathcal{I}_s[\![x]\!]$ that x assumes at s. ∎

TPSs are closely related to the *semi-Markov decision processes* introduced in Howard [How60, How63] and De Cani [DC64] and later studied, among others, by Veinott [Vei69] and Ross [Ros70b, Ros70a]. In a semi-Markov decision process, to each state and action is associated the exact or expected amount of time spent at the state when the action is chosen.

3.2.1 Time Divergence and Non-Zenoness

Given a behavior prefix $s_0, a_0, \ldots, s_n$, the expected time elapsed along the prefix is given by $\sum_{k=0}^{n-1} \mathit{time}(s_k, a_k)$. Similarly, we say that the time *diverges* along a behavior iff $\sum_{k=0}^{\infty} \mathit{time}(X_k, Y_k)$ does.

Since our systems are finite-state, time does not diverge along a behavior iff there is K such that $\mathit{time}(X_k, Y_k) = 0$ for all $k \geq K$. These behaviors do not have a physical meaning: thus, we would like to ensure that time diverges with probability 1 along the behaviors. We adopt this requirement, rather than the stricter one of divergence along all behaviors, since this latter requirement can complicate the construction of system models with its strictness. In a later chapter, we will discuss an alternative approach to time divergence, due to Segala [Seg95b].

Formally, we say that a TPS Π is *non-Zeno* if the elapsed time along a behavior diverges with probability 1.

Definition 3.10 (non-Zeno TPS) A TPS Π is *non-Zeno* if

$$\Pr_s^\eta\Big(\sum_{i=0}^{\infty} \mathit{time}(X_i, Y_i) = \infty\Big) = 1$$

for any policy η, and for any initial state $s \in S_{in}$. ∎

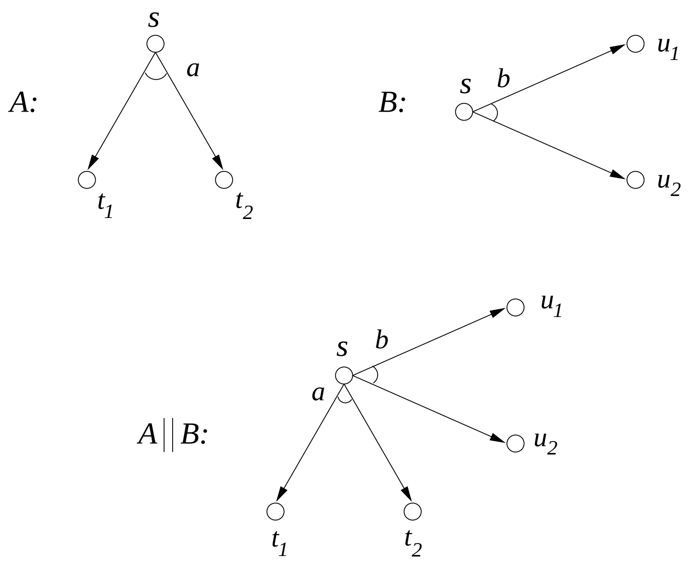

Figure 3.2: Portions of two modules A and B and of their parallel composition, in the interleaving semantics. In this example, nondeterminism is used to abstract from the scheduling mechanisms of the two modules.

We will apply our specification languages and model-checking algorithms only to TPSs that satisfy the following *non-Zenoness assumption.*

Assumption 3.1 (non-Zenoness) The TPS is non-Zeno. ∎

In the next chapter, we will provide algorithms to check whether this assumption holds.

3.2.2 The Role of Nondeterminism

As stated in the introduction, nondeterminism can be used for several purposes. One of these purposes is to represent concurrency and unknown scheduling mechanisms, as illustrated by the following example.

Example 3.2 (nondeterminism and concurrency) Consider a system composed of two modules A and B. As depicted in Figure 3.2, each of the two modules A and B can execute a transition from a given state s, each yielding a given probability distribution for the successor state. If we choose to represent parallel composition by interleaving (the approach taken, for example, in [MP91]), we can use the nondeterminism to leave the scheduling mechanism between A and B unspecified, in complete analogy with the approach based on fair transition systems. The transitions out of state s in the parallel composition of A and B are then as depicted in Figure 3.2. ∎

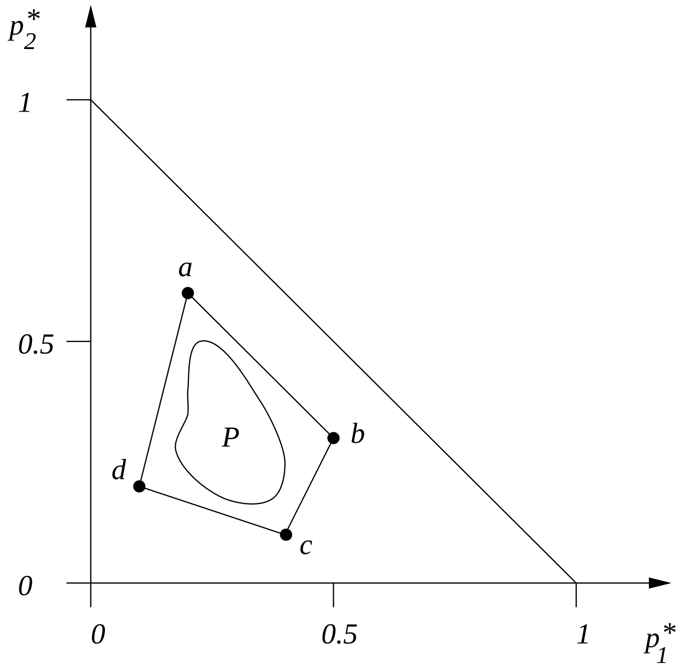

Figure 3.3: Set P of possible values for the probability distribution $[p_1^*, p_2^*, p_3^*]$, and its approximation by a set of 4 actions a, b, c and d, whose probability distributions are also depicted. The horizontal axis represents the values of p_1^*, the vertical axis those of p_2^*; probability p_3^* is given by $p_3^* = 1 - (p_1^* + p_2^*)$.

Nondeterminism can also be used to represent transitions whose next-state probability distribution or expected completion time is not known with precision. Since a system is considered correct if it satisfies the specification under *all* possible policies, as we will see in the next chapter, the idea is to ensure the existence of at least one policy which gives rise to the true probability distribution or expected times. Note that it is not important that this policy be known in advance. The following examples illustrate this use of nondeterminism and policies.

Example 3.3 (nondeterminism and uncertainty in the probabilities) Consider a transition that can lead from a state s to one of the destination states t_1, t_2, t_3. For $1 \leq i \leq 3$, let p_i^* be the probability that the transition from s leads to t_i. Assume that the precise value of the vector $[p_1^*, p_2^*, p_3^*]$ is not known, but it is known to lie inside a region P of possible values, as illustrated in Figure 3.3. To represent this knowledge about the transition probabilities, and to obtain a conservative model for the system, we associate with state s four actions a, b, c and d. The probability distributions associated with these actions are given in Table 3.2

Once at s, a policy is free to choose the probabilities q_a, q_b, q_c, q_d with which the actions are selected, so that the probability of a transition to t_i, for $1 \leq i \leq 3$, is given by the linear combination

Action	$p_{s,t_1}(\cdot)$	$p_{s,t_2}(\cdot)$	$p_{s,t_3}(\cdot)$
a	0.2	0.6	0.2
b	0.5	0.3	0.2
c	0.4	0.1	0.5
d	0.1	0.2	0.7

Table 3.2: Transition probabilities of actions a, b, c, d.

$q_a p_{st_i}(a) + \cdots + q_d p_{st_i}(d)$. By appropriately choosing the probabilities which which actions a, b, c, d are chosen, a policy can give rise to all the probability distributions for transitions to t_1, t_2, t_3 that lie inside the polygon delimited by a, b, c, d in Figure 3.3. This polygon includes the region P where the true distribution is known to lie. Thus, the model represents a conservative approximation of the system. ∎

Example 3.4 (nondeterminism and uncertainty in the expected times) Assume that the system, from state s, always takes a transition to state t. The expected time of this transition is not known with precision, but it is known to lie in the interval $[3, 5]$. In analogy with the previous example, this situation can be modeled by associating with s two actions a and b, that lead deterministically to t and have expected times $\mathit{time}(s, a) = 3$ and $\mathit{time}(s, b) = 5$. ∎

The two uses of nondeterminism to represent imprecise knowledge illustrated by the above examples can be combined.

We note that in the above examples the different actions available at a state do not represent different system events, but rather represent the same event, occurring with different characteristics (probability distribution, or expected time). In other words, the different actions are used as an artifact to represent uncertain knowledge. It is this use of nondeterminism, which is quite common in the modeling of real systems, which has led us in this dissertation to de-emphasize the role of actions in the specification languages. In particular, the specification languages we consider cannot refer to the names of the actions, but only to their expected duration and effects.

3.2.3 Expected Time vs. Exact Time

Given a state s of a TPS and an action $a \in A(s)$, the quantity $\mathit{time}(s, a)$ represents the *expected time* for which the system stays at state s under action a, rather than the exact time. This choice represents a compromise between the expressive power of the model and the complexity of the verification algorithms.

Since the probability distribution of transition times is not known, we cannot verify *probabilistic deadline properties,* i.e. properties stating that the probability of meeting a timing deadline is above or below a specified threshold.[1]

The model-checking algorithms that have been proposed so far for the verification of probabilistic deadline properties rely on the explicit or implicit introduction of *clocks* (or *counters*) to measure the time to the deadline. Such algorithms have been described, for example, in Hansson and Jonsson [HJ89, HJ94], Hansson [Han94], Beauquier and Slissenko [BS96] and de Alfaro [dA97]. There are two drawbacks with this approach. First, these algorithms can be applied only to systems in which time can assume only integer values, due to their reliance on clocks. Moreover, the introduction of these clocks causes a large increase in the size of the state-space of the system. This increase is proportional to the specified distance to the deadline (measured in clock-ticks): it is thus computationally impractical to verify probabilistic deadline properties of non-trivial systems.

The approach adopted in this dissertation enables us to consider system models in which time is modeled by the set of non-negative real numbers. While we cannot consider probabilistic deadline properties, the specification languages we introduce can express properties related to the average time before events, and to the average system performance and reliability. By avoiding the introduction of clocks, it leads to efficient model-checking algorithms. Our approach is more closely related to the point of view of performance evaluation and reliability analysis, rather than to the study of (non-probabilistic) real-time systems.

3.3 End Components

Many of the results presented in this dissertation rest upon a connection between the stochastic properties of an MDP and its structure when viewed as a graph-like structure. This connection is based, to a large extent, on the definition of *end components*. To define end components, we first introduce state-action sets and *sub-MDPs*.

Definition 3.11 (state-action sets and sub-MDP) Given an MDP $\Pi = (S, A, p)$, a *state-action set* is a subset $\chi \subseteq \{(s,a) \mid s \in S \wedge a \in A(s)\}$. A *sub-MDP* of Π is a pair (C, D), where $C \subseteq S$ and D is a function that associates to each $s \in C$ a set $D(s) \subseteq A(s)$ of actions. Clearly, there is a one-to-one relation between sub-MDPs and state-action sets:

- given a state-action set χ, we denote by $sub(\chi) = (C, D)$ the sub-MDP defined by

$$C = \{s \mid \exists a \,.\, (s,a) \in \chi\} \qquad D(s) = \{a \mid (s,a) \in \chi\}$$

- given a sub-MDP (C, D), we denote by $sa(C, D) = \{(s,a) \mid s \in C \wedge a \in D(s)\}$ the state-action set corresponding to (C, D). ∎

[1] In fact, using Markov's inequality [Wil91] it is possible in some instances to obtain bounds for the probability of meeting deadlines. Since the bounds are not always sufficiently strict to be useful for verification, we have not pursued this approach in this dissertation.

Each sub-MDP (C, D) induces an *edge relation:* there is an edge (s, t) from $s \in C$ to $t \in S$ iff it is possible to go from s to t in one step with positive probability. The formal definition is as follows.

Definition 3.12 (edge relation ρ) For a sub-MDP (C, D), we define the relation $\rho_{(C,D)}$ by

$$\rho_{(C,D)} = \left\{ (s,t) \in C \times S \;\middle|\; \exists a \in D(s) \,.\, p_{st}(a) > 0 \right\} .$$

We also let

$$\rho_S = \left\{ (s,t) \in S \times S \;\middle|\; \exists a \in A(s) \,.\, p_{st}(a) > 0 \right\} . \qquad \blacksquare$$

The difference between state-action sets and sub-MDPs is simply one of notation. End components are a particular type of sub-MDPs.

Definition 3.13 (end components) A sub-MDP (C, D) is an *end component* if:

- $\mathrm{Succ}(s, a) \subseteq C$ for all $s \in C$ and $a \in D(s)$;
- the graph $(C, \rho_{(C,D)})$ is strongly connected.

We say that an end component (C, D) is *contained* in a sub-MDP (C', D') if $sa(C, D) \subseteq sa(C', D')$; we say that an end component (C, D) is *maximal* in a sub-MDP (C', D') if there is no other end component (C'', D'') such that $sa(C, D) \subset sa(C'', D'') \subseteq (C, D)$. We extend these definitions in the obvious way to containment and maximality in an MDP. $\blacksquare$

Intuitively, end components represent sets of state-action pairs that, once entered, can be followed forever if the policy chooses the actions in an appropriate way. This intuition will be made precise by our basic theorems on end components. From the above definition, we see that end components are a generalization of the *stable sets* defined by Bianco and de Alfaro [BdA95] and the *strongly connected stable sets* of de Alfaro [dA97]. These sets, in turn, are related to the ones described in the proof of Courcoubetis and Yannakakis [CY95, Proposition 4.2.3], as well as to the *controllably recurrent states* of Courcoubetis and Yannakakis [CY90]. The *K-ergodic sets* of Hart, Sharir and Pnueli [HSP82, HSP83] are also related to end components.

3.3.1 Basic Properties of End Components

The following theorems, derived in part from [dA97], summarize the basic properties of end components.

The first theorem states that, once an end component is entered, it is possible to stay in it forever, while taking each state-action pair in the end component infinitely often with probability 1. To make this statement precise, we formulate it in terms of policies: the theorem states that it is possible to modify each policy so that, once an end component is entered, it is never exited, and each state-action pair of the component is taken infinitely often with probability 1.

Theorem 3.1 (stability of end components) *Let (C, D) be an end component. Then, for every policy η there is a policy η', which differs from η only over C, such that*

$$\mathrm{Pr}_s^{\eta}(\mathit{reach}(C)) \;=\; \mathrm{Pr}_s^{\eta'}(\mathit{reach}(C)) \;=\; \mathrm{Pr}_s^{\eta'}(\mathit{inft}(\omega) = \mathit{sa}(C, D)) \tag{3.3}$$

for all $s \in S$.

Proof. Consider the policy η' defined as follows, for every sequence $s_0 \cdots s_n$ with $n \geq 0$:

- If $s_n \in C$, policy η' chooses an action from $D(s_n)$ with uniform probability, i.e.

$$Q_{\eta'}(a \mid s_0 \cdots s_n) = \begin{cases} |D(s_n)|^{-1} & \text{if } a \in D(s_n); \\ 0 & \text{otherwise.} \end{cases}$$

- If $s_n \notin C$, policy η' coincides with η, i.e.

$$Q_{\eta'}(a \mid s_0 \cdots s_n) = Q_\eta(a \mid s_0 \cdots s_n)\ .$$

The first equality in (3.3) is a consequence of the fact that η and η' coincide outside C.

For the second equality, notice that under policy η' a behavior that enters C never leaves C afterwards. Moreover, since the graph $(C, \rho_{(C,D)})$ is strongly connected, under policy η' the end component (C, D) behaves like an ergodic Markov chain: hence, once in C a behavior will visit all states of C infinitely often with probability 1. By definition of η', this also implies that once a behavior enters C, it takes all state-action pairs in $\mathit{sa}(C, D)$ infinitely often with probability 1. This completes the argument. ∎

The next result states that, for any initial state and policy, a behavior will end up with probability 1 in an end component. This fact is at the origin of the name "end component".

Theorem 3.2 (fundamental theorem of end components) *For any $s \in S$ and any policy η, $\mathrm{Pr}_s^{\eta}(\mathit{sub}(\mathit{inft}(\omega))$ is an end component$) = 1$.*

Proof. Consider a sub-MDP (C, D) which is not an end component, and let $\Omega_s^{(C,D)} = \{\omega \in \Omega_s \mid \mathit{inft}(\omega) = \mathit{sa}(C, D)\}$. There are two cases, depending on which condition of Definition 3.13 does not hold.

First, assume that there is $(t, a) \in \mathit{sa}(C, D)$ such that $\mathrm{Succ}(t, a) \not\subseteq C$. Let $r = \sum_{u \in C} p_{tu}(a)$. Then, since every behavior in $\Omega_s^{(C,D)}$ takes the state-action pair t, a infinitely often, we have that $\mathrm{Pr}_s^{\eta}(\omega \in \Omega_s^{(C,D)}) < r^k$ for all $k > 0$, and from $r < 1$ we obtain $\mathrm{Pr}_s^{\eta}(\omega \in \Omega_s^{(C,D)}) = 0$.

Otherwise, assume that there are $t_1, t_2 \in C$ such that there is no path from t_1 to t_2 in $(C, \rho_{(C,D)})$. Let

$$q = \max\left\{ \sum_{u \in C} p_{tu}(a) \;\middle|\; t \in C \wedge a \in D(t) \wedge \mathrm{Succ}(t, a) \not\subseteq C \right\}\ .$$

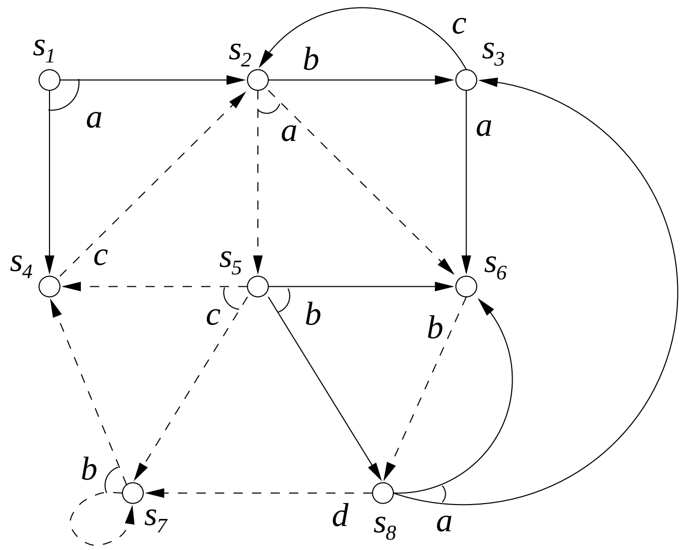

Figure 3.4: An MDP $\Pi = (S, A, p)$, with one of its end components depicted by dashed lines.

The lack of path from t_1 to t_2 in $(C, \rho_{(C,D)})$ implies that, for every subsequence $s_m a_m s_{m+1} \cdots s_n$ of behavior in $\Omega_s^{(C,D)}$ going from $s_m = t_1$ to $s_n = t_2$, there is $j \in [m..n-1]$ such that $\mathrm{Succ}(s_j, a_j) \notin C$. Thus, at most a fraction q of behaviors in $\Omega_s^{(C,D)}$ can go from t_1 to t_2 without leaving (C, D). Since every behavior $\omega \in \Omega_s^{(C,D)}$ contains infinitely many disjoint subsequences from t_1 to t_2, we have that $\Pr_s^\eta(\omega \in \Omega_s^{(C,D)}) \leq q^k$ for all $k > 0$. As $q < 1$, this implies again $\Pr_s^\eta(\omega \in \Omega_s^{(C,D)}) = 0$.

The result then follows from the fact that there are only finitely many sub-MDPs (C, D) in the original MDP. ∎

End components of Markov chains. As mentioned earlier, a Markov chain is an MDP in which to each state is associated a single action. From an exam of the definition of end component, we have the following characterization of end components of Markov chains.

Corollary 3.1 (end components of Markov chains) *Consider an MDP $\Pi = (S, A, p)$ corresponding to a Markov chain. A sub-MDP (C, D) is an end component iff:*

- *C is a closed recurrent class of the chain;*
- *$D(s) = A(s)$ for all $s \in C$.*

The corollary is useful to adapt the model-checking algorithms we give for general MDPs to Markov chains.

Example 3.5 (depicting end components) Figure 3.4 depicts an MDP $\Pi = (S, A, p)$ and a end component (B, C) of the MDP. The states and actions of the MDP are defined by $S = \{s_1, \ldots, s_8\}$, and $A(s_1) = \{a\}$, $A(s_2) = \{a, b\}$, $A(s_3) = \{a, c\}$, $A(s_4) = \{c\}$, $A(s_5) = \{b, c\}$, $A(s_6) = \{b\}$, $A(s_7) = \{b\}$, $A(s_8) = \{a\}$.

The end component (B, C) of the MDP is given by $B = \{s_2, s_4, s_5, s_6, s_7, s_8\}$, and $C(s_2) = \{a\}$, $C(s_4) = \{c\}$, $C(s_5) = \{c\}$, $C(s_6) = \{b\}$, $C(s_7) = \{b\}$, $C(s_8) = \{d\}$. ∎

3.3.2 Maximal End Components

Very often, we will be interested in the maximal end components contained in a given sub-MDP. We introduce the following abbreviations to denote this set.

Definition 3.14 ($maxEC$) Given a sub-MDP (B, C), we denote by $maxEC(B, C)$ the set of the maximal end components of (B, C).

Moreover, given a subset of states B, we denote with $A_{\setminus S}$ the restriction to domain B of the action assignment A, and we abbreviate $maxEC(B, A_{\setminus B})$ by $maxEC(B)$. In other words, $maxEC(B)$ is the set of maximal end components contained in the sub-MDP induced by the subset B of states. ∎

Given a sub-MDP (C, D), the set of maximal end components of (C, D) can be computed using the following algorithm, reminiscent of the procedure presented in [CY95, Proposition 4.2.3].

Algorithm 3.1 (computation of maximal end components)

Input: A sub-MDP (C, D).

Output: $\mathcal{L} = maxEC(C, D)$.

Initialization: $\mathcal{L} := \{(C, D)\}$.

Repeat the following steps:

- Select $(E, F) \in \mathcal{L}$.
- For all $s \in E$, let $F'(s) := \{a \in F(s) \mid \mathrm{Succ}(s, a) \subseteq E\}$.
- Let $E_1, \ldots, E_n$ be the strongly connected components of the graph $(E, \rho_{(E,F')})$, and let $F_i(s) = F'(s)$ for all $i \in [1..n]$ and $s \in E_i$.
- Replace $(E, F) \in \mathcal{L}$ with $(E_1, F_1), \ldots, (E_n, F_n)$.

Until: $\mathcal{L}$ cannot be changed by the above iteration. ∎

Given an MDP (S, A, p), it is easy to see that this algorithm runs in time polynomial in $\sum_{s \in S} |A(s)|$.

3.3.3 State-Action Frequencies and End Components (‡)

State-action frequencies represent the frequency with which state-action pairs appear in system behaviors. They are a classical concept in the study of Markov decision processes, and they will be used in the proofs of several results in the next chapters. Derman [Der70, Chapter 7] is an introduction to the topic which also summarizes earlier work, such as Derman [Der63, Der64], Derman and Strauch [DS66], and Strauch and Veinott [SV66]. Given a state-action pair s, a, let N^k_{sa} be the random variable denoting the number of times a behavior has taken state-action pair s, a before position k: formally,

$$N^k_{sa} = \sum_{i=0}^{k-1} \delta[X_i = s \wedge Y_i = a] \; , \tag{3.4}$$

where $\delta[\mathit{true}] = 1$ and $\delta[\mathit{false}] = 0$. We define the *state-action frequency* $x_{sa}(t, \eta)$ by

$$x_{sa}(t, \eta) = \lim_{n \to \infty} \frac{\mathrm{E}^\eta_t\{N^n_{sa}\}}{n} \; , \tag{3.5}$$

provided that the limit exists: $x_{sa}(t, \eta)$ represents the long-run frequency with which the pair s, a appears in a behavior, with initial state t and under policy η. When the initial state t and the policy η are clear from the context, we write simply x_{sa}.

Not surprisingly, there is a connection between state-action frequencies and end components, as both of these are related to the long-run behavior of the system. One aspect of the connection is described by the following theorem.

Theorem 3.3 (state-action frequencies and end components) *Fix any initial state and any policy η, and let*

$$\chi = \{(s, a) \mid x_{sa} > 0\}$$

be the set of state-action pairs having positive frequency, assuming that the limit (3.5) exists for all state-action pairs. Then, χ can be written as $\chi = \bigcup_{i=1}^n \chi_i$, where each sub$(\chi_i)$, for $1 \leq i \leq n$, is an end component.

Proof. Define the abbreviations

$$B_\chi = \{s \in S \mid \exists a \,.\, (s, a) \in \chi\}$$
$$E_\chi = \Big\{(s, t) \in S \times S \;\Big|\; \exists a \in A(s) \,.\, \big[(s, a) \in \chi \wedge t \in \mathrm{Succ}(s, a)\big]\Big\} \; .$$

The state-action frequencies are related by [Ber95]:

$$\sum_{s,a \in \chi_\Pi} p_{st}(a)\, x_{sa} = \sum_{a \in A(t)} x_{ta} \; , \qquad\qquad t \in S \; . \tag{3.6}$$

To show that χ can be written in the form $\chi = \bigcup_{i=1}^{n} \chi_i$, with $sub(\chi_1), \ldots, sub(\chi_n)$ end components, we have to show that the two following statements hold.

- *If* $(s, a) \in \chi$, *then* $\mathrm{Succ}(s, a) \subseteq B_\chi$. If $(s, a) \in \chi$ and $t \in \mathrm{Succ}(s, a)$, then by (3.6) it must be $x_{tb} > 0$ for at least one $b \in A(t)$, which proves the result.

- *The graph* (B_χ, E_χ) *is formed by the union of strongly connected subgraphs.* By defining the "flow" f_{st} as $f_{st} = \sum_{a \in A(s)} x_{sa} p_{st}(a)$, equations (3.6) can be rewritten as

$$\sum_{s \in S} f_{st} = \sum_{s' \in S} f_{ts'} \ , \qquad\qquad t \in S \ ,$$

which can be interpreted as the law of flow conservation. If the graph (B_χ, E_χ) could not be written as the union of strongly connected subgraphs, there would states $s, t \in B_\chi$ such that there is a path from s to t in (B_χ, E_χ), but no path from t to s, violating flow conservation.

An alternative proof of this result can be obtained by using the results of Derman [Der70, Chapter 7]. There, Derman states that the limit points of the state-action frequencies are contained in the convex hull determined by the state-action frequencies of deterministic policies. To obtain our result, it suffices to note that the sets of non-zero state-action frequencies of deterministic policies correspond to the union of end components, as can be easily proved. ∎

3.4 The Stochastic Shortest Path Problem

We now summarize some results about the *stochastic shortest path* (SSP) problem. This problem will be a component of several model-checking algorithms for the probabilistic logics discussed in this dissertation, so that it is convenient to introduce it at this early stage.

Informally, the stochastic shortest path problem consists in computing the minimum expected cost for reaching a given subset of destination states, from any non-destination state of an MDP. Bertsekas and Tsitsiklis [BT91] present a detailed account of this problem; this section is based on [BT91] and [Ber95]. In Chapter 7, we will improve on these results by presenting solution methods for instances of the SSP problem that cannot be solved with the methods of [BT91, Ber95].

Formally, an instance of the *stochastic shortest path* (SSP) problem is defined as follows.

Definition 3.15 (instance of stochastic shortest path problem) An instance of the *stochastic shortest path* problem consists of an MDP $\Pi = (S, A, p, U, c, g)$, in which the components U, c and g are defined as follows:

- U is the set of *destination states.*

- c is the *cost function,* which associates to each state $s \in S - U$ and action $a \in A(s)$ the cost $c(s, a)$.
- $g : U \mapsto \mathbb{R}$ is the *terminal cost function,* which associates to each $s \in U$ its terminal cost $g(s)$. ∎

Define $T_U = \min\{k \mid X_k \in U\}$, so that T_U is a random variable indicating the position of first entrance in U. Informally, the SSP problem consists in determining the minimum cost of reaching U when following a policy that reaches U with probability 1. We call these policies *SSP-proper.*

Definition 3.16 (SSP-proper policy) Given an instance (S, A, p, U, c, g) of SSP problem, a policy η is *SSP-proper* if $\Pr_s^\eta(T_U < \infty) = 1$ for all $s \in S - U$. We denote by $\boldsymbol{\eta}_P$ the class of SSP-proper policies. A policy that is not SSP-proper is called *SSP-improper.* ∎

The *cost* v_s^η of a policy η at $s \in S - U$ is defined by

$$v_s^\eta = \mathrm{E}_s^\eta \left\{ g(X_{T_U}) + \sum_{k=0}^{T_U - 1} c(X_k, Y_k) \right\} . \tag{3.7}$$

The SSP problem is formally defined as follows.

Definition 3.17 (SSP problem) Given an instance (S, A, p, U, c, g) of SSP problem, determine

$$v_s^* = \inf_{\eta \in \boldsymbol{\eta}_P} v_s^\eta$$

for all $s \in S - U$. ∎

Bertsekas and Tsitsiklis [BT91] provide solution methods for the SSP problem under the following assumptions:

SSP Assumption 1: There is at least one Markovian SSP-proper policy.

SSP Assumption 2: If η is Markovian and not SSP-proper, then $v_s^\eta = \infty$ for at least one $s \in S - U$.

As mentioned before, in Chapter 7 we will study the solution of SSP problems under different assumptions. The following theorem is taken from [BT91] (see also [Ber95]).

Theorem 3.4 (Bellman equations and SSP problem) *Let (S, A, p, U, c, g) be an instance of the SSP problem. Denote with $\boldsymbol{v} = [v_s]_{s \in S - U}$ a vector of real numbers, and define the functional L on the space of $\boldsymbol{v}$ by*

$$[L\boldsymbol{v}]_s = \min_{a \in A(s)} \left[c(s, a) + \sum_{t \in S - U} p_{st}(a)\, v_t + \sum_{t \in U} p_{st}(a)\, g(t) \right] \qquad s \in S - U . \tag{3.8}$$

If SSP Assumptions 1 and 2 are satisfied, the following assertions hold:

- *The functional L admits exactly one fixpoint $\boldsymbol{v}^\bullet$ such that $\boldsymbol{v}^\bullet = L\boldsymbol{v}^\bullet$.*
- *The fixpoint $\boldsymbol{v}^\bullet$ of L is the single optimal solution of the following linear programming problem on the set $\{v_s\}_{s \in S-U}$ of variables: Maximize $\sum_{s \in S-U} v_s$ subject to*

$$v_s \leq c(s,a) + \sum_{t \in S-U} p_{st}(a)\, v_t + \sum_{t \in U} p_{st}(a)\, g(t) \qquad s \in S-U\,. \tag{3.9}$$

- *It is $v_s^\bullet = v_s^*$ for all $s \in S-U$ or, in vector form, $\boldsymbol{v}^\bullet = \boldsymbol{v}^*$.*
- *Consider any Markovian policy η that selects at every $s \in S-U$ only actions that realize the minimum on the right hand side of (3.8): formally,*

$$Q_\eta(a \mid s) > 0 \qquad \textit{iff} \qquad a \in \arg\min_{a \in A(s)} \left[c(s,a) + \sum_{t \in S-U} p_{st}(a)\, v_t + \sum_{t \in U} p_{st}(a)\, g(t) \right],$$

for every $s \in S-U$. Then, policy η is SSP-proper, and it is $v_s^\eta = v_s^* = v_s^\bullet$ for all $s \in S-U$.

3.5 The Maximal Reachability Probability Problem

Another problem that will often recur in the model checking algorithms for the logics we present is the *maximal reachability probability problem.* This problem consists in determining the maximal probability with which a set of destination states can be reached from a given state. An algorithm for solving this problem has been provided by Courcoubetis and Yannakakis [CY90]. We return to the topic for the sake of completeness in our presentation of the verification algorithms, and to present the correctness proof for the algorithm, which is not included in [CY90]. Moreover, in the next subsection we will present results that can be used to improve the efficiency of the algorithms.

An instance of the *maximal reachability probability problem* is defined as follows.

Definition 3.18 (instance of maximal reachability probability problem) An instance of the maximal reachability probability problem consists of an MDP $\Pi = (S, A, p, U)$, where $U \subseteq S$ is the set of *destination states.* ∎

The problem can be defined as follows.

Definition 3.19 (maximal reachability probability problem) Given an instance $\Pi = (S, A, p, U)$, the maximal reachability probability problem consists in computing the quantity

$$v_s^* = \sup_\eta \Pr_s^\eta(\mathit{reach}(U))$$

for all $s \in S-U$. ∎

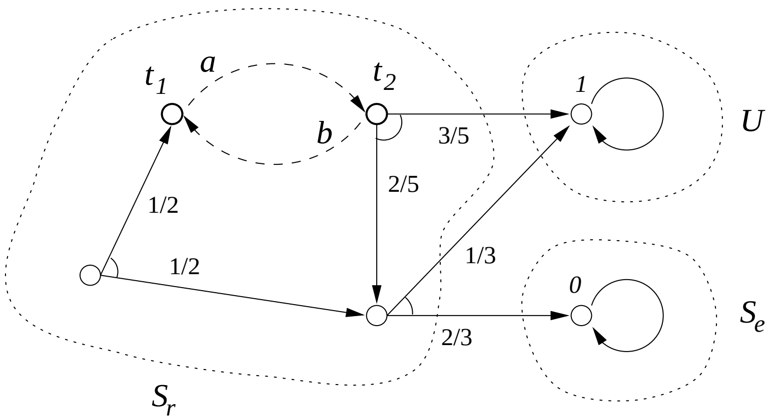

Figure 3.5: A TPS with sets U, S_r and S_e, along with the transition probabilities arising from the actions. The names of the actions, as well as the transition probabilities of deterministic actions, have not been indicated. The states of U and S_e are labeled with the terminal cost function of the corresponding SSP problem. The end component with state-action pairs (t_1, a) and (t_2, b) contained in S_r is indicated by dashed lines.

3.5.1 The Classical Algorithm

To solve the maximal reachability probability problem, we first compute the set of states from which the maximal reachability probability is non-zero. Let $S_e \subseteq S$ be the set of states that cannot reach U, and let $S_r = S - (S_e \cup U)$. Thus, S_r is the subset of states not in U that can reach U. The following lemma states that $S_r \cup U$ is exactly the set of states where the maximal reachability probability is non-zero.

Lemma 3.1 *The following assertions hold:*

- $\sup_\eta \Pr_s^\eta(\mathit{reach}(U)) > 0$ *iff* $s \in S_r \cup U$*;*
- *If* $s \in U$*, then* $\sup_\eta \Pr_s^\eta(\mathit{reach}(U)) = 1$.

Proof. Immediate. ∎

Thus, in the computation of the maximal reachability probability we can restrict our attention to the set S_r. Unfortunately, the problem of computing the maximal reachability probability on all states of S_r cannot be reduced to an instance of the SSP problem that satisfies SSP Assumptions 1 and 2. This is illustrated by the following example.

Example 3.6 Consider the TPS depicted in Figure 3.5. In the corresponding SSP problem, the destination set is $U \cup S_e$, and the terminal cost function is defined by $g(s) = 1$ if $s \in U$, and $g(s) = 0$

if $s \in S_e$. The cost function c is always 0. In this way, for all $s \in S_r$ and all policies η it is $v_s^\eta = \Pr_s^\eta(reach(U))$.

Consider a deterministic policy η_0 such that chooses action a at t_1 and b at t_2. Policy η_0 is not SSP-proper, since $\Pr_{t_1}^{\eta_0}(reach(U \cup S_e)) = 0$. On the other hand, the expected cost under η_0 is such that $0 \leq v_s^{\eta_0} \leq 1$ for all states s, since the cost c is always 0. This indicates that SSP Assumption 2 does not hold. In the next example, we will see that the fixpoint operator corresponding to the SSP problem has more than one fixpoint, which would not be possible if SSP Assumption 2 held. ∎

Consider a vector $\boldsymbol{v} = [v_s]_{s \in S_r}$, and define on the space of $\boldsymbol{v}$ the functional

$$[L\boldsymbol{v}]_s = \max_{a \in A(s)} \left[\sum_{t \in S_r} p_{st}(a)\, v_t + \sum_{t \in U} p_{st}(a) \right] \qquad s \in S_r\ . \tag{3.10}$$

For any policy η, define also $v_s^\eta = \Pr_s^\eta(reach(U))$, and let $\boldsymbol{v}^\eta = [v_s^\eta]_{s \in S_r}$. Unlike the case for the instances of the SSP problem considered in the preceding section, the operator L may admit more than one fixpoint. This is illustrated by the following example.

Example 3.7 Figure 3.6 depicts an MDP Π' together with two fixpoints of L; the lesser of these two fixpoints is the least fixpoint. Note that the least fixpoint correspond to the reachability probabilities: in the next theorems we show that this is no coincidence. We can trace the existence of more than one fixpoint to the presence of end components in S_r. ∎

Even though L admits more that one fixpoint, it can be proved that L admits a unique least fixpoint, and that this least fixpoint corresponds to the solution to the maximal reachability probability problem. The proof is given later in the chapter. As a result, we will obtain the following theorem.

Theorem 3.5 *Consider the following linear programming problem, over the set $\{v_s\}_{s \in S_r}$ of variables: Minimize $\sum_{s \in S_r} v_s$ subject to*

$$v_s \geq \sum_{t \in S_r} p_{st}(a)\, v_t + \sum_{t \in U} p_{st}(a) \qquad s \in S_r\ . \tag{3.11}$$

This problem admits exactly one optimal solution vector $\boldsymbol{v}^\bullet$. For every $s \in S_r$, it is $\sup_\eta \Pr_s^\eta(reach(U)) = v_s^\bullet$. Moreover, there is a Markovian (in fact, deterministic) policy η^* such that

$$\Pr_s^{\eta^*}(reach(U)) = \sup_\eta \Pr_s^\eta(reach(U)) = \max_\eta \Pr_s^\eta(reach(U))\ .$$

Note that the forms of the linear programming problems (3.9) and (3.11) are very similar, even though the justifications for the reductions are different.

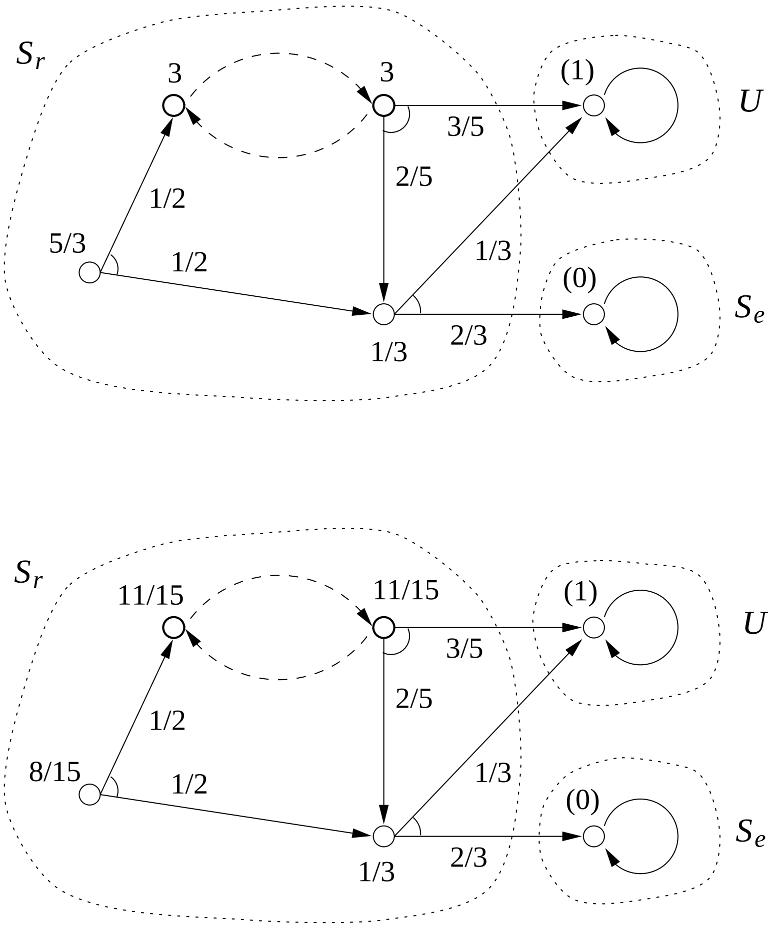

Figure 3.6: Two fixpoints of operator L on an MDP Π'. The values 0 and 1 in U and S_e are indicated in parentheses, since they do not belong to the vector $\boldsymbol{v}$. The end component in S_r is drawn with dashed lines. The fixpoint depicted below is the least fixpoint of L.

3.5.2 Improved Algorithms for Probabilistic Reachability (‡)

In the preceding subsection, we have presented an algorithm that computes the maximal probability of reaching a given subset of *destination states* U in the MDP. The method we have described is in fact rather classical (see, for example, [CY90]). While the reduction to linear programming provides us with the desired complexity bound for the algorithm, other computational approaches are also possible. As the proof of Theorem 3.10 hints, methods based on value and policy iteration can also be used. Such methods are described in several books on dynamic programming and optimal control, among which Derman [Der70], Puterman [Put94], and Bertsekas [Ber95].

Since the computation of maximal reachability probabilities is a common problem in probabilistic verification, we present an improved algorithm that relies on two preprocessing step to reduce the size of the linear programming problem to be solved. Since the preprocessing steps use only graph-theoretical concepts, the reduction in the number of states can lead to overall time savings.

First Preprocessing Step

The first preprocessing step consists in adding to the set U all states $U_1 \subseteq S_r$ that can reach U with maximal probability 1. This set U_1 can be determined with the following algorithm, that will be useful also in the future.

Algorithm 3.2 (set of states that can reach target with max probability 1)

Input: An MDP (S, A, p), and a subset $U \subseteq S$ of states.

Output: The subset $U_1 \subseteq S - U$ of states that can reach U with maximal probability 1.

Method:

- Let $S' = S$, $A' = A$.
- Repeat:
  - Let B' be the set of states that cannot reach S_ϕ in the graph $(S', \rho_{(S',A')})$.
  - Remove from S' all states in B'.
  - Repeat:
    - Remove from A' all the actions leading to at least one removed state.
    - Remove from S' all states $s \notin S_\phi$ that have no actions left in A'.
  - Until no more states or actions can be removed from S', A'.
- Until no more states or actions can be removed from S', A'.
- Output $U_1 := S'$. ∎

The correctness of the algorithm is stated and proved as follows.

Theorem 3.6 *If (S, A, p) and U are the inputs and U_1 the output of the above algorithm, then*

$$\max_{\eta} \Pr_s^{\eta}(\mathit{reach}(U)) = 1 \quad \text{iff} \quad s \in U \cup U_1$$

for all $s \in S$.

Proof. In one direction, we can prove by induction on the iteration on which a state s is removed from U_1 that $\max_\eta \Pr_s^{\eta}(\mathit{reach}(U)) < 1$.

In the other direction, let $\bar{A}$ be the value of A' when the algorithm terminates. Notice that $\bar{A}(s) \neq \emptyset$ for $s \in U_1$, or else s would have been eliminated during the iteration of the algorithm. Consider the policy η_1 that chooses at each state $s \in U_1$ an action from $\bar{A}(s)$ with uniform probability.

Notice that every state of U_1 can reach U in $(U_1 \cup U, \rho_{(U_1 \cup U, \bar{A})})$: otherwise, the state would have been removed from U_1 by the algorithm. Thus, in the Markov chain arising from η_1 there is no closed recurrent class that is contained in U_1. Under η_1, a behavior will therefore leave U_1 with probability 1. Note also that the only way a behavior can leave U_1 under policy η_1 is by passing through U. Hence, we conclude that under η_1 a behavior from any state $s \in U_1$ will reach U with probability 1. This yields the result. ∎

Once the correctness of the algorithm has been established, the first preprocessing step can be stated and proved correct as follows.

Theorem 3.7 (first preprocessing step) *Let (S, A, p, U) be an instance of maximal reachability probability problem, and let U_1 be the set computed by Algorithm 3.2 on input (S, A, p) and U. Then, for all $s \in S - U$ it is*

$$\max_{\eta} \Pr_s^{\eta}(\mathit{reach}(U)) = \max_{\eta} \Pr_s^{\eta}(\mathit{reach}(U \cup U_1)) \; . \tag{3.12}$$

Thus, we can substitute the instance (S, A, p, U) with an equivalent instance $(S, A, p, U \cup U_1)$ in which the maximal reachability probability must be computed for a smaller set of states.

Proof. Let B be the set of states that cannot reach U. Notice that these states also cannot reach U_1, so that they trivially satisfy (3.12). Also, by the correctness of Algorithm 3.2 it is $\Pr_s^{\eta}(\mathit{reach}(U)) = 1$ for all $s \in U_1$, so that the states in U_1 also satisfy (3.12).

For a state $s \in S - (U \cup U_1 \cup B)$, the inequality

$$\max_{\eta} \Pr_s^{\eta}(\mathit{reach}(U)) \leq \max_{\eta} \Pr_s^{\eta}(\mathit{reach}(U \cup U_1))$$

trivially holds. To show that in fact equality holds, let η_0 be a Markovian policy that realizes the maximum for the left hand side of (3.12). From η_0, we can define a modified policy η_m, which outside of U_1 behaves like η_0, and inside of U_1 coincides with the policy η_1 defined in the proof of the previous theorem. It is easy to check that under η_m it is

$$\max_\eta \Pr_s^\eta(\mathit{reach}(U \cup U_1)) = \Pr_s^{\eta_m}(\mathit{reach}(U)) = \Pr_s^{\eta_0}(\mathit{reach}(U \cup U_1)) ,$$

which concludes the proof. ∎

Second Preprocessing Step

The second preprocessing step consists in "short-circuiting" the end components in S_r, replacing them by single states, while preserving the reachability probabilities. This is done by applying the following algorithm.

Algorithm 3.3 (end component elimination)

Input: Instance $\Pi = (S, A, p, U)$, together with subsets S_r and S_e.

Output: Instance $\widehat{\Pi} = (\widehat{S}, \widehat{A}, \widehat{p}, \widehat{U})$, with the subsets $\widehat{S}_e$, $\widehat{S}_r$ of states.

Method: Let $\{(B_1, D_1), \ldots, (B_n, D_n)\} = \mathit{maxEC}(S_r)$. Define $\widehat{U} = U$, $\widehat{S}_e = S_e$, and

$$\widehat{S} = S \cup \{\widehat{s}_1, \ldots, \widehat{s}_n\} - \bigcup_{i=1}^n B_i \qquad \widehat{S}_r = S_r \cup \{\widehat{s}_1, \ldots, \widehat{s}_n\} - \bigcup_{i=1}^n B_i .$$

The action sets are defined by:

$$\begin{aligned} s \in S - \bigcup_{i=1}^n B_i : &\qquad \widehat{A}(s) = \{\langle s, a\rangle \mid a \in A(s)\} \\ 1 \le i \le n : &\qquad \widehat{A}(\widehat{s}_i) = \Big\{\langle s, a\rangle \;\Big|\; s \in B_i \wedge a \in A(s) - D(s)\Big\} . \end{aligned}$$

For $s \in \widehat{S}$, $t \in S - \bigcup_{i=1}^n B_i$ and $\langle u, a\rangle \in \widehat{A}(s)$, the transition probabilities are defined by

$$\widehat{p}_{st}(\langle u, a\rangle) = p_{ut}(a) \qquad \widehat{p}_{s\,\widehat{s}_i}(\langle u, a\rangle) = \sum_{t \in B_i} p_{ut}(a) . \quad ∎$$

The following theorem enables to use $\widehat{\Pi}$ for the computation of the maximal reachability probabilities in Π.

Theorem 3.8 *Let $\widehat{\Pi}$ be the MDP obtained from Π as in Algorithm 3.3. Denoting with $\Pr$, $\widehat{\Pr}$ the probabilities computed in Π, $\widehat{\Pi}$, it is*

$$s \in S_r - \bigcup_{i=1}^n B_i : \qquad \max_\eta \Pr_s^\eta(\mathit{reach}(U)) = \max_\eta \widehat{\Pr}{}_s^\eta(\mathit{reach}(\widehat{U})) \tag{3.13}$$

$$s \in B_i,\, 1 \le i \le n : \qquad \max_\eta \Pr_s^\eta(\mathit{reach}(U)) = \max_\eta \widehat{\Pr}{}_{\hat{s}_i}^\eta(\mathit{reach}(\widehat{U})) \,. \tag{3.14}$$

The proof of this theorem is presented later in the section. The following example illustrates the application of Algorithm 3.3.

Example 3.8 Figure 3.7 depicts an MDP Π, and the MDP $\widehat{\Pi}$ obtained by applying Algorithm 3.3 to Π. The actions corresponding to the end components of S_r are drawn using dashed lines. These end components are

$$\mathit{sub}\{(s_3, d), (s_4, i), (s_7, j)\} \qquad\qquad \mathit{sub}\{(s_5, f), (s_6, g)\}$$

and they are replaced by the new states $\hat{s}_1$ and $\hat{s}_2$. This example illustrates the potential reduction of the state-space of the system effected by the algorithm. ∎

3.5.3 Proof of the Results (‡)

We now prove the results on the maximal reachability probability problem. To simplify the arguments, we assume that once the set U of destination states is entered, it can never be exited. Formally, this is expressed by the requirement $\mathrm{Succ}(s, a) \subseteq U$ for all $s \in U$ and $a \in A(s)$. It is immediate to see that this assumption does not affect the generality of the arguments.

The Standard Algorithm

First, we present a preparatory lemma.

Lemma 3.2 *If $\boldsymbol{v}^\bullet = L\boldsymbol{v}^\bullet$, then $v_s^\bullet \ge 0$ for all $s \in S_r$.*

Proof. Assume that $v_- = \min\{v_t^\bullet \mid t \in S_r\} < 0$, and let $S_- = \{s \in S_r \mid v_s^\bullet = v_-\}$ be the subset of S_r consisting of the states at which $\boldsymbol{v}^\bullet$ reaches its minimum value v_-. Since each state of S_r can reach U, there must be $s \in S_-$ and $a \in A(s)$ such that $\mathrm{Succ}(s, a) \not\subseteq S_-$. This implies that

$$v_s^\bullet < \sum_{t \in S_r} p_{st}(a)\, v_t^\bullet + \sum_{t \in U} p_{st}(a) \,,$$

contradicting the fact that $\boldsymbol{v}^\bullet$ is a fixpoint of L. ∎

Our second result states that the fixpoints of L provide an upper bound for $\boldsymbol{v}^\eta$.

Theorem 3.9 *If $\boldsymbol{v}^\bullet = L\boldsymbol{v}^\bullet$, then $\boldsymbol{v}_s^\eta \le \boldsymbol{v}_s^\bullet$ for every η.*

Proof. From (3.10), it is

$$v_s^\bullet \ge \sum_{t \in S_r} p_{st}(a)\, v_t^\bullet + \sum_{t \in U} p_{st}(a)$$

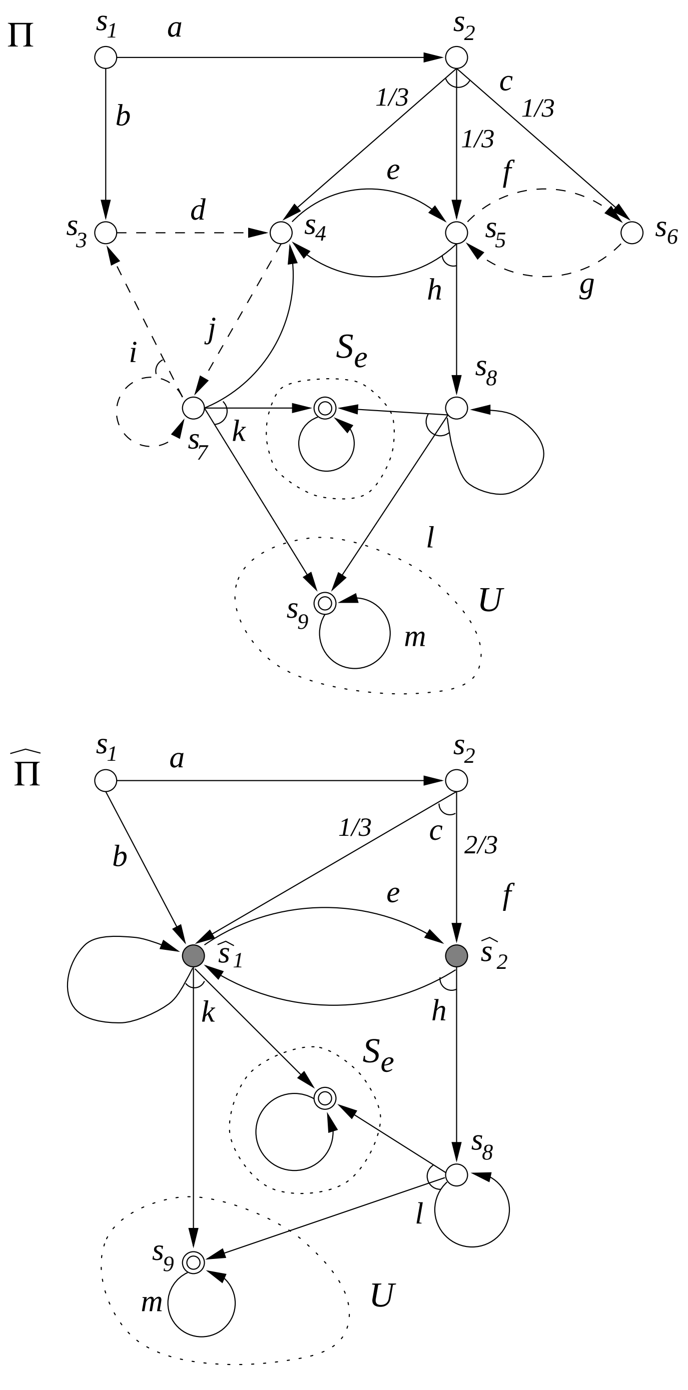

Figure 3.7: An example of application of Algorithm 3.3. To simplify the diagrams, we have indicated only the transition probability corresponding to action c. The new states introduced to replace the zero-cost end components are indicated by filled circles.

for all $s \in S_r$ and $a \in A(s)$. Iterating, for a general policy η and $n \geq 0$ we have

$$v_s^\bullet \geq \sum_{t \in S_r} \Pr_s^\eta(X_n = t)\, v_t^\bullet + Pr_s^\eta(X_n \in U)$$

for all $n \geq 0$. Taking the limit $n \to \infty$ and using Lemma 3.2, we obtain

$$\Pr_s^\eta(\mathit{reach}(U)) \;=\; \lim_{n\to\infty} \Pr_s^\eta(X_n \in U) \;\leq\; v_s^\bullet - \lim_{n\to\infty} \sum_{t \in S_r} \Pr_s^\eta(X_n = t)\, v_t^\bullet \;\leq\; v_s^\bullet \;,$$

which concludes the proof. ∎

Given a deterministic policy η we indicate by $\eta(s) \in A(s)$ the action chosen by η at $s \in S$. We also say that a Markovian policy η is *proper* if

$$\Pr_s^\eta(\mathit{reach}(S - S_r)) = 1$$

for every $s \in S_r$. To prove our main result, we need the following additional lemma.

Lemma 3.3 *Given a proper deterministic policy η, there is a function $\lambda : S_r \mapsto S_r \cup U$ with the following properties:*

1. $\lambda(s) \in \arg\max\left\{ v_t^\eta \;\middle|\; t \in \mathrm{Succ}(s, \eta(s)) \right\}$, *for all $s \in S_r$.*
2. $v_s^\eta \leq v_{\lambda(s)}^\eta$, *for all $s \in S_r$.*
3. *The graph $(S_r \cup U, E)$ with edges $E = \{(s, \lambda(s)) \mid s \in S_r\}$ is an in-forest (i.e. a forest in which edges are directed towards the roots) with roots in U.*

Proof. Extend $\boldsymbol{v}^\eta$ to U by $v_s^\eta = 1$ for $s \in U$. We construct λ as follows. Initially, λ is undefined on all $s \in S_r$; we then iterate the following steps:

- Choose $s \in S_r$ such that λ is not yet defined on s, and such that there is

  $$t \in \arg\max\left\{ v_t^\eta \;\middle|\; t \in \mathrm{Succ}(s, \eta(s)) \right\}$$

  such that either $t \in U$ or $\lambda(t)$ is already defined.
- Define $\lambda(s) = t$.

The iteration continues until the domain of λ cannot be further extended.

If the iteration defines λ on all S_r, it is immediate to see that properties 1, 2 and 3 hold. To see that the iteration defines λ on all S_r, assume towards the contradiction that after a certain number of iterations λ has been defined on a subset $S_d \subseteq S_r$ of states, and no $s \in S_r - S_d$ can be found for the next iteration. Let

$$S_m = \arg \max_{s \in S_r - S_d} v_s^\eta$$

be the subset of $S_r - S_d$ where $\boldsymbol{v}^\eta$ attains maximal value. Since η is proper, there must be $s_0 \in S_m$ such that

$$\mathrm{Succ}(s_0, \eta(s_0)) \not\subseteq S_m \; . \tag{3.15}$$

Since the iteration cannot extend the domain of λ, it is also

$$v_t^\eta < v_{s_0}^\eta \tag{3.16}$$

for all $t \in (U \cup S_d) \cap \mathrm{Succ}(s_0, \eta(s_0))$. Putting together (3.15) and (3.16), and considering the definition of S_m again, we get

$$v_{s_0}^\eta = \sum_{t \in S_r \cup U} p_{st}(\eta(s_0))\, v_t^\eta < v_{s_0}^\eta \; ,$$

which is the desired contradiction. ∎

Finally, we show that L has at least one fixpoint, and that the fixpoint corresponds to a reachability probability attained by a policy.

Theorem 3.10 *There is a deterministic policy η such that $\boldsymbol{v}^\eta = L\boldsymbol{v}^\eta$.*

Proof. Notice that there is at least one proper policy η_0, which can in fact be determined when the set of states S_r that can reach U is computed. Starting from η_0, we construct a sequence of proper deterministic policies $\eta_0, \eta_1, \ldots$. The construction proceeds as follows.

First, given a proper deterministic policy η_k, for $k \geq 0$, we construct a function $\lambda_k : S_r \mapsto S$ with the properties mentioned in the previous lemma. Next, we let

$$S_k = \left\{ s \;\middle|\; v_s^{\eta_k} < \max_{a \in A(s)} \left[\sum_{t \in S_r} p_{st}(\eta_k(s))\, v_t^{\eta_k} + \sum_{t \in U} p_{st}(\eta_k(s)) \right] \right\} . \tag{3.17}$$

If $S_k = \emptyset$, the sequence of policies terminates with η_k. Otherwise, we choose $s^k \in S_k$, and we define policy η_{k+1} to be the policy that coincides with η_k everywhere except at s^k, where

$$\eta_{k+1}(s^k) = \arg \max_{a \in A(s)} \left[\sum_{t \in S_r} p_{st}(\eta_k(s))\, v_t^{\eta_k} + \sum_{t \in U} p_{st}(\eta_k(s)) \right] .$$

If there is more than one action realizing the maximum, one such action is chosen arbitrarily. To see that η_{k+1} is also proper, let λ'_k be the function that coincides with λ_k on $S_r - \{s^k\}$, and defined on s^k by

$$\lambda'_k(s^k) = \arg\max \left\{ v_t^{\eta_k} \;\middle|\; t \in \mathrm{Succ}(s^k, \eta_{k+1}(s^k)) \right\} .$$

If there is more than one state realizing the maximum, one such state is chosen arbitrarily. Like λ_k, also λ'_k induces a in-forest of states with roots in U. In fact, no loops in λ'_k can be created, due to Property 2 of λ_k, to the strict inequality in (3.17), and to the definition of λ'_k in terms of λ_k. This indicates that η_{k+1} is also proper, since every state of S_r can reach U with positive probability under η_{k+1}.

Since η_{k+1} is proper, $\boldsymbol{v}^{\eta_{k+1}}$ is well-defined, and by the monotonicity of L and (3.17) we have $\boldsymbol{v}^{\eta_k} < \boldsymbol{v}^{\eta_{k+1}}$. As there is only a finite number of deterministic policies, a value of $k = K$ will be reached for which there is no state $s \in S_r$ that satisfies (3.17). If η_K is the final policy, the absence of s satisfying (3.17) shows that $\boldsymbol{v}^{\eta_K} = L\boldsymbol{v}^{\eta_K}$; the conclusion follows then from the fact that η_K is proper. ∎

Putting together this theorem and the previous one, we have the following corollary, which connects the fixpoints of L to the desired maximum reachability probability.

Corollary 3.2 *For every state $s \in S_r$, it is*

$$\sup_{\eta} \Pr_s^{\eta}(\mathit{reach}(U)) = v_s^* ,$$

where $\boldsymbol{v}^*$ is the least fixpoint of L. Moreover, there is a deterministic policy η such that $\boldsymbol{v}^\eta = \boldsymbol{v}^*$, so that the supremum is attained by at least one policy.

Proof. By Theorem 3.10, L admits a fixpoint $\boldsymbol{v}^\bullet = \boldsymbol{v}^{\eta^*}$ for some deterministic η^*. By Theorem 3.9, $\boldsymbol{v}^{\eta^*} \leq \boldsymbol{v}^\eta$ for all η. This yields the result. ∎

To complete the proof of Theorem 3.5, we need to prove that the linear programming problem (3.11) admits $\boldsymbol{v}^*$ as unique optimal solution, where $\boldsymbol{v}^*$ is the least fixpoint of L. The argument is completely standard (see for example Bertsekas [Ber95]); we sketch it here purely for the sake of completeness, since we will often use similar arguments in this dissertation.

Proof of Theorem 3.5. Let $\boldsymbol{v}^*$ be the least fixpoint of L. Clearly, $\boldsymbol{v}^*$ is a feasible solution of the linear programming problem.

In the other direction, consider a feasible solution $\boldsymbol{v}^\circ$ of the linear programming problem. From (3.11), reasoning as in the proof of Theorem 3.9 we can show that $\boldsymbol{v}^\eta \leq \boldsymbol{v}^\circ$ for all policies η. By Theorem 3.10, this implies $\boldsymbol{v}^* \leq \boldsymbol{v}^\circ$.

These two considerations show that $\boldsymbol{v}^*$ is the least feasible solution of the linear programming problem. ∎

The Improved Algorithm

Before proving Theorem 3.8, we need a preparatory lemma. To state the lemma, we introduce the following notation. Given a Markovian policy η, we denote by $\mathcal{S}(\eta)$ the following system of equations in $\{v_s\}_{s \in S_r}$:

$$v_s = \sum_{a \in A(s)} \sum_{t \in S_r} p_{st}(a)\, Q_\eta(a \mid s)\, v_t + \sum_{a \in A(s)} \sum_{t \in U} p_{st}(a)\, Q_\eta(a \mid s) \qquad s \in S_r \; .$$

Lemma 3.4 *If a Markovian policy η is proper, then the system of equations $\mathcal{S}(\eta)$ has $\boldsymbol{v}^\eta$ as unique solution.*

Proof. Let $P = [p_{st}]_{s,t \in S_r}$ be the matrix of the Markov chain associated to η on S_r, and define $\boldsymbol{q} = [q_s]_{s \in S_r}$ by

$$q_s = \sum_{a \in A(s)} \sum_{t \in U} p_{st}(a)\, Q_\eta(a \mid s)$$

for all $s \in S_r$. Then, $\mathcal{S}(\eta)$ can be rewritten as $\boldsymbol{v} = P\boldsymbol{v} + \boldsymbol{q}$. Since η is proper, P is the matrix of a transient Markov chain, and $\det(I - P) \neq 0$. Thus, $\mathcal{S}(\eta)$ admits a unique solution. The lemma then follows from the fact that the reachability probabilities $\boldsymbol{v}^\eta$ also satisfy $\mathcal{S}(\eta)$. ∎

Theorem 3.8 can be proved as follows.

Proof of Theorem 3.8. To prove the result, we break up the transformation effected by Algorithm 3.3 in two steps. Let $\{(B_1, D_1), \ldots, (B_n, D_n)\} = maxEC(S_r)$. First, from Π we construct an MDP $\widetilde{\Pi} = (S, \widetilde{A}, \widetilde{p})$ defined as follows:

$$\widetilde{A}(s) = \begin{cases} \{\langle s, a\rangle \mid a \in A(s)\} & \text{if } s \not\in \bigcup_{i=1}^n B_i; \\ \{\langle t, a\rangle \mid t \in B_i \wedge a \in A(t) - D_i(t)\} & \text{if } s \in B_i,\ 1 \leq i \leq n. \end{cases} \tag{3.18}$$

and $\widetilde{p}_{su}(\langle t, a\rangle) = p_{tu}(a)$ for all $s, u \in S$ and $\langle t, a\rangle \in \widetilde{A}(s)$. The sets U and S_r, like S, are not changed.

Let $\widetilde{L}$ be the operator defined for $\widetilde{\Pi}$ in the same way as L for Π. We have the following fact.

Fact 1. If $\boldsymbol{v}^\bullet = \widetilde{L}\boldsymbol{v}^\bullet$ is a fixpoint of $\widetilde{L}$, and $s, t \in B_i$, for $1 \leq i \leq n$, then $v^\bullet_s = v^\bullet_t$.

To see this, note that for $1 \leq i \leq n$ all states in B_i have the same set of actions, which lead to the same sets of destinations: from the definition of $\widetilde{L}$, we see that the value of $\boldsymbol{v}$ at all states of B_i must be the same. This leads to our next fact.

Fact 2. If $\boldsymbol{v} = \widetilde{L}\boldsymbol{v}$, then $\boldsymbol{v} = L\boldsymbol{v}$.

This can be seen as follows. Assume $\boldsymbol{v} = \widetilde{L}\boldsymbol{v}$. On $S - \bigcup_{i=1}^n B_i$, L and $\widetilde{L}$ are the same. Consider thus $s \in B_i$, $1 \leq i \leq n$. If $a \in A(s)$ then $\langle s, a\rangle \in \widetilde{A}(s)$, and from

$$v_s \geq \sum_{t \in S} \widetilde{p}_{st}(\langle s, a\rangle)\, v_t$$

we have

$$v_s \geq \sum_{t \in S} p_{st}(a)\, v_t \ . \tag{3.19}$$

Moreover, for each state $s \in B_i$ there is at least one action $a \in D_i(s)$. From $\mathrm{Succ}(s,a) \subseteq B_i$ follows

$$v_s = \sum_{t \in S} p_{st}(a)\, v_t \ , \tag{3.20}$$

showing that the maximum in the definition of L is realized at least for one state-action pair. From (3.19) and (3.20) follows then $\boldsymbol{v} = L\boldsymbol{v}$, concluding the proof of Fact 2.

Consider a reduction to the SSP problem in which we take the destination set to be $S - S_r$, the cost function c to be 0 for all state-action pairs, and the terminal cost $g(s)$ equal to -1 for $s \in U$ and 0 for $s \in S - (S_r \cup U)$. Since there are no end components in S_r, it is $\widetilde{\mathrm{Pr}}_s^{\eta}(\mathit{reach}(S - S_r)) = 1$ for all η, so that all policies are SSP-proper. Thus, from Theorem 3.4 we have the following fact.

Fact 3. There is exactly one $\boldsymbol{v}^\bullet$ such that $\boldsymbol{v}^\bullet = \widetilde{L}\boldsymbol{v}^\bullet$.

From the fixpoint $\boldsymbol{v}^\bullet$ we construct a deterministic policy $\widetilde{\eta}$ as follows. For each $s \in S_r$, define the set $A_{max}(s)$ by

$$A_{max}(s) = \arg\max_{a \in \widetilde{A}} \left[\sum_{t \in S_r} \widetilde{p}_{st}(a)\, v_t^\bullet + \sum_{t \in U} \widetilde{p}_{st}(a) \right] . \tag{3.21}$$

At each $s \in S_r$, policy $\widetilde{\eta}$ chooses deterministically an arbitrary action selected from $A_{max}(s)$. From Fact 1, for each $1 \leq i \leq n$ it is $A_{max}(s) = A_{max}(t)$ for all $s, t \in B_i$. Thus, we can force $\widetilde{\eta}$ to choose the *same* action $\langle u_i, a_i \rangle$ at all states of B_i, for $1 \leq i \leq n$.

Since there are no end components in S_r, policy $\widetilde{\eta}$ is proper. Since $\widetilde{\eta}$ chooses only actions that realize the maximum in (3.21), by Theorem 3.4 we have $\boldsymbol{v}^\bullet = \boldsymbol{v}^{\widetilde{\eta}}$.

From $\widetilde{\eta}$, we construct a Markovian policy η for Π such that $\boldsymbol{v}^\eta = \boldsymbol{v}^{\widetilde{\eta}}$. At $s \in S - \bigcup_{i=1}^n B_i$, policy η behaves like $\widetilde{\eta}$. At $s \in B_i$, with $1 \leq i \leq n$, there are two cases:

- if $s = u_i$, policy η chooses deterministically a_i;
- if $s \neq u_i$, policy η chooses with uniform probability an action from the set $D_i(s)$.

The idea behind this definition of η is as follows. Under η, when a behavior enters B_i it starts a random walk. If uninterrupted, this random walk would lead the behavior to visit each state of B_i with probability 1. The random walk terminates when the behavior visits u_i, at which point η chooses a_i, imitating the choice of $\langle u_i, a_i \rangle$ effected by $\widetilde{\eta}$. From this explanation, we see that η is proper.

Since η is proper, by the previous lemma the system of equations $\mathcal{S}(\eta)$ has $\boldsymbol{v}^\eta$ as unique solution. To show that $\boldsymbol{v}^\eta = \boldsymbol{v}^{\widetilde{\eta}}$, we show that $\boldsymbol{v}^{\widetilde{\eta}}$ is also a solution of $\mathcal{S}(\eta)$. The proof is by cases:

- If $s \in B_i$ for $1 \leq i \leq n$, and $s \neq t_i$ (where $\langle t_i, a_i \rangle$ is the action chosen by $\widetilde{\eta}$), then policy η chooses with uniform probability an action from $D_i(s)$. These actions lead only to destinations in B_i. From Fact 1, $\boldsymbol{v}^{\widetilde{\eta}} = \boldsymbol{v}^{\bullet}$ is constant on B_i, and the result follows.

- If $s \in B_i$ and $s = t_i$, for $1 \leq i \leq n$, or if $s \in S_r - \bigcup_{i=1}^{n} B_i$, the systems of equations $\mathcal{S}(\eta)$ and $\mathcal{S}(\widetilde{\eta})$ coincide on s, and the result follows.

Since $\boldsymbol{v}^{\widetilde{\eta}} = \boldsymbol{v}^{\bullet}$, it is also $\boldsymbol{v}^{\eta} = \boldsymbol{v}^{\bullet}$, and from Fact 2 it is $\boldsymbol{v}^{\eta} = \boldsymbol{v}^{\bullet} = L\boldsymbol{v}^{\bullet}$. By Theorem 3.9, policy η realizes the maximal probability of reaching U, indicating that this maximal probability is equal to $\boldsymbol{v}^{\bullet}$.

To complete the proof, we note that MDP $\widehat{\Pi}$ is obtained from $\widetilde{\Pi}$ by merging the states belonging to each B_i, for $1 \leq i \leq n$. The justification for this merging step is as follows.

By construction, if $s, t \in B_i$ for some $1 \leq i \leq n$, then $\widetilde{A}(s) = \widetilde{A}(t)$, and for any other state $u \in S$ and $a \in \widetilde{A}(s)$ we have $\widetilde{p}_{su}(a) = \widetilde{p}_{tu}(a)$. This indicates that s, t are in fact identical copies. The MDP $\widehat{\Pi}$ is obtained from $\widetilde{\Pi}$ by merging all the identical copies corresponding to B_i into a single representative state $\widehat{s}_i$, letting

$$\widehat{p}_{u,\hat{s}_i}(a) = \sum_{s \in B_i} \widetilde{p}_{us}(a)$$

for every $a \in \widehat{A}(s)$.

From these considerations, we see that the merging step does not influence the solution of the fixpoint equations. Specifically, if $\boldsymbol{v}^{\circ} = \widehat{L}\boldsymbol{v}^{\circ}$ is a fixpoint for $\widehat{\Pi}$, then we can show that

$$\begin{aligned} s \in S_r - \bigcup_{i=1}^{n} B_i : \qquad & v^{\circ}_s = v^{\bullet}_s \\ s \in B_i,\ 1 \leq i \leq n : \qquad & v^{\circ}_{\hat{s}_i} = v^{\bullet}_s \ , \end{aligned}$$

justifying relations (3.13) and (3.14). ∎

Chapter 4

Basic Probabilistic Logic

Several types of formalisms have been applied to the formal specification of probabilistic properties of systems, as we discussed in the introduction. In this dissertation, we follow the approach of extending branching-time temporal logics with probabilistic operators. The reason branching-time temporal logics are preferred to linear-time ones as the basis of probabilistic extensions is that probability measures on system behaviors are structurally similar to path quantifiers, as we will see.

Our probabilistic extensions are based on the branching-time temporal logics CTL and CTL*, introduced in Emerson and Sistla [CE81], Ben-Ari, Pnueli, and Manna [BAPM83] and Emerson and Halpern [EH85]. The extensions are obtained by augmenting CTL and CTL* with operators that enable the expression of probabilistic system properties. As mentioned in the introduction, we consider four new operators: P, D, $\bar{\mathrm{P}}$, and $\bar{\mathrm{D}}$, which enable the expression of four types of probabilistic properties, as summarized in Table 1.1.

In this chapter we consider the operators P and D, postponing to the next chapter the introduction of $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ and a detailed discussion of the classification of probabilistic temporal operators. The operator P is used to express bounds on the probability of system behaviors. Precisely, if ϕ is a formula encoding a property of behaviors, the formula $\mathrm{P}_{\geq a}\phi$ (resp. $\mathrm{P}_{\leq a}\phi$) holds at a state if the probability of satisfying ϕ from s is at least (resp. at most) a, under any policy. The operator D is used to express bounds on average times. If ψ is a state property, formula $\mathrm{D}_{\geq a}\psi$ (resp. $\mathrm{D}_{\leq a}\psi$) holds at a state s if a state that satisfies ψ can be reached from s in average time at least (resp. at most) a, regardless of the policy.

4.1 A Brief History of Probabilistic Temporal Logics

The practice of extending branching-time logics with additional operators for the expression of probabilistic properties is not new. The first probabilistic extension of branching-time logics for the expression of probabilistic system properties was proposed by Hansson and Jonsson [HJ89, HJ94].

This proposal considered systems modeled as Markov chains and introduced a temporal logic, PCTL, to specify the probability of satisfying temporal formulas within a given amount of chain transitions. The formulas of PCTL are obtained by adding subscripts and superscripts to CTL formulas; as an example, the formula $\diamond^{\leq 50}_{\geq 0.8}\phi$ means that with probability 0.8, formula ϕ will become true after a sequence of less than 50 state transitions of the Markov chain.

The model-checking algorithms of [HJ89, HJ94] rely on results on Markov chains and dynamic programming. Their time complexity is polynomial in the size of the system and linear in the numerical values of the time bounds appearing in the superscripts. This time complexity is typical of the algorithms that rely on the direct or indirect use of clocks, such as the dynamic programming recursions used to compute the probability of meeting timing deadlines.

The logic PCTL was later extended by Hansson [Han94] to systems that include nondeterminism, and which provide a more refined model of time. The model-checking algorithms proposed by [Han94] for this extension are based on the enumeration of all possible deterministic policies, leading to exponential time-complexity in the size of the system.

Aziz et al. [ASB+95] introduced the logic pCTL*, derived from CTL* by adding the operator P, which is defined as in the logics presented here. Two types of system models are considered: Markov chains and generalized Markov chains (called *Markov processes* and *generalized Markov processes* in [ASB+95]). The model checking of pCTL* specifications on Markov chains is based on the results of Courcoubetis and Yannakakis [CY88], and is of polynomial complexity in the size of the Markov chain. Generalized Markov chains are Markov chains in which some transition probabilities are given as parameters; these parameters must satisfy a specified set of constraints. The model checking of pCTL* specifications on generalized Markov chains is shown to be elementarly decidable using results from the theory of the real closed field, but no practical verification algorithm is provided.

The logic pCTL* was later extended to systems with nondeterminism by Bianco and de Alfaro [BdA95], who also provided model-checking algorithms for the resulting logics. Later, [dA97] considered a computational model related to TPSs, and introduced the operator D for the expression of average times. The model-checking algorithms presented improve on [BdA95], and are essentially the ones presented for pTL and pTL* in this chapter.

Recent developments include the study of probabilistic model checking in presence of fairness by Baier and Kwiatkowska [KB96]. This work, which was influential in the development of our approach to fairness, will be discussed in detail in Chapter 8.

An algorithm for the symbolic model-checking of pCTL and pCTL* formulas on Markov chains was presented by Baier, Kwiatkowska and Ryan [BKR+97]. The algorithm proposed is a symbolic implementation of the algorithms described in [BdA95, dA97] for the case of Markov chains. The algorithm uses *Multi-Terminal Binary Decision Diagrams* [CFZ96] to represent transition probability matrices and to compute the necessary reachability probabilities; this representation is found to be more compact and efficient than one based on traditional sparse matrices. It would be of great interest to study how this symbolic approach can be applied to systems with nondeterminism, and to the model-checking algorithms presented in this dissertation for the various logics presented.

The operators D, $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ have been introduced in the work that led to this dissertation, along with the classification of probabilistic properties given in Table 1.1.

4.2 The Logics pTL and pTL*

The definitions of syntax and semantics of the logics pTL and pTL* are taken from [dA97], except that the use of *instrumentation clocks* has been replaced by the notion of expected transition times in TPSs.

4.2.1 Syntax of pTL and pTL*

As in CTL and CTL*, the formulas of pTL and pTL* are divided in two mutually exclusive classes: the class *Stat* of *state formulas* (whose truth value is evaluated on the states), and the class *Seq* of *sequence formulas* (whose truth value is evaluated on infinite sequences of states). To define these classes, we assume that we have a fixed set $\mathcal{V}$ of typed variables, and fixed sets of interpreted predicate and function symbols. We let *Atomic* be the class of *atomic formulas* containing all first-order logic formulas formed from variables in $\mathcal{V}$ using these interpreted predicate and function symbols. The classes *Stat* and *Seq* are then defined by the following inductive clauses.

State formulas:

$$\phi \in \mathit{Atomic} \implies \phi \in \mathit{Stat} \tag{4.1}$$

$$\phi \in \mathit{Stat} \implies \neg\phi \in \mathit{Stat} \tag{4.2}$$

$$\phi,\ \psi \in \mathit{Stat} \implies \phi \wedge \psi \in \mathit{Stat} \tag{4.3}$$

$$\phi \in \mathit{Seq} \implies \mathrm{A}\phi,\ \mathrm{E}\phi \in \mathit{Stat} \tag{4.4}$$

$$\phi \in \mathit{Seq} \implies \mathrm{P}_{\bowtie a}\phi \in \mathit{Stat} \tag{4.5}$$

$$\phi \in \mathit{Stat} \implies \mathrm{D}_{\bowtie b}\phi \in \mathit{Stat} \tag{4.6}$$

Sequence formulas:

$$\phi \in \mathit{Stat} \implies \phi \in \mathit{Seq} \tag{4.7}$$

$$\phi \in \mathit{Seq} \implies \neg\phi \in \mathit{Seq} \tag{4.8}$$

$$\phi,\ \psi \in \mathit{Seq} \implies \phi \wedge \psi \in \mathit{Seq} \tag{4.9}$$

$$\phi \in \mathit{Seq} \implies \Box\phi,\ \Diamond\phi \in \mathit{Seq} \tag{4.10}$$

$$\phi,\ \psi \in \mathit{Seq} \implies \phi\,\mathcal{U}\,\psi \in \mathit{Seq}\,. \tag{4.11}$$

In the above definition, $\bowtie$ stands for one of $\{<, \leq, \geq, >\}$, and a and b are real numbers such that $0 \leq a \leq 1$ and $b \geq 0$. The temporal operators $\Box$, $\Diamond$, $\mathcal{U}$, and the path quantifiers A, E are taken from CTL*. As usual, the other propositional connectives are defined in terms of $\neg$, $\wedge$.

Just as CTL is a restricted version of CTL*, the logic pTL is a restricted version of pTL*; its definition is obtained by replacing clauses (4.7)–(4.11) with the clauses

$$\phi \in \mathit{Stat} \implies \Box\phi,\ \Diamond\phi \in \mathit{Seq} \tag{4.12}$$

$$\phi,\ \psi \in \mathit{Stat} \implies \phi\,\mathcal{U}\,\psi \in \mathit{Seq}\,. \tag{4.13}$$

4.2.2 Semantics

Consider a TPS $\Pi = (S, A, p, S_{in}, \mathit{time}, \mathcal{I})$. The semantics of pTL and pTL* formulas with respect to Π is defined by two satisfaction relations, one for state formulas and one for sequence formulas. For $\phi \in \mathit{Stat}$, we indicate with $s \models \phi$ the satisfaction of ϕ at $s \in S$. For $\psi \in \mathit{Seq}$ and $k \in \mathbb{N}$ we indicate with $\omega \models_k \psi$ the satisfaction of ψ at position k of behavior $\omega \in \bigcup_{s \in S} \Omega_s$. The base case for (4.1) and the cases for the logical connectives are immediate.

Temporal Operators

The satisfaction relation $\omega \models_k \phi$ for a behavior ω and $\phi \in \mathit{Seq}$ is defined in the usual way (see for example [MP93, MP91]):

$$\begin{array}{rrcl}
\phi \in \mathit{Stat} & \omega \models_k \phi & \mathit{iff} & s_k \models \phi \\
\phi \in \mathit{Seq} & \omega \models_k \Box\phi & \mathit{iff} & \forall i\,.\,[i \geq k \to \omega \models_i \phi] \\
\phi \in \mathit{Seq} & \omega \models_k \Diamond\phi & \mathit{iff} & \exists i\,.\,[i \geq k \wedge \omega \models_i \phi] \\
\phi, \psi \in \mathit{Seq} & \omega \models_k \phi\,\mathcal{U}\,\psi & \mathit{iff} & \exists i\,.\,\Big[i \geq k \wedge \omega \models_i \psi \wedge \forall j\,.\,(k \leq j < i \to \omega \models_j \phi)\Big]\,.
\end{array}$$

Path Quantifiers

The semantics of the path quantifiers is defined as in CTL and CTL* [Eme90]. For $s \in S$ and $\phi \in \mathit{Seq}$, it is

$$\begin{array}{lcl}
s \models \mathrm{A}\phi & \mathit{iff} & \forall\omega \in \Omega_s\,.\,\omega \models_0 \phi \\
s \models \mathrm{E}\phi & \mathit{iff} & \exists\omega \in \Omega_s\,.\,\omega \models_0 \phi\,.
\end{array}$$

Probabilistic Operator P

The semantics of the probabilistic quantifiers is defined as in [BdA95]: for sequence formula $\phi \in \mathit{Seq}$, state $s \in S$, probability bound $0 \leq a \leq 1$ and $\bowtie \in \{<, \leq, \geq, >\}$,

$$s \models \mathrm{P}_{\bowtie a}\phi \quad \textit{iff} \quad \forall \eta \,.\, \mathrm{Pr}^{\eta}_{s}(\omega \models_0 \phi) \bowtie a \,. \tag{4.14}$$

The intuitive meaning of (4.14) is that $\mathrm{P}_{\bowtie a}\phi$ holds at $s \in S$ if a behavior from s satisfies ϕ with probability $\bowtie a$, regardless of the policy.

Average-Time Operator D

Given a behavior ω and a state formula $\phi \in \mathit{Stat}$, let $T_\phi = \min\{k \mid \omega \models_k \phi\}$ be the first position of ω at which ϕ holds, with $T_\phi = \infty$ if $\forall k \,.\, \omega \not\models_k \phi$. For state $s \in S$, formula $\phi \in \mathit{Stat}$, bound $b \geq 0$ and $\bowtie \in \{<, \leq, \geq, >\}$ we define

$$s \models \mathrm{D}_{\bowtie b}\phi \quad \textit{iff} \quad \forall \eta \,.\, \mathrm{E}^{\eta}_{s}\left\{ \sum_{k=0}^{T_\phi - 1} \mathit{time}(X_k, Y_k) \right\} \bowtie b \,. \tag{4.15}$$

The intuitive meaning of (4.15) is that $\mathrm{D}_{\bowtie d}\phi$ holds at $s \in S$ if the TPS reaches a ϕ-state in average time $\bowtie d$, regardless of the policy.

To see that the semantics of pTL and pTL* is well defined, it is possible to show by induction on the structure of formulas that the events and functions mentioned in the above definitions are measurable. In particular, if $\phi \in \mathit{Seq}$ and $s \in S$, the truth-set $\{\omega \in \Omega_s \mid \omega \models_0 \phi\}$ is measurable; if $\phi \in \mathit{Stat}$, then T_ϕ is a random time, and $\sum_{k=0}^{T_\phi - 1} \mathit{time}(X_k, Y_k)$ is a measurable function.

We say that a TPS satisfies a specification if the specification holds on all initial states of the TPS.

Definition 4.1 (validity over TPS) Given a TPS $\Pi = (S, A, p, S_{in}, \mathit{time}, \mathcal{I})$ and a pTL or pTL* specification $\phi \in \mathit{Stat}$, we say that Π *satisfies* ϕ, written $\Pi \models \phi$, iff $s \models \phi$ for all $s \in S_{in}$. ∎

4.2.3 Examples of Specification

We present here only two simple examples of specifications that can be written in the logics pTL and pTL*. In spite of their simplicity, these examples illustrate some typical uses of the operators P and D.

Example 4.1 (expected time to failure) Assume that Π is a TPS that models an industrial plant. Assume that we have the following atomic state formulas:

- a formula *good* which characterizes system states lying inside a prescribed safe control region;

- a formula *broken* which characterizes system states at which some failure has occurred.

The states satisfying $\neg good \wedge \neg broken$ are states that do not lie in the safe control region, but at which no failure has yet occurred. The pTL formula

$$\mathrm{A}\Box\Big(good \rightarrow \mathrm{D}_{\geq a}\, broken\Big)$$

specifies that, if the system state is in the safe control region, a failure will occur with expected time no less than a. ∎

Example 4.2 (probability of deadlock) Assume that Π is a TPS that models a distributed system. Assume that we have the following atomic state formulas:

- formulas *task-start, task-in-prog, task-complete* which characterize the states at which a task is begun, is in progress, and has been completed, respectively;

- a formula *deadlock* which characterizes states at which a deadlock has occurred.

We assume that the above four formulas are mutually exclusive. The pTL* formula

$$\mathrm{A}\Box\Big[\textit{task-start} \rightarrow \mathrm{P}_{\geq a}\Big((\textit{task-start} \vee \textit{task-in-prog})\, \mathcal{U}\, \textit{task-comp}\Big)\Big]$$

specifies that, whenever a new task is started, the probability it completes without deadlocks is at least a. ∎

4.3 Checking Non-Zenoness

The first task in the verification of a TPS is to check whether it is non-Zeno. The following theorem provides a criterion for non-Zenoness that leads to an efficient checking algorithm.

Theorem 4.1 (criterion for non-Zenoness) *Let $\Pi = (S, A, p, S_{in}, time, \mathcal{I})$ be a TPS. For each $s \in S$, define $B(s) = \{a \in A(s) \mid time(s,a) = 0\}$ to be the set of actions with zero expected time from s. Then, Π is Zeno iff there is an end component (C, D) of (S, B) which is reachable in the graph (S, ρ_S) from some initial state in S_{in}.*

Proof. In one direction, assume that there is an end component (C, D) of (S, B) reachable from a state $s_0 \in S_{in}$ in the graph (S, ρ_S). Since C is reachable from s_0, there is a policy η such that $\Pr^{\eta}_{s_0}(\exists k \,.\, X_k \in C) > 0$. By Theorem 3.1, there is a policy η' such that $\Pr^{\eta}_{s_0}(\mathrm{I}(\omega) = (C, D)) > 0$. Since (C, D) is contained in (S, B), this leads to

$$\Pr^{\eta'}_{s_0}\Big(\sum_{i=0}^{\infty} time(X_i, Y_i) < \infty\Big) > 0\ ,$$

showing that Π is Zeno.

Conversely, assume that (S, B) does not contain any end component. By Theorem 3.2, the set $\mathit{inft}(\omega)$ is an end component with probability 1. Since for each end component (C, D) reachable from an initial state there is at least one state-action pair $(s, a) \in \mathit{sa}(C, D)$ with $\mathit{time}(s, a) > 0$, with probability 1 every behavior takes infinitely many steps with $\mathit{time} > 0$. This leads immediately to

$$\Pr_s^\eta\Big(\sum_{i=0}^{\infty} \mathit{time}(X_i, Y_i) = \infty\Big) = 1$$

for all η and all $s \in S_{in}$, which is the definition of non-Zenoness. ∎

This theorem leads immediately to the following algorithm.

Algorithm 4.1 (checking non-Zenoness)

Input: TPS $\Pi = (S, A, p, S_{in}, \mathit{time}, \mathcal{I})$.

Output: Yes/no answer, stating whether Π is non-Zeno.

Method: For each $s \in S$ let $B(s) = \{a \in A(s) \mid \mathit{time}(s, a) = 0\}$, and use Algorithm 3.1 to compute the set $\mathit{maxEC}(S, B) = \{(C_1, D_1), \ldots, (C_n, D_n)\}$ of maximal end components of (S, B). Check whether $\bigcup_{i=1}^{n} C_i$ is reachable from S_{in}, and answer "no" if it is, and "yes" if it is not. ∎

4.4 Model Checking for Operator P in pTL*

In this and in the next section we present algorithms to decide whether a TPS Π satisfies a specification ϕ written in pTL or pTL*. The model checking algorithms share the same basic structure of those proposed in Emerson and Lei [EL85] for CTL and CTL*. Given a TPS Π and a formula $\phi \in \mathit{Stat}$, the algorithms recursively evaluate the truth values of the state subformulas of ϕ at all states $s \in S$, following the recursive definitions (4.1)–(4.6), until the truth value of ϕ itself can be computed at all $s \in S$. Since the logics pTL and pTL* are obtained by adding the operators P and D to CTL and CTL*, we need to examine only the cases corresponding to these operators. In this section, we present algorithms for the model checking of the P operator; in the next section we will present the algorithms for D.

The model checking of operator P in the logic pTL* can be done using the results of Courcoubetis and Yannakakis [CY90]; we will present below an alternative algorithm. The model checking of operator P in the logic pTL can be done using the algorithms for pTL*, since pTL is a subset of pTL*. However, for the case of pTL it is also possible to use a simpler algorithm, due to Bianco and de Alfaro [BdA95]; we will present this algorithm in Section 4.8. The algorithms for the model checking of operator D are the same in pTL and pTL*.

From definition (4.14), to decide whether $s \models \mathrm{P}_{\bowtie a}\phi$ for a state s and a formula $\phi \in \mathit{Seq}$ it suffices to compute the minimum and maximum probabilities with which a computation from s satisfies ϕ. These probabilities can be computed using the algorithm presented by Courcoubetis and Yannakakis [CY90], which realizes the optimal complexity bound given in [CY88, CY95]. We also recall that if the TPS corresponds to a Markov chain, the model checking can be done with the algorithms of [CY88], which yield a better complexity bound.

Here we present an alternative algorithm, first described in [dA97]. This algorithm relies on the complete determinization procedure for ω-automata presented by Safra [Saf88, Saf92], instead of the partial determinization used in [CY88, CY90]. The algorithm achieves the optimal complexity of the one of [CY90]. By relying on a complete determinization, the algorithm has a fairly simple structure, and it can be easily generalized to handle different types of probabilistic logics. The algorithm has been recently extended by Baier and Kwiatkowska [KB96] to logics with fairness assumptions on the policies; we will also extend the algorithm to systems that include fairness in Chapter 8.

The algorithm. Let $\phi \in \mathit{Seq}$ be a sequence formula of pTL*. For any state $s \in S$ and policy η, it is

$$\mathrm{Pr}_s^\eta(\phi) = 1 - \mathrm{Pr}_s^\eta(\neg\phi) \ . \tag{4.16}$$

Thus, the problem of minimizing $\mathrm{Pr}_s^\eta(\phi)$ can be reduced to the problem of maximizing $\mathrm{Pr}_s^\eta(\neg\phi)$, and it will suffice to consider the maximization problem.

Let $\alpha_1, \ldots, \alpha_n \in \mathit{Stat}$ be the maximal state subformulas of ϕ, i.e. the state subformulas of ϕ that are not proper subformulas of any other state subformula of ϕ. Define $\phi' = \phi[r_1/\alpha_1]\ldots[r_n/\alpha_n]$ to be the result of replacing each α_i with a new propositional symbol r_i in ϕ. The formula ϕ' is therefore a linear-time temporal formula constructed from $r_1, \ldots, r_n$ using the temporal operators $\Box$, $\Diamond$, $\mathcal{U}$. As usual, we assume that truth values of $\alpha_1, \ldots, \alpha_n$ have already been computed at all states of the TPS, and we define the *label* $l(t)$ of $t \in S$ by $l(t) = \{r_i \mid 1 \le i \le n \wedge t \models \alpha_i\}$. We will refer to the TPS Π with the additional label l as the *l-labeled TPS*.

By the results of Vardi and Wolper [VW86] and Safra [Saf88, Saf92], formula ϕ' can be translated into a deterministic Rabin automaton $DR_{\phi'} = (Q, q_{in}, \Sigma, \gamma, U)$ with state space Q, initial state $q_{in} \in Q$, alphabet $\Sigma = 2^{\{r_1,\ldots,r_n\}}$, transition relation $\gamma : Q \times \Sigma \mapsto Q$, and acceptance condition U. The acceptance condition is a list $U = \{(P_1, R_1), \ldots, (P_m, R_m)\}$ of pairs of subsets of Q. An infinite sequence $\sigma : b_0 b_1 b_2 \cdots$ of symbols of Σ is accepted by $DR_{\phi'}$ if it induces a sequence $\omega_\sigma : q_0 q_1 q_2 \cdots$ of states of Q such that:

- $q_0 = q_{in}$;
- $\gamma(q_i, b_i) = q_{i+1}$ for all $i \ge 0$;
- for some $1 \le j \le m$, it is $\mathit{inftst}(\omega_\sigma) \subseteq P_j$ and $\mathit{inftst}(\omega_\sigma) \cap R_j \ne \emptyset$, where $\mathit{inftst}(\omega)$ denotes the set of states occurring infinitely often along ω.

Given the l-labeled TPS Π and $DR_{\phi'}$, we construct the *product MDP* $\Pi_{\phi'} = \Pi \otimes DR_{\phi'}$ as follows.

Algorithm 4.2 (product of l-labeled TPS and Rabin automaton)

Input: l-labeled TPS $\Pi = (S, A, p, S_{in}, \mathit{time}, \mathcal{I})$ and Rabin automaton $DR_{\phi'} = (Q, q_{in}, \Sigma, \gamma, U)$.

Output: Product MDP $\Pi_{\phi'} = \Pi \otimes DR_{\phi'} = (S', A, p', U')$.

Method: The product MDP $\Pi' = (S', A', p', U')$ is defined as follows:

- $S' = S \times Q$.
- For $(t, q) \in S'$, $A'(t, q) = A(t)$.
- For each $(t, q) \in S'$ and $a \in A(t)$, the probability $p'_{(t,q)(t',q')}(a)$ of a transition to $(t', q') \in S'$ is equal to $p_{t,t'}(a)$ if $\gamma(q, l(t')) = q'$, and is equal to 0 otherwise.
- $U' = \{(P'_1, R'_1), \ldots, (P'_m, R'_m)\}$, where $P'_i = S \times P_i$ and $R'_i = S \times R_i$ for $1 \le i \le m$. ∎

Given the product MDP $\Pi' = (S', A', p', U')$, we can compute $\sup_\eta \Pr^\eta_s(\phi) = \max_\eta \Pr^\eta_s(\phi)$ at every state of the original MDP Π by using the following result.

Theorem 4.2 (maximal probability of TL formula) *Assume* $U' = \{(P'_1, R'_1), \ldots, (P'_m, R'_m)\}$. *For each* $1 \le i \le m$, *let*

$$\mathcal{L}_i = \Big\{(C, D) \;\Big|\; (C, D) \in \mathit{maxEC}(P'_i) \wedge C \cap R'_i \neq \emptyset\Big\}$$

be the set of maximal end components of P'_i *that have non-empty intersection with* R'_i. *Let then* $S_\phi = \bigcup_{i=1}^m \bigcup_{(C,D)\in\mathcal{L}_i} C$ *be the union of all the states belonging to these end components. Note that the set* S_ϕ *can also be defined more concisely as*

$$S_\phi = \bigcup \Big\{B \;\Big|\; \exists (P, R) \in U' \,.\, \exists (C, D) \in \mathit{maxEC}(P) \,.\, \big[B = C \wedge C \cap R \neq \emptyset\big]\Big\} \;.$$

Then,

$$\sup_\eta \Pr^\eta_s(\phi) \;=\; \max_\eta \Pr^\eta_s(\phi) \;=\; \max_\eta \Pr^\eta_{(s,q_{in})}(\mathit{reach}(S_\phi)) \;,$$

where the rightmost reachability probability is computed in Π_ϕ.

The reachability probability mentioned by the theorem can be computed with the methods of the previous chapter. The proof of the theorem uses the following lemma, which will be useful also in Chapter 8.

Lemma 4.1 *Given a policy* η, *we can construct a policy* η' *such that*

$$\Pr^\eta_{(s,q_{in})}(\mathit{reach}(S_\phi)) = \Pr^\eta_{(s,q_{in})}(\mathit{reach}(S_\phi) \wedge \phi') \;.$$

Moreover, η' is Markovian in S_ϕ, i.e. the action chosen at a state in S_ϕ does not depend on the past history of the behavior.

Proof. We construct η' from η as follows. Let

$$\mathcal{L} = \Big\{(C,D) \;\Big|\; \exists (P,R) \in U' \,.\, \Big[(C,D) \in maxEC(P) \wedge C \cap R \neq \emptyset\Big]\Big\}$$

be the list of end components that contributed to S_ϕ. Intuitively, η' coincides with η outside of S_ϕ, and once in S_ϕ it will follow one of the end components of $\mathcal{L}$ forever. Note that these end components need not be mutually disjoint: the end components resulting from a single $(P,R) \in U'$ are mutually disjoint, but those corresponding to different acceptance pairs need not be so. To force the behavior to eventually follow one of these end components, we number them in some arbitrary order, so that $\mathcal{L}$ can be written as

$$\mathcal{L} = \Big\{(C_1,D_1),\ldots,(C_k,D_k)\Big\} \;.$$

For each $s \in S_\phi$, define the *index* $\iota(s)$ of s by $\iota(s) = \min\{i \mid s \in C_i\}$. Policy η' is then defined as follows. At $s \notin S_\phi$, η' coincides with η. At $s \in S_\phi$, η' chooses uniformly at random one action from $D_{\iota(s)}(s)$.

From the definition of $\iota(s)$ and η', once a behavior ω enters S_ϕ at position j_0, it is $\iota(X_j) \geq \iota(X_{j+1})$ for all $j \geq j_0$. Since $\iota(X_j)$ cannot decrease forever along ω, there is $1 \leq i \leq k$ such that $\mathit{inft}(\omega) = sa(C_i, D_i)$. Thus, if a behavior enters S_ϕ under η', it is $\omega \models \phi'$, and the lemma is proved. ∎

The proof of the theorem proceeds then as follows.

Proof of Theorem 4.2. By construction of Π', it is $\Pr_s^+(\phi) = \sup_\eta \Pr^\eta_{(s,q_{in})}(\phi')$. We can write

$$\Pr^\eta_{(s,q_{in})}(\phi') \;=\; \Pr^\eta_{(s,q_{in})}(\mathit{reach}(S_\phi) \wedge \phi') + \Pr^\eta_{(s,q_{in})}(\neg \mathit{reach}(S_\phi) \wedge \phi') \;. \tag{4.17}$$

A behavior $\omega \in \Omega_{(s,q_{in})}$ that satisfies ϕ' must, for some $(P,R) \in U'$,

(a) be eventually confined to P;

(b) visit infinitely often some state of R.

From (a), by Theorem 3.2, with probability 1 the behavior is eventually confined to the union of the end components in $maxEC(P)$. From (b), the behavior can be eventually confined only to the end components that share some state with R. Putting these two requirements together, we see that the behavior will be confined with probability 1 in S_ϕ. Thus, the second term on the right side of

(4.17) is 0, and (4.17) reduces to

$$\Pr^{\eta}_{(s,q_{in})}(\phi') = \Pr^{\eta}_{(s,q_{in})}(\mathit{reach}(S_\phi) \wedge \phi') \; . \tag{4.18}$$

Taking $\sup_\eta$ of both sides and using Lemma 4.1 we have

$$\begin{aligned} \sup_\eta \Pr^{\eta}_{(s,q_{in})}(\phi') \;&=\; \sup_\eta \Pr^{\eta}_{(s,q_{in})}(\mathit{reach}(S_\phi) \wedge \phi') \\ &=\; \sup_\eta \Pr^{\eta}_{(s,q_{in})}(\mathit{reach}(S_\phi)) \;=\; \max_\eta \Pr^{\eta}_{(s,q_{in})}(\mathit{reach}(S_\phi)) \; , \end{aligned}$$

where the last equality is justified by the results on probabilistic reachability described in the previous chapter. ∎

This final corollary summarizes the procedure for the model-checking of the P operator in pTL*.

Corollary 4.1 *Given an MDP Π and a sequence formula $\phi \in Seq$ of pTL*, the truth value of $\mathrm{P}_{\bowtie a}(\phi)$ can be decided at all states s of Π as follows:*

- $\bowtie \in \{\leq, <\}$:
$$s \models \Pr_{\bowtie a} \phi \quad \mathit{iff} \quad \max_\eta \Pr^{\eta}_{(s,q_{in})}(\mathit{reach}(S_\phi)) \bowtie a \; ,$$
where the probability on the right-hand side is computed on the MDP $\Pi_\phi = \Pi \otimes DR_\phi$, and S_ϕ is defined as in Theorem 4.2.

- $\bowtie \in \{\geq, >\}$:
$$s \models \Pr_{\bowtie a} \phi \quad \mathit{iff} \quad 1 - \max_\eta \Pr^{\eta}_{(s,q_{in})}(\mathit{reach}(S_{\neg\phi})) \bowtie a \; ,$$
where the probability on the right hand side is computed on the MDP $\Pi_{\neg\phi} = \Pi \otimes DR_{\neg\phi}$, and $S_{\neg\phi}$ is defined on $\Pi_{\neg\phi}$ as in Theorem 4.2.

4.5 Model Checking for Operator D in pTL and pTL*

Given a non-Zeno TPS $\Pi = (S, A, p, S_{in}, \mathit{time}, \mathcal{I})$, a state formula $\phi \in \mathit{Stat}$ and $a \geq 0$, we now consider the problem of computing the truth value of $\mathrm{D}_{\bowtie a}\phi$ all states $s \in S$. The model-checking algorithm we propose for this problem can be used for both pTL and pTL*. We assume that the truth value of ϕ has already been computed at all $s \in S$, and we define $S_\phi = \{s \in S \mid s \models \phi\}$.

By comparing (4.15) with (3.7), we see that the problem of the model-checking of the D operator is equivalent to an instance of the SSP problem, in which the set of destination states U is S_ϕ, the terminal cost g is always 0, and $c(s,a)$ is equal to either $\mathit{time}(s,a)$ or $-\mathit{time}(s,a)$, depending on whether we are interested in computing the minimum or maximum expected time to reach S_ϕ. To be able to use the results of Theorem 3.4, we must ensure that SSP Assumptions 1 and 2 hold. We treat separately the cases for $\bowtie \in \{\leq, <\}$ and $\bowtie \in \{\geq, >\}$.

4.5.1 Model Checking $\mathrm{D}_{\bowtie a}$ for $\bowtie \in \{\leq, <\}$

The model-checking for this case relies on the computation of the maximum expected time to S_ϕ. Let $B = \bigcup\{C \mid (C, D) \in maxEC(S - S_\phi)\}$ be the union of the end components not in S_ϕ, and let $\widehat{B} \subseteq S - S_\phi$ be the subset of states of $S - S_\phi$ from which B is reachable in the graph $(S - S_\phi, \rho_{S-S_\phi})$; of course, $B \subseteq \widehat{B}$. The following lemma, reminiscent of a result by Hart, Sharir and Pnueli [HSP82, HSP83], enables us to solve the model-checking problem on $\widehat{B}$.

Lemma 4.2 *For $\bowtie \in \{\leq, <\}$, the following assertions hold:*

$$\text{if } \; s \in S_\phi \; \text{ then } \; s \models \mathrm{D}_{\bowtie a}\phi \; ; \qquad\qquad \text{if } \; s \in \widehat{B} \; \text{ then } \; s \not\models \mathrm{D}_{\bowtie a}\phi \; .$$

Proof. The result for S_ϕ is immediate.

If $s \in \widehat{B}$, there is η such that $\Pr_s^\eta(reach(B)) > 0$. Hence, by Theorem 3.1 there is η' such that $\Pr_s^{\eta'}(inft(\omega) \subseteq sa(B, A_{\setminus B})) > 0$. This implies that $\Pr_s^{\eta'}(reach(S_\phi)) < 1$, or $\Pr_s^{\eta'}(T_\phi = \infty) > 0$. Since the MDP is non-Zeno, $\sum_{k=0}^{\infty} time(X_k, Y_k)$ diverges with probability 1 on a behavior. Thus,

$$\Pr_s^{\eta'}\left(\sum_{k=0}^{T_\phi - 1} time(X_k, Y_k) = \infty\right) > 0 \; ,$$

yielding

$$\mathrm{E}_s^{\eta'}\left\{\sum_{k=0}^{T_\phi - 1} time(X_k, Y_k)\right\} = \infty$$

as desired. ∎

We must still compute the truth value of $\mathrm{D}_{\bowtie a}\phi$ at the states in $C = S - (S_\phi \cup \widehat{B})$. Note that the set C, by construction, does not contain any end component, and that $\widehat{B}$ is not reachable from C.

Consider an SSP problem with set of states $C \cup S_\phi$ and set of destination states S_ϕ. The cost of a state-action pair (s, a) is equal to $-time(s, a)$. Since there are no end components in C, all policies are SSP-proper, so that SSP Assumptions 1 and 2 both hold. Restating Theorem 3.4 for this case we obtain the following theorem, which completes the model-checking algorithm.

Theorem 4.3 *Consider the following linear programming problem on the set of variables $\{v_s\}_{s \in C}$:*

Minimize $\sum_{s \in C} v_s$ subject to

$$v_s \geq time(s, a) + \sum_{t \in C} p_{st}(a)\, v_t \qquad\qquad s \in C, a \in A(s) \; .$$

The problem admits exactly one optimal solution $\boldsymbol{v}^\bullet$. For all $s \in C$ and $\bowtie \in \{\leq, <\}$, it is

$$s \models \mathrm{D}_{\bowtie a}\phi \;\; \textit{iff} \;\; v_s^\bullet \bowtie a \; .$$

4.5.2 Model Checking $\mathrm{D}_{\bowtie a}$ for $\bowtie \in \{\geq, >\}$

The model-checking for this case relies on the computation of the minimum expected time to S_ϕ. Let $B \subseteq S$ be the set of states that cannot reach S_ϕ. We have the following immediate result.

Lemma 4.3 *The following assertions hold:*

- *For $\bowtie \in \{\geq, >\}$, $a \geq 0$ and $s \in B$ it is $s \models \mathrm{D}_{\bowtie a}\phi$.*
- *If $s \in S_\phi$, then $s \models \mathrm{D}_{\bowtie a}\phi$ iff both $a = 0$ and $\bowtie$ is $\geq$.*

We can improve on the first part of this lemma by computing the sub-MDP consisting of the states that can reach S_ϕ with probability 1. This leads to a better conditioned linear programming problem, as it eliminates the states from which the expected time to S_ϕ diverges under any policy. To this end, let U_1 be the set computed by Algorithm 3.2 with input (S, A, p) and S_ϕ. Define $S' = S_\phi \cup U_1$, and

$$A'(s) = \{a \in A(s) \mid \mathrm{Succ}(s, a) \subseteq S'\}$$

for all $s \in S'$. Note that if $s \in S_\phi$ it is possible that $A'(s) = \emptyset$, but no ill consequence will follow from this. The following lemma can be used to decide the value of $\mathrm{D}_{\bowtie a}\phi$ at all states of $S - S'$.

Lemma 4.4 *If $s \in S - S'$, it is*

$$\mathrm{E}_s^\eta\left\{\sum_{k=0}^{T_\phi - 1} \mathit{time}(X_k, Y_k)\right\} = \infty$$

for any policy η. Thus, for $\bowtie \in \{\geq, >\}$, $a \geq 0$ and $s \in S - S'$ it is $s \models \mathrm{D}_{\bowtie a}\phi$.

Proof. If $s \in S - S'$, then $\Pr_s^\eta(\mathit{reach}(S_\phi)) < 1$ for all policies η, so that $\Pr_s^\eta(T_\phi = \infty) > 0$ for all η. Since the TPS is non-Zeno,

$$\Pr_s^\eta\left(\sum_{k=0}^{\infty} \mathit{time}(X_k, Y_k) = \infty\right) = 1 \ ,$$

for all η, from which follows

$$\mathrm{E}_s^\eta\left\{\sum_{k=0}^{\infty} \mathit{time}(X_k, Y_k)\right\} = \infty$$

for all η. This concludes the argument. ∎

The remaining problem consists in computing the truth value of $\mathrm{D}_{\bowtie a}\phi$ on the states in $C = S' - S_\phi$. This problem can be solved directly by a reduction to the SSP problem. The set of terminal states is $U = S_\phi$, the terminal cost function g is identically 0 on all states of U, and the cost function c is defined by $c(s, a) = \mathit{time}(s, a)$ for all $s \in C$ and $A \in A'(s)$.

To see that SSP Assumptions 1 and 2 hold, we reason as follows.

SSP Assumption 1. Since all states of C can reach S_ϕ, there is at least one Markovian policy which either reaches S_ϕ or leaves C with probability 1. This policy is thus SSP-proper.

SSP Assumption 2. Since the TPS is non-Zeno, the summation $\sum_{k=0}^{\infty} \mathit{time}(X_k, Y_k)$ diverges with probability 1 on a behavior. Thus, if η is a policy such that $\Pr_s^\eta(\mathit{reach}(U)) < 1$, then $\Pr_s^\eta(\sum_{k=0}^{T_U} \mathit{time}(X_k, Y_k) = \infty) > 0$, yielding $\mathrm{E}_s^\eta\{\sum_{k=0}^{T_U} \mathit{time}(X_k, Y_k)\} = \infty$, as desired.

From Theorem 3.4 we immediately obtain the following result, which concludes our presentation of model-checking algorithms for the logics pTL and pTL*.

Theorem 4.4 *Consider the following linear programming problem on the set of variables $\{v_s\}_{s\in C}$:*

Maximize $\sum_{s\in C} v_s$ subject to

$$v_s \leq \mathit{time}(s,a) + \sum_{t\in C} p_{st}(a)\, v_t \qquad\qquad s \in C, a \in A'(s)\ .$$

This linear programming problem admits exactly one optimal solution $\boldsymbol{v}^\bullet$. Moreover,

$$s \models \mathrm{D}_{\bowtie a}\phi \quad \mathit{iff} \quad v_s^\bullet \bowtie a$$

for all $s \in C$ and $\bowtie \in \{\geq, >\}$.

4.6 Complexity of pTL and pTL* Model Checking

To measure the complexity of the model-checking algorithms presented, we need first to define the *size* of an MDP.

Definition 4.2 (size of MDP) Given an MDP $\Pi = (S, A, p, \mathcal{L})$ with list of additional labels $\mathcal{L}$, we define the *graph size* of Π, indicated by $||\Pi||$, by $||\Pi|| \stackrel{def}{=} \sum_{s,a\in A(s)} |\mathrm{Succ}(s,a)|$.

We define the *size* of Π, indicated by $|\Pi|$, to be the length of its encoding, where we assume that:

- the transition probabilities p are represented by listing, for all $s, t \in S$ and $a \in A(s)$, the value of $p_{st}(a)$ expressed as ratio between integers (encoded in binary notation);
- the labels in $\mathcal{L}$ with type *integer* or *real* are represented by listing their values expressed as ratios between integers (encoded in binary notation);
- the other labels in $\mathcal{L}$, which in this dissertation are always sets of states or actions, are represented by simply listing their components.

We sometimes refer to $|\Pi|$ as the *description size* of the MDP, to emphasize the fact that $|\Pi|$ depends on the precision with which the transition probabilities are specified. ∎

Since we are interested in measuring the complexity of the temporal and probabilistic portion of the model-checking algorithms, when discussing their complexity we assume that the truth value of all formulas belonging to class *Atomic* has already been evaluated at all states of the system.

In any case we note that, since the state space of the system is finite, it is possible to assume without loss of expressive power that all variables in $\mathcal{V}$ have finite domain, all interpreted function symbols and predicates are specified by giving the relations defining them, and all atomic formulas are quantifier-free. Under these assumptions, computing the truth value of the atomic state formulas at all states can be done in polynomial time in the size of the system and domain.

Likewise, in the definition of length of a formula we abstract from the form of the atomic state formulas occurring in it.

Definition 4.3 (length of formula) We define the *length* of a pTL or pTL* formula ϕ to be the number of symbols composing ϕ, counting each occurrence of atomic state formula in ϕ as a single symbol. ∎

In this dissertation, we will examine only the time complexity of the algorithms (to which we will sometimes refer simply as the "complexity"), rather than their space complexity. Since the complexity of linear programming problems depends on the precision with which the coefficients of the problems are known [Sch87], the complexity of algorithms relying on linear programming will generally be polynomial in the description size of the TPS, rather than in its graph size. The following theorem summarizes the complexity of the model-checking algorithms for pTL and pTL*.

Theorem 4.5 (complexity of pTL, pTL* model checking) *Given a TPS Π, the following assertions hold:*

1. *Checking whether Π is non-Zeno can be done in polynomial time in $||\Pi||$.*

2. *Model checking a pTL formula ϕ has time-complexity linear in $|\phi|$ and polynomial in $|\Pi|$.*

3. *Model checking a pTL* formula ϕ has time-complexity doubly exponential in $|\phi|$ and polynomial in $|\Pi|$.*

Proof. The fact that the complexities given above are upper bounds is derived from an examination of the model-checking algorithms. In fact, the computation of end components and the other related graph-theoretical operations on MDPs can be done in time polynomial in $||\Pi||$; linear programming then makes the complexity polynomial in $|\Pi|$. The fact that the stated complexities are also lower bounds is a direct consequence of the lower bounds proved by Courcoubetis and Yannakakis [CY88, CY95]. ∎

4.7 Extending the D Operator to Past Formulas

To conclude, we discuss an extension of the logics pTL and pTL* that increases the expressive power of the logics by allowing the formula ϕ in $\mathrm{D}_{\bowtie b}\phi$ to be a *past* temporal formula, instead of a state formula. A past temporal formula is a formula constructed from state formulas in *Stat* using the temporal operators $\boxminus$, $\diamondminus$ and $\mathcal{S}$ [MP93].[1]

The semantics of this extension can be defined as follows. Given a behavior ω and a past formula ϕ, define the random time T_ϕ by $T_\phi = \min\{k \mid \omega \models_k \phi\}$, where $\omega \models_k \phi$ indicates that ϕ holds at position k of ω. The truth value of $\mathrm{D}_{\bowtie b}\phi$ can be defined as in (4.15).

This extended version of the D operator can be model checked by combining the techniques of [BdA95] with the algorithms presented in this chapter. Specifically, given a TPS Π and a formula $\mathrm{D}_{\bowtie b}\phi$, it is possible to construct a TPS Π^p_ϕ in which the states keep track of the truth values of the past subformulas of ϕ (ϕ itself included). The truth value of $\mathrm{D}_{\bowtie b}\phi$ can be computed by applying the algorithms of Section 4.5 to the TPS Π^p_ϕ. Since the complexity of the model checking is dominated by the doubly-exponential dependency arising from the operator P, the bounds expressed by Theorem 4.5 apply also to this extended version of the logic.

4.8 Model Checking of Operator P in pTL (‡)

Since the logic pTL is a subset of PTL*, the model checking of operator P in pTL can be done using the algorithm presented for pTL*. However, in the case of pTL it is also possible to use an algorithm due to Bianco and de Alfaro [BdA95], which takes advantage of the simpler structure of pTL. In this section, we present the algorithm of [BdA95] restated in terms of end components.

First, note that for every state $s \in S$, formula $\phi \in \mathit{Seq}$ and policy η, equation (4.16) holds. Thus, checking that $\mathrm{Pr}^\eta_s(\phi) \bowtie a$ for all η is equivalent to checking that $1 - \mathrm{Pr}^\eta_s(\neg\phi) \bowtie a$ for all η, or

$$\mathrm{Pr}^\eta_s(\neg\phi) \mathrel{\widehat{\bowtie}} (1-a)$$

for all η, where $\widehat{\bowtie}$ is the complementary inequality of $\bowtie$. Using this idea, together with the equivalences

$$\Box\psi \leftrightarrow \neg\Diamond\neg\psi \qquad\qquad \Diamond\psi \leftrightarrow \mathit{true}\,\mathcal{U}\,\psi$$

we see that we need to consider only the case of sequence formulas ϕ of the form $\phi = \gamma\,\mathcal{U}\,\psi$. Let $S_d = \{s \in S \mid s \models \psi\}$ be the set of destination states, and let $S_p = \{s \in S - S_d \mid s \models \gamma\}$ be the set

[1] The use of past temporal operators in non-probabilistic branching-time logics is discussed in depth by Kupferman and Pnueli [KP95].

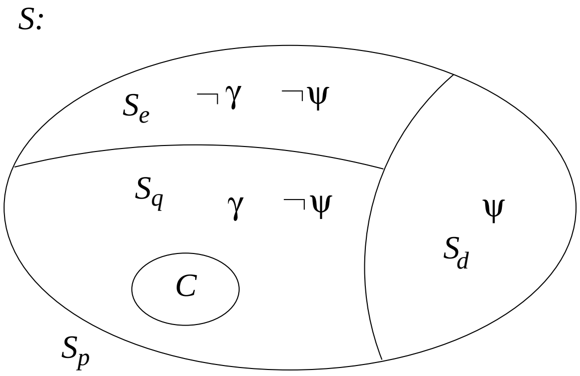

Figure 4.1: Containment relationships between the set of states S_d, S_e, S_p, S_q and $C \subseteq S_p$. In the sets are also indicated the truth-values of the subformulas γ, ψ of $\gamma \,\mathcal{U}\, \psi$. Formulas γ and $\neg\psi$ hold also in C, since $C \subseteq S_p$.

of intermediate states. Let also

$$S_e = S - (S_d \cup S_p) = \{s \mid s \models \neg\gamma \wedge \neg\psi\} \ .$$

Informally, formula $\gamma \,\mathcal{U}\, \psi$ holds for all behaviors that eventually enter S_d, and enter S_d before entering S_e.

We consider two cases, for $\bowtie \in \{\geq, >\}$ and $\bowtie \in \{\leq, <\}$. In the first case, we have to determine the minimal probability $\Pr_s^-(\phi) = \min_\eta \Pr_s^\eta(\phi)$; in the second case the maximal probability $\Pr_s^+(\phi) = \max_\eta \Pr_s^\eta(\phi)$. The arguments will also show the existence of such maxima and minima (and not only suprema and infima).

4.8.1 Computation of $\Pr_s^-(\phi)$.

Let $\{(C_1, D_1), \ldots, (C_n, D_n)\} = maxEC(S_p)$, and let $C = \bigcup_{i=1}^n C_i$. We have the following result.

Lemma 4.5 *The value of $\Pr_s^-(\phi)$ on S_e, C and S_d is as follows:*

$$\begin{aligned} s \in S_e : &\quad \Pr_s^-(\phi) = 0 \\ s \in C : &\quad \Pr_s^-(\phi) = 0 \\ s \in S_d : &\quad \Pr_s^-(\phi) = 1 \ . \end{aligned}$$

Proof. The results for S_e and S_d are immediate. For $s \in C$, notice that there is a policy η such that $\Pr_s^\eta(\forall k \geq 0 \,.\, X_k \in C) = 1$. This leads to the result. ∎

Let $S_q = S_p - C$ be the set of states on which $\Pr_s^-(\phi)$ must still be determined. The relation between S_d, S_e, S_p, S_q and C is illustrated in Figure 4.1. Note that a behavior will leave S_q with probability 1, since there are no end components in S_q. We now reduce the problem of computing $\Pr_s^-(\phi)$ at all $s \in S_q$ to the problem of solving an instance of the SSP problem. Let the set of destination states be $U = S_d \cup S_e \cup C$, and consider a modified MDP Π' defined as follows:

- each state $s \in U$ has a single action a that leads deterministically to s;
- for $s \in U$, let $g(s) = 1$ if $s \in S_d$, and $g(s) = 0$ for $s \in S_e \cup C$;
- for $s \in S_q$, let $c(s,a) = 0$ for all $a \in A(s)$.

Denoting by $\widehat{\Pr}$ the probability structure corresponding to Π', we have

$$\Pr_s^\eta(\phi) = \widehat{\Pr}_s^\eta(\mathit{reach}(S_d)) = v_s^\eta$$

for all $s \in S_q$, where the cost v_s^η of η at s is computed in the MDP Π'. Since S_q does not contain any end component, all Markovian policies in Π' are SSP-proper, and SSP Assumptions 1 and 2 hold. Theorem 3.4 provides then the desired model-checking algorithm.

Theorem 4.6 *Consider the following linear programming problem on the set of variables $\{v_s\}_{s \in S_q}$:*

Maximize $\sum_{s \in S_q} v_s$ subject to

$$v_s \leq \sum_{t \in S_q} p_{st}(a)\, v_t + \sum_{t \in S_d} p_{st}(a) \qquad s \in S_q \ .$$

This linear programming problem admits exactly one optimal solution $\mathbf{v}^\bullet$. The truth value of $\mathrm{P}_{\bowtie a}(\phi)$ for $\bowtie \in \{\geq, >\}$ can be decided at all $s \in S_q$ by

$$s \models \mathrm{P}_{\bowtie a}\phi \quad \textit{iff} \quad v_s^\bullet \bowtie a \ .$$

4.8.2 Computation of $\Pr_s^+(\phi)$.

Let S_r be the set of the states of S_p from which there is a path to S_d in the graph $(S_d \cup S_p, A_{\backslash (S_d \cup S_p)})$. In other words, S_r consists of the states of S_p that can reach S_d while following a path contained in $S_p \cup S_d$. Figure 4.2 summarizes the partition of the state space for this case.

The following lemma states that $\Pr_s^+(\phi) > 0$ iff $s \in S_r \cup S_d$.

Lemma 4.6 *If $s \in S_d$, then $\Pr_s^+(\phi) = 1$. Moreover, $\Pr_s^+(\phi) > 0$ iff $s \in S_r \cup S_d$.*

Proof. Immediate. ∎

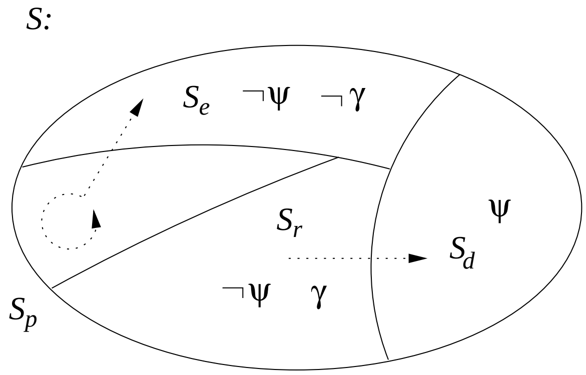

Figure 4.2: Containment relationships between the set of states S_d, S_e, S_p and $S_r \subseteq S_p$. In the sets are also indicated the truth-values of the subformulas γ, ψ of $\gamma\,\mathcal{U}\,\psi$. Formulas γ and $\neg\psi$ hold also in S_r, since $S_r \subseteq S_p$. The dotted arrows provide a reminder of the definition of S_r, which consists of the states of S_p that can reach S_d without leaving $S_p \cup S_d$. Thus, a behavior from $S_p - S_r$ must either be confined to $S_p - S_r$, or proceed to S_e.

To complete the model-checking algorithm, we must compute $\mathrm{Pr}_s^+(\phi)$ at all $s \in S_r$. To this end, we let $\Pi' = (S, A', p')$ be the modified MDP obtained by assigning to each state $s \in S - S_r$ a single action a that leads deterministically to s; the probability function p' coincides with p on all other state-action pairs. It is easy to see that the probability of satisfying ϕ is equal to the probability of reaching S_d in Π': in formulas,

$$\mathrm{Pr}_s^\eta(\phi) = \widehat{\mathrm{Pr}}_s^\eta(\mathit{reach}(S_d)) \ ,$$

where the probability $\widehat{\mathrm{Pr}}$ is computed on the MDP Π'. To compute $\mathrm{Pr}_s^+(\phi)$ at $s \in S_r$ is suffices then to compute the maximum reachability probability $\sup_\eta \widehat{\mathrm{Pr}}_s^\eta(\mathit{reach}(S_d))$, which can be done with the methods presented in Section 3.5.

The following theorem summarizes the result.

Theorem 4.7 (model checking of operator P **in pTL)** *For every* $s \in S_r$*, operator* $\bowtie \in \{\leq, <\}$ *and bound* $0 \leq a \leq 1$*, it is*

$$s \models \mathrm{P}_{\bowtie a}\phi \quad \textit{iff} \quad \mathrm{Pr}_s^+(\phi) \bowtie a \ .$$

Chapter 5

Specification of Long-Run Average Properties

In the discipline of performance modeling and evaluation, the system models are usually translated into Markov chains with additional labels to represent quantities of interest. The performance evaluation of a system involves the computation of the steady-state probability distribution of the chain, which provides information about the long-run average behavior of the system. This approach is at the basis of the evaluation of performance measures such as the average system throughput and response time.

By contrast, the model checking of pTL and pTL* specifications on Markov chains does not require an analysis of the steady-state distribution, but the computation of reachability probability of closed recurrent classes or other sets of states of the chain. This suggests that the class of properties expressible in pTL and pTL* does not include these traditional performance measures.

In this chapter we will show that this is indeed the case. To remedy to this deficiency, we introduce a specification style based on *experiments.* Experiments are small labeled graphs that specify behaviors and average times of interest for the performance and reliability analysis of the system. Experiments are composed synchronously with the system, and they can be used to measure the long-run average value of the performance or reliability parameters they specify.

After introducing experiments and the logics based on them, we discuss the classification of probabilistic properties mentioned in the introduction to this dissertation. This classification justifies in retrospect the need for the four probabilistic operators used in our logics.

Due to the presence of nondeterminism in the system models, the evaluation of long-run average properties of systems is based on more involved algorithms than the computation of the steady-state distribution of Markov chains. The algorithms for the model checking of the new logics will be presented in the next chapter, along with their correctness proofs.

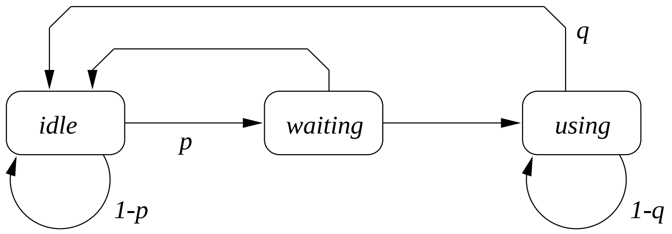

Figure 5.1: Behavior of a single user in system SHARED-RES. The probabilities of the two transitions from state *waiting* depend on the states of the other users in the system.

5.1 A Motivational Example

To gain a better understanding of the limitations of pTL and pTL* with respect to the specification of long-run average behavior, we introduce our first example: a system consisting of many users sharing access to a single resource.

5.1.1 Multiple Shared Resource

The system SHARED-RES consists of N users that can access a shared resource. The resource can be used by at most $M \leq N$ users at any single time. Each user can be in one of three states: *idle*, *requesting* and *using*. For the sake of simplicity, we model the behavior of the system as a discrete-time Markov chain. Initially, all users are in *idle*. At each time step, if a user is *idle* it has probability p of going to *requesting* and $1-p$ of staying in *idle*. If the user is in *using*, it has probability q of going to *idle*, and $1-q$ of staying in requesting. If at the beginning of a time step there are j users in *requesting* and $k \leq M$ in *using*, at the next time step $\min\{j, M-k\}$ users will go to *using*, while the remaining $j - \min\{j, M-k\}$ are sent back to *idle*. The behavior of a single user is depicted in Figure 5.1.

This system was originally devised as a very simple model for the behavior of people placing phone calls: N is the number of people, and M is the maximum number of calls that can be active at the same time. The transition from *idle* to *requesting* corresponds to the act of lifting the handset to place a call; the transition from *using* to *idle* corresponds to hanging up at the end of the call. The transitions out of the *requesting* state model the acts of either getting the connection or of hanging up upon hearing the busy signal.

From the above informal description, for given N, M, p and q it is possible to produce a TPS $\Pi(N, M, p, q)$.

5.1.2 Specifying Call Throughput of SHARED-RES

To exemplify the limitations of pTL and pTL*, we will take the following property as our intended specification.

Req1: *When a user tries to place a call, the probability of getting the connection is at least b_0.*

As the situation is symmetrical for all users, let us concentrate on the first one. Let I_1, R_1, U_1 be atomic formulas representing the fact that user 1 is at *idle, requesting* or *using*, respectively. It might seem plausible at first to encode the requirement Req1 with the following pTL formula:

$$\tilde{\phi}_{Req1} : \mathrm{A}\Box(R_1 \rightarrow \mathrm{P}_{\geq b_0}(R_1 \, \mathcal{U} \, U_1)) \; . \tag{5.1}$$

Let us analyze this formula. Call any state that satisfies R_1 and that is reachable from the initial state s_{in} a *reachable R_1-state*. By definition, $s_{in} \models \tilde{\phi}_{Req1}$ iff every reachable R_1-state satisfies $\mathrm{P}_{\geq 0.8}(R_1 \, \mathcal{U} \, U_1)$. The formula in parentheses specifies that the U_1-state is reached directly from the R_1 state, rather than as a consequence of a later request.

The problem with specification $\tilde{\phi}_{Req1}$ is that there are many reachable R_1-states in the system: in some of them, few users are accessing the resource; in others, the resource is fully utilized. Clearly, from these latter R_1-states, the probability that the first user gets access to the resource is 0. Thus, as long as there is a reachable R_1-state in which already M users are using the resource, the specification $\tilde{\phi}_{Req1}$ will not be satisfied, regardless of the long-run average probability with which the first user succeeds in accessing the resource.

Example 5.1 Figure 5.2 depicts the steady-state distribution and the access probability for one instance of the SHARED-RES system. Note that at states in which M processes are using the resource the probability of access is 0. Since these states are reachable, the property $\tilde{\phi}_{Req1}$ will not hold. However, calculations done using the steady-state distribution of the chain indicate that a call attempt is successful with a long-run average probability of approximately 0.88. ∎

More generally, the problem is due to the fact that the logics pTL and pTL*, while suited for the specification of the probability with which behaviors satisfy temporal formulas from given states, do not take into account the long-run probability of being in those states.

5.1.3 Possible Pitfalls in Extending the Logics

Let us derive a mathematical expression that encodes specification Req 1. It is easy to see that SHARED-RES is an ergodic Markov chain, i.e. a Markov chain in which all states belong to a single closed recurrent class [KSK66]. Thus, there is a steady-state distribution π, that gives for each state s the long-run probability π_s of being at s. For each state s, let $r_s = \Pr_s(X_1 \models U_1)$ be the probability that user 1 is using the resource in the next time unit. Since a user cannot continuously

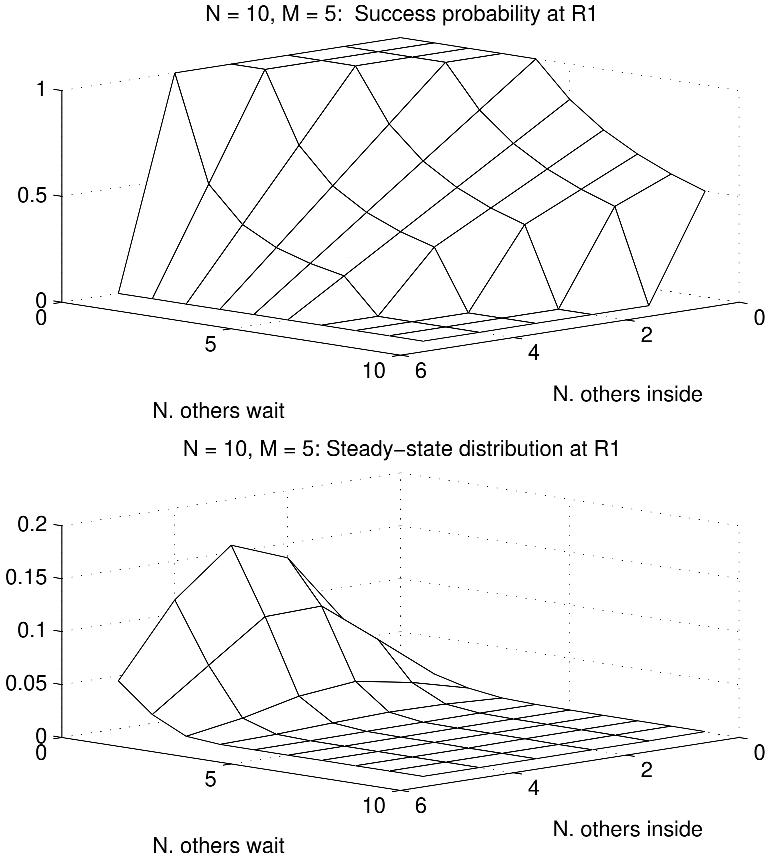

Figure 5.2: Probability of successful access and steady-state distribution for system SHARED-RES, with $N = 10$, $M = 5$, $p = 0.1$, $q = 0.2$, taken at R_1. In the diagrams, one of the horizontal axes indicates how many users are accessing the resource when user 1 is at *requesting;* the other horizontal axis indicates how many users aside from user 1 are in *requesting.* The top diagram indicates the success probability $\Pr_s(R_1 \,\mathcal{U}\, U_1)$, conditional to the fact that user 1 is at *requesting.* The diagram below shows the steady-state distribution of the system, conditional to the fact that user 1 is at *requesting.*

stay in *requesting* for longer than one time unit, the long-run fraction of successful requests of user 1 is given by:

$$\frac{\sum_{s \models R_1} \pi_s r_s}{\sum_{s \models R_1} \pi_s} \tag{5.2}$$

This equation suggests the following approach to the specification of average-case behavior of Markov chains. Assume that we are interested in the long-run probability of ϕ starting from a ψ-state, and let $S_\psi \stackrel{def}{=} \{s \in S \mid s \models \psi\}$. Let P be the transition matrix of the chain, and let

$$P^* = \lim_{n \to \infty} \frac{1}{n} \sum_{k=0}^{n-1} P^k ,$$

be the limiting matrix of the chain. If the system is ergodic, all rows of the matrix are equal, and their common value is the steady-state distribution π. For all $s \in S_\psi$ let $r(s) \stackrel{def}{=} \Pr_s(\phi)$ be the probability of ϕ holding after starting from state s. For all $s \in S$ such that $\sum_{t \in S_\psi} P^*_{st} > 0$ (i.e., for all s that can reach S_ψ),

$$q_s \stackrel{def}{=} \frac{\sum_{t \in S_\psi} P^*_{st} r_t}{\sum_{t \in S_\psi} P^*_{st}} .$$

represents the average probability that ϕ follows a ψ-state, when starting at s. Finally, we could define the operator L by

$$s \models \mathrm{L}_{\geq b}(\psi \leadsto \phi) \quad \textit{iff} \quad q_s \geq b \vee \sum_{t \in S_\psi} P^*_{st} = 0 . \tag{5.3}$$

Requirement Req1 could then be expressed by the formula

$$\mathrm{L}_{\geq b_0}(R_1 \leadsto (R_1 \, \mathcal{U} \, U_1)) . \tag{5.4}$$

Example 5.2 For the instance of SHARED-RES described in Example 5.1, the long-run probability that a request from user 1 is granted is given by

$$q_{S_{in}} = \frac{\sum_{s \models R_1} \pi_s r(s)}{\sum_{s \models R_1} \pi_s} \simeq 0.88 .$$

This value can be compared with b_0 to decide whether specification (5.4) is satisfied. ∎

While this approach works well for Markov chains, it does not generalize properly to TPS. To see this, assume for simplicity that $\psi \equiv$ *true*,[1] so that $S_\psi = S$, and consider the modifications that

[1] It is possible to present verification methods for the general case using the results presented later in the chapter. However, we omit the details, since we do not advocate following this approach.

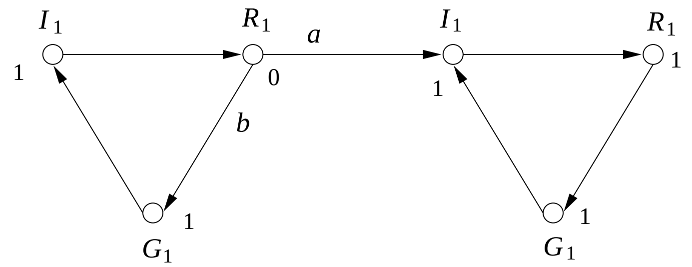

Figure 5.3: A TPS Π, with indicated the extensions of the atomic formulas I_1, R_1, G_1. The transitions are deterministic. Each state is labeled with its value of r, as defined by (5.5).

are necessary to take into account the existence of policies. The definition of r becomes

$$r(s) = \min_{\eta} \Pr_s^{\eta}(\phi) \ ; \tag{5.5}$$

its value can be computed with the results of the previous chapter. Let

$$V_s^{\eta} \stackrel{def}{=} \lim_{n \to \infty} \mathrm{E}_s^{\eta} \left\{ \frac{1}{n} \sum_{k=0}^{n-1} r(X_k) \right\}$$

be the average value of r when using policy η starting from s, and let

$$V_s^{-} = \min_{\eta} V_s^{\eta} \tag{5.6}$$

be minimum of this average over all policies. The value V_s^{-} can be computed by using classical results about the computation of the minimum average reward of MDPs (see, for example, Derman [Der70] and Bertsekas [Ber95]). Then, the obvious generalization of (5.3) is (remembering that $\psi \equiv \mathit{true}$):

$$s \models \mathrm{L}_{\geq b}(\psi \rightsquigarrow \phi) \quad \mathit{iff} \quad V_s^{-} \geq b \ . \tag{5.7}$$

The cases for the other inequalities $<$, $\leq$ and $>$ can be defined similarly. The problem with this definition is that the policy η_1 that realizes the minimum in (5.5) can be different from the policy η_2 that realizes the minimum in (5.6). To understand why this is a problem, assume that ϕ expresses the property of something good happening. Then, η_1 will minimize the probability $\Pr_s^{\eta_1}(\phi)$ of something good happening from any given state s. This minimal probability is $r(s)$. Policy η_2 will try to maximize the amount of time the TPS spends in "risky" states, i.e. in states s where $r(s)$ is small. If $\eta_1 \neq \eta_2$, it might very well be that the policy that maximizes the time spent in risky states will actually ensure that the "danger" $\neg\phi$ will never strike! The following example illustrates this apparently paradoxical situation.

Example 5.3 Consider the TPS depicted in Figure 5.3. We would like to write a specification requiring that the long-run average probability of accessing the resource upon a request is at least 0.9. Intuitively, this property should hold, since at most one call attempt can fail along a behavior. However, the specification

$$\mathrm{L}_{\geq 0.9}(R_1 \rightarrow R_1 \,\mathcal{U}\, G_1)$$

does not hold. In fact, the policy η_0 that realizes the minimum of

$$r(s) = \min_{\eta} \Pr_s^{\eta}(R_1 \rightarrow R_1 \,\mathcal{U}\, G_1)$$

always chooses action a, and yields the values for r depicted in the figure. The policy η_1 that minimizes the long-run average value of r chooses instead always action b, and leads to $V_s^{\eta_1} = 2/3$ for some states s. It can be seen that the specification

$$\mathrm{L}_{\geq 0.9}(R_1 \rightsquigarrow R_1 \,\mathcal{U}\, G_1)$$

would suffer from an analogous problem. In fact, the policy η_1' that minimizes the probability of $R_1 \,\mathcal{U}\, G_1$ would again choose always action a, yielding the values of r depicted in the figure. The policy that minimizes the long-run average value of r from R_1-states would instead always choose action b, yielding a value of 0 for this long-run average from some states. ∎

Fundamentally, the definition of L suffers from the problem of *policy multiplicity:* the definition depends on the nested use of policies, and the policies that are optimal/pessimal at different levels may not be the same. This nested use of policies makes the specifications difficult to understand. This problem is shared by other specification languages based on the nested use of policies, and it is the reason why we do not advocate the nesting of probabilistic operators in our specification languages, even though the syntax and semantics of our logics enables such nesting.

5.2 GPTL and GPTL*: Augmenting the Logics With Experiments

Our choice of a specification language for the long-run behavior or probabilistic systems has been guided by the following three requirements:

- It should be applicable not only to Markov chains, but also to TPS, which include nondeterminism.
- It should enable the specification of long-run properties of systems without leading to pitfalls such as the one discussed in the previous subsection.

- The model-checking problem should have polynomial time-complexity in the size of the TPS.

In the remainder of the chapter we present an extension of pTL and pTL* that satisfies all three of these requirements, and that is based on the concept of *experiments*.

5.2.1 Experiments

An *experiment* is simply a finite deterministic automaton, with a distinguished initial state and some additional labelings. Experiments describe system behaviors of interest, and are applied to a TPS by forming the synchronous composition between the experiment and the TPS. Experiments thus are reminiscent of the tests of Wang and Larsen [WL92] and Jonsson, Ho-Stuart and Wang [JHW94], except that experiments, unlike tests, are meant to be repeated an infinite number of times. Each time an experiment is performed, it yields an outcome, or a duration. Accordingly, we distinguish two types of experiments: *P-experiments,* to measure probability, and *D-experiments,* to measure durations.

The specification of properties using experiments is mediated by the two logical operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$, which enable to specify bounds for the long-run average outcome or duration of experiments. Before detailing how these long-run average outcomes and durations are measured, we define *experiments,* which provide the common structure for both P- and D-experiments.

Definition 5.1 (experiment) An *experiment* $\Psi = (V, E, E_r, V_{in}, \lambda)$ is a labeled directed graph (V, E), with set of vertices V and set of edges $E \subseteq V \times V$. The experiment has a distinguished set of *initial vertices* $V_{in} \subseteq V$, a set of *reset edges* $E_r \subseteq E - \{(v,v) \mid v \in V\}$, and a labeling $\lambda : V \mapsto \mathit{Form}(\mathcal{V})$. For all $u \in V$, edge (u,u) must belong to E. For $u \in V$, we denote by $\mathit{dst}(u) = \{v \in V \mid (u,v) \in E\}$ the set of vertices that can be reached in one step from u (thus, $u \in \mathit{dst}(u)$). The labeling of the vertices must be deterministic and total. Specifically, for all $u, v_1, v_1 \in V$ it must be

$$v_1 \in \mathit{dst}(u) \wedge v_2 \in \mathit{dst}(u) \quad \to \quad \neg[\lambda(v_1) \wedge \lambda(v_2)] \tag{5.8}$$

$$v_1 \in V_{in} \wedge v_2 \in V_{in} \quad \to \quad \neg[\lambda(v_1) \wedge \lambda(v_2)] \; ; \tag{5.9}$$

moreover, for every $v \in V$ the formulas

$$\bigvee_{v \in \mathit{dst}(u)} \lambda(v) \qquad\qquad \bigvee_{v \in V_{in}} \lambda(v) \tag{5.10}$$

must be valid (i.e. true in any type-consistent variable interpretation). ∎

When an experiment Ψ is applied to a PTS Π by taking the synchronous composition $\Pi \otimes \Psi$, the vertex labels λ of Ψ are used to synchronize the transitions of Ψ and Π. The fact that $u \in \mathit{dst}(u)$

for all $u \in V$ ensures that, if the variable assignment does not change, the experiment remains at the same vertex.

The reset edges are used to count the number of experiments performed: each time a reset edge is traversed, we say that the experiment *ends,* so that the number of reset edges traversed along a behavior indicates how many experiments have been completed. To keep track of this number, we define a labeling w as follows.

Definition 5.2 (w labeling of experiment edges) Given an experiment $\Psi = (V, E, E_r, V_{in}, \lambda)$, for each $(u, v) \in E$ we define $w(u, v) = 1$ if $(u, v) \in E_r$, and $w(u, v) = 0$ otherwise. ∎

In a P-experiment, to each reset edge is associated an *outcome*, a real number representing a reward earned when the experiment is ended. This will make it possible to talk about the long-run average outcome of P-experiments.

Definition 5.3 (P-experiment) A *P-experiment* is an experiment in which each reset edge $(u, v) \in E_r$ is labeled with an *outcome* $\gamma(u, v) \in \mathbb{R}$, a real number representing the success, or failure, of the experiment. ∎

In many cases, we will be interested in experiments whose outcome is binary: they can either succeed or fail. In this case, we will associate outcome 1 to the reset edges that represent a successful completion of the experiment, and outcome 0 to the reset edges representing failures.

In a D-experiment, we distinguish a set of *timed* vertices: the duration of a D-experiment will be defined as the total time spent at timed vertices during the experiment.

Definition 5.4 (D-experiment) A *D-experiment* is an experiment in which we distinguish a nonempty subset $V_t \subseteq V$ of *timed vertices.* ∎

Drawing experiments. To draw an experiment, we draw its graph structure, depicting reset edges with dashed lines. The initial vertices are represented by a double-circle. In D-experiments, timed vertices are drawn as filled circles, and untimed ones as empty circles. Since every vertex has a self-loop to itself, we omit the self-loops, to reduce clutter. Similarly, we often omit from experiments edges that are never followed by the system in which we are interested: the experiments we depict can easily be completed with additional edges to comply with (5.8) and (5.10).

Example 5.4 (average success in SHARED-RES) In Figure 5.4 we present a P-experiment that can be used to express Requirement Req1. If user 1 proceeds from *requesting* to *using*, and then from *using* to *idle*, the experiment has outcome 1; if after *using* user 1 proceeds to *idle*, the outcome is 0, indicating an unsuccessful attempt. Requirement Req1 will be encoded by stating that the average outcome of the experiment should be at least b_0. ∎

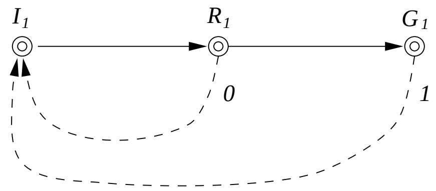

Figure 5.4: P-experiment Ψ_{Req1}, for the specification of Requirement Req1 of SHARED-RES.

5.2.2 Long-Run Average Outcome of Experiments

To use experiments for the specification of long-run probabilistic properties of systems, we need to compose them with the system and to define their *long-run average outcome*. The composition with a system is defined as follows.

Definition 5.5 (product of TPS and experiment) Given a TPS $\Pi = (S, A, p, S_{in}, time, \mathcal{I})$ and an experiment $\Psi = (V, E, E_r, V_{in}, \lambda)$, we define their *product MDP* $\Pi_\Psi = \Pi \otimes \Psi = (\widehat{S}, \widehat{A}, \widehat{p}, r, w)$ as follows:

- $\widehat{S} = \{\langle s, u\rangle \mid s \models \lambda(u)\}$;
- $\widehat{A}(\langle s, u\rangle) = A(s)$;
- For all states $\langle s, u\rangle \in \widehat{S}$, actions $a \in A(s)$ and destination states $t \in \mathrm{Succ}(s, a)$, the transition structure from $\langle s, u\rangle$ is as follows. Let $v_u^t \in V$ be the unique vertex such that

$$t \models \lambda(v_u^t) \qquad\qquad (u, v_u^t) \in E \ .$$

  If Π is at s and Ψ at u, and Π takes a transition to t, Ψ will take a transition to v_u^t. The transition probabilities $\widehat{p}$ of Π_Ψ and the labeling w are defined as follows:

$$\widehat{p}_{\langle s,u\rangle\langle t,v\rangle}(a) = \begin{cases} p_{st}(a) & \text{if } v = v_u^t; \\ 0 & \text{otherwise} \end{cases} \qquad\qquad w(\langle s, u\rangle, \langle t, v\rangle) = w(u, v) \ .$$

  If Ψ is a P-experiment, we let $r(\langle s, u\rangle, a, \langle t, v\rangle) = w(u, v)\,\gamma(u, v)$; if it is a D-experiment, we let $r(\langle s, u\rangle, a, \langle t, v\rangle)$ be $time(s, a)$ if u is a timed vertex, and 0 otherwise.

As a consequence of (5.9), to each $s \in S$ corresponds a *unique* vertex $u \in V_{in}$ such that $\langle s, u\rangle \in \widehat{S}$. We denote this unique vertex by $v_{in}(s)$, for $s \in S$. ∎

In the product MDP, the sum $\sum_{k=0}^{n-1} w(X_k, X_{k+1})$ indicates how many experiments have been completed in the first n steps of a behavior. Similarly, the sum $\sum_{k=0}^{n-1} r(X_k, Y_k, X_{k+1})$ indicates the total "outcome" for the first n steps, where the outcome of D-experiments is the time spent at timed vertices.

5.2.3 The Extended Logics

To enable the specification of long-run average properties of systems, we introduce the logics GPTL and GPTL*, which extend pTL and pTL* respectively by introducing two new operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$. These operators are used to express bounds on the average long-run outcome of experiments from given starting states. For $\bowtie \in \{<, \leq, \geq, >\}$ and $a \in \mathbb{R}$, we introduce the following state formulas.

- If Ψ is a P-experiment, then $\bar{\mathrm{P}}_{\bowtie a}(\Psi)$ is a state formula.
- If Ψ is a D-experiment, then $\bar{\mathrm{D}}_{\bowtie a}(\Psi)$ is a state formula.

Intuitively, $\bar{\mathrm{P}}_{\bowtie a}(\Psi)$ holds at a state s if, under any policy, a behavior that performs infinitely many experiments yields a long-run average outcome that is $\bowtie a$ with probability 1. Similarly, $\bar{\mathrm{D}}_{\bowtie a}(\Psi)$ holds at a state s if, under any policy, a behavior that performs infinitely many experiments yields a long-run average experiment duration that is $\bowtie a$ with probability 1.

The definition of the semantics of operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ relies on the definition of two additional quantities. First, we define the *n-stage average outcome* of a behavior.

Definition 5.6 (n-stage average outcome) Given a behavior ω of a product between a TPS and an experiment, we define the *n-stage average outcome* of ω by

$$\mathcal{H}_n(\omega) = \frac{\displaystyle\sum_{k=0}^{n-1} r(X_k, Y_k, X_{k+1})}{\displaystyle\sum_{k=0}^{n-1} w(X_k, X_{k+1})} . \tag{5.11}$$

This outcome represents the average outcome or duration of an experiment in the first n positions of the behavior. ∎

Then, we define predicate I as follows.

Definition 5.7 (predicate I) Given a behavior ω of a product between a TPS and an experiment, we define the predicate I by:

$$\Psi \textit{ is a P-experiment:} \qquad \omega \models I \textit{ iff } \sum_{k=0}^{\infty} w(X_k, X_{k+1}) = \infty ;$$

$$\Psi \textit{ is a D-experiment:} \qquad \omega \models I \textit{ iff } \sum_{k=0}^{\infty} \Big[w(X_k, X_{k+1}) + r(X_k, Y_k, X_{k+1}) \Big] = \infty .$$

Thus, I holds either if ω spends an infinite amount of time at timed vertices, or if ω performs infinitely many experiments. It is not difficult to show that the truth-set of I is measurable. ∎

The semantics of operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ can then be defined in terms of $\mathcal{H}$ and I as follows.

Definition 5.8 (semantics of $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$) Given a TPS $\Pi = (S, A, p, S_{in}, \mathit{time}, \mathcal{I})$ and an experiment $\Psi = (V, E, E_r, V_{in}, \lambda)$, let $\Pi_\Psi = \Pi \otimes \Psi$ be the product MDP.

First, we define the semantics of $\bar{\mathrm{P}}$, $\bar{\mathrm{D}}$ on Π_Ψ. For a state $\langle s, v\rangle$ of Π_Ψ and for $\bowtie \in \{\geq, >\}$, $\langle s, v\rangle \models \bar{\mathrm{P}}_{\bowtie a}\Psi$ (resp. $\langle s, v\rangle \models \bar{\mathrm{D}}_{\bowtie a}\Psi$) holds if, for all policies η,

$$\Pr^{\eta}_{\langle s,v\rangle}\Big(I \rightarrow \liminf_{n\to\infty} \mathcal{H}_n(\omega) \bowtie a\Big) = 1 \; . \tag{5.12}$$

The definition of $\langle s, v\rangle \models \bar{\mathrm{P}}_{\bowtie a}\Psi$ (resp. $\langle s, v\rangle \models \bar{\mathrm{D}}_{\bowtie a}\Psi$) for $\bowtie \in \{\leq, <\}$ is analogous. The semantics of $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ on Π is defined by taking, for all states $s \in S$ of Π,

$$s \models \Xi_{\bowtie a}(\Psi) \quad \mathit{iff} \quad \langle s, v_{in}(s)\rangle \models \Xi_{\bowtie a}(\Psi) \; ,$$

where Ξ is one of $\bar{\mathrm{P}}$, $\bar{\mathrm{D}}$ and $\bowtie \in \{<, \leq, \geq, >\}$. ∎

According to the way r and w have been defined, $\mathcal{H}_n(\omega)$ represents the average outcome or duration of an experiment in the first n positions of a behavior. The operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ specify lower and upper bounds for the limit values of this ratio for behaviors that satisfy I, i.e. for behaviors that performs infinitely many experiments or spend an infinite amount of time at timed vertices.

The reason why the bounds are specified only for behaviors that satisfy I is that, if the behaviors traverse only a finite number of reset edges, or spend a finite amount of time at timed vertices, the long-run average outcome of experiments along them is not well defined. The following example illustrates this point.

Example 5.5 (gambling in the short and long run) In Figure 5.5 we present a TPS Π which corresponds to a gambling system, and an experiment Ψ that measures the average reward of the gambling. Also depicted is the product MDP Π_Ψ. We can see that $s_0 \models \bar{\mathrm{P}}_{\leq 0}\Psi$, since if the gambling is repeated infinitely many times, it has long-run average outcome 0.

On the other hand, the short-term average outcome of the gambling might be different from 0. In particular, let η' be the policy that prescribes to gamble exactly 4 times, and then be idle forever. Clearly,

$$\Pr^{\eta'}_{s_0}(\liminf_{n\to\infty} \mathcal{H}_n(\omega) \geq 4) = \frac{1}{16} \; .$$

Hence, if we dropped the restriction to behaviors satisfying I in (5.12) we would have $s_0 \not\models \bar{\mathrm{P}}_{\leq 0}\Psi$. Dropping this restriction would thus alter drastically the semantics of operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$, preventing their use for the specification of long-run average system behavior. ∎

The following example shows how experiments can be used in conjunction with the operator $\bar{\mathrm{P}}$ to specify the call throughput of system SHARED-RES.

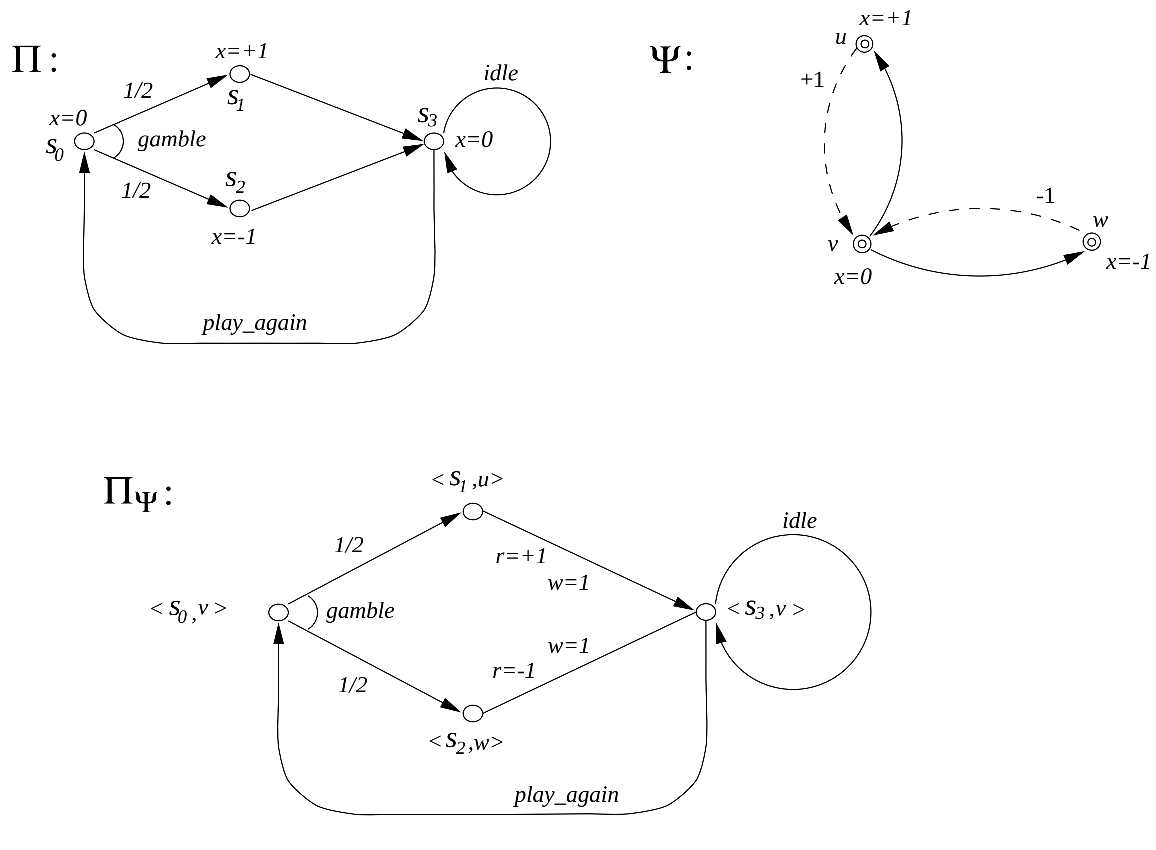

Figure 5.5: A system Π representing a gambling activity, together with an experiment Ψ that measures the long-run average outcome of the gambles, and their product MDP Π_Ψ. System Π has only one variable, x, whose values at the states of Π are depicted in the figure. In the representation of Π_Ψ, we have indicated the values of r and w only for the transitions in which they are not 0.

Example 5.6 (specifying the call throughput of SHARED-RES) Using the experiment Ψ of Figure 5.4, we can specify the call throughput requirement Req1 of system SHARED-RES in the logic GPTL using the formula $\phi_{Req1} : \bar{\mathrm{P}}_{\geq b_0}(\Psi)$. ∎

Threshold outcomes. To be able to talk about the behavior of the system with respect to an experiment under the most favorable and unfavorable policies, it is convenient to define *threshold outcomes.* Threshold outcomes represent the maximum and minimum values of the long-run average outcome of an experiment that can be attained with non-zero probability, and they are defined as follows.

Definition 5.9 (threshold outcomes) Consider a TPS Π, an experiment Ψ and their product MDP Π_Ψ. For each state $\langle s, v\rangle$ of Π_Ψ, we define the *maximum* and *minimum threshold outcomes* $\bar{\mathrm{T}}^+_{\langle s,v\rangle}$ and $\bar{\mathrm{T}}^-_{\langle s,v\rangle}$ by

$$\bar{\mathrm{T}}^+_{\langle s,v\rangle} = \sup\Big\{ a \in \mathbb{R} \;\Big|\; \exists \eta \,.\, \mathrm{Pr}^\eta_{\langle s,v\rangle}\Big(I \wedge \limsup_{n\to\infty} \mathcal{H}_n(\omega) \geq a \Big) > 0 \Big\}$$
$$\bar{\mathrm{T}}^-_{\langle s,v\rangle} = \inf\Big\{ a \in \mathbb{R} \;\Big|\; \exists \eta \,.\, \mathrm{Pr}^\eta_{\langle s,v\rangle}\Big(I \wedge \liminf_{n\to\infty} \mathcal{H}_n(\omega) \leq a \Big) > 0 \Big\} \, ,$$

with the convention that, if $\mathrm{Pr}^\eta_{\langle s,v\rangle}(I) = 0$ for all η, then $\bar{\mathrm{T}}^+_{\langle s,v\rangle} = -\infty$, $\bar{\mathrm{T}}^-_{\langle s,v\rangle} = +\infty$. For a state s of Π, we define then

$$\bar{\mathrm{T}}^+_s(\Psi) = \bar{\mathrm{T}}^+_{\langle s, v_{in}(s)\rangle} \qquad \bar{\mathrm{T}}^-_s(\Psi) = \bar{\mathrm{T}}^-_{\langle s, v_{in}(s)\rangle} \, .$$ ∎

5.3 A Classification of Probabilistic Properties

Now that the presentation of the probabilistic operators P, D, $\bar{\mathrm{P}}$, $\bar{\mathrm{D}}$ is concluded, we return to the topic of the classification of probabilistic properties of systems presented in in Table 1.1. The intuitive difference between single-event and long-run average properties has been mentioned in the introduction.

Single-event properties refer to the probability of a single event, or to the expected value of a random time. These probabilities and expected values are measured on the set of all behaviors. Long-run average properties refer to the long-run average value along a behavior of quantities that are evaluated infinitely often along the behavior, such as the duration or outcome of experiments. Thus, single-event and long-run average properties correspond to the two different type of averages in stochastic systems: *averages over behaviors,* and *averages over time.*

To complete the claim that these classes of properties form indeed a classification, we present an argument that shows that the properties belonging to the different classes are orthogonal, i.e. independent one from the other. We present this argument for the classes of properties expressed by the P and $\bar{\mathrm{P}}$ operators; a simpler argument can also be made for D and $\bar{\mathrm{D}}$.

The model checking of GPTL* formula $\mathrm{P}_{\bowtie a}\phi$ involves the construction of the product between the TPS and the deterministic Rabin automaton for ϕ or $\neg\phi$. Deterministic Rabin automata are strictly more expressive than deterministic Büchi automata (see McNaughton [McN66] and Thomas [Tho90]). Nonetheless, for the sake of simplicity we consider an algorithm that computes the product with a deterministic Büchi automaton instead. Such an algorithm can be used for the subclass of formulas that can be encoded as deterministic Büchi automata.

In the resulting product structure, the question $s \models \mathrm{P}_{\bowtie a}\phi$ is equivalent to the question:

> *From s, under any policy, does a behavior visit infinitely often accepting states with probability $\bowtie a$?*

Consider now the specification $\bar{\mathrm{P}}_{\bowtie a}(\Psi)$. In the structure resulting from the product between the system and the experiment, the question $s \models \bar{\mathrm{P}}_{\bowtie a}(\Psi)$ is equivalent to asking whether the following is true with probability 1:

> *If a behavior traverses infinitely often reset edges, the long-run average outcome is $\bowtie a$.*

From these considerations, we see that there is a correspondence between deterministic Büchi automata and experiments: specifically, to the accepting states of Büchi automata correspond the reset edges of experiments. The two verification questions asked by operators P and $\bar{\mathrm{P}}$, however, are different. For the operator P, the question is essentially:

> *What is the probability of visiting the accepting states infinitely often?*

For the operator $\bar{\mathrm{P}}$, note that the long-run average outcome depends on the relative frequency with which we visit the various accepting states. The corresponding question is then:

> *If we visit the accepting edges infinitely often, with what relative frequency do we visit them?*

The comparison between the two questions asked by operators P and $\bar{\mathrm{P}}$ gives an indication of the independence of the single-event and long-run average classes of properties.

A Comparison with Temporal-Logic Classifications

The classification of probabilistic properties described above might recall the classification of temporal properties into the *safety* and *liveness* classes proposed by Lamport [Lam77, Lam83], or the *safety-progress* classification of Chang, Manna and Pnueli [CMP92]. There are, nonetheless, significant differences between these classifications.

Safety-progress classification. The safety-progress classification is closed under boolean operators, in the sense that boolean combinations of formulas belonging to the classification still belong to the classification. This is not the case for the single-event/long-run average classification. For example, given a sequence formula ϕ and an experiment Ψ, the GPTL* formula

$$\mathrm{P}_{\geq a}\phi \vee \bar{\mathrm{D}}_{>b}\Psi \tag{5.13}$$

does not belong to either of the classes of properties.

Safety-liveness classification. A fundamental result about the safety-liveness classification is that every temporal property can be written as the intersection between a safety and a liveness property. This result has also an interpretation in terms of open and closed sets of behaviors. Temporal formulas can express a thorough "mix" of safety and liveness properties.

While formulas such as (5.13) also realize a mix, the mix is somewhat less thorough. For example, it is possible that the two policies that minimize, respectively, the probability of ϕ and the long-run average duration of Ψ in (5.13) are different. It would be somehow more satisfactory to have the means of combining ϕ with Ψ, and asking a single question about the combined object.

It is an interesting open question whether there are probabilistic specification languages that can express a fuller spectrum of properties, including these thorough mixes of single-event and long-run average properties, while retaining model-checking algorithms of polynomial complexity in the size of the system.

5.4 Expressive Power of Experiments

To be useful for the formal specification of systems, the structure of an experiment must satisfy two requirements:

- Experiments must be capable to encode interesting portions of system behaviors, and to specify durations and outcomes of interest.
- The synchronous composition between an experiment and a finite TPS must be a finite MDP.

Our Definition 5.1 proposes one of the simplest structures satisfying these two requirements. Indeed, many alternative choices are possible, and some of these choices extend the expressive power of experiments in useful ways. We mention two of these choices here: labeling the experiment edges, and adding auxiliary variables.

5.4.1 Labeling the Edges

It is possible to define an extended type of experiments in which edges are labeled by formulas in $\mathit{Form}(\mathcal{V}, \mathcal{V}')$. The edge can be followed only when the TPS takes a transition that satisfies the

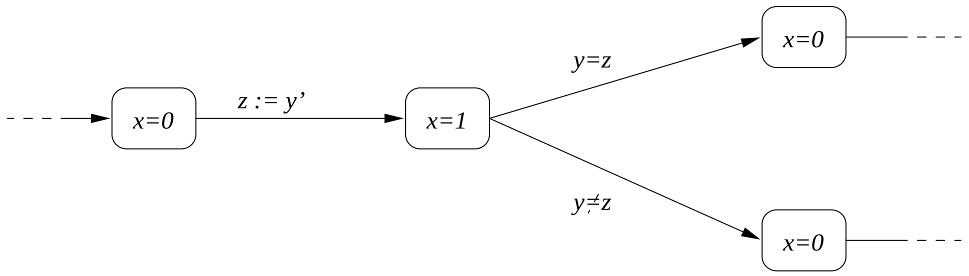

Figure 5.6: A portion of experiment that uses auxiliary variables.

formula labeling the edge. This edge labeling can be used either as an alternative, or in addition to the vertex labeling.

Labeling the edges of an experiment increases its expressive power with respect to infinite-state TPSs, and can yield more compact experiments (with fewer vertices) when composed with finite TPSs. For example, assume the variable edge has domain $\{1, \ldots, N\}$, or $\{1, 2, \ldots\}$. If we label an edge with the formula $x' = x + 1$, the edge will be traversed when the value of variable x increases by 1. An equivalent experiment that used only vertex labels would need to be of size comparable to the domain of x, and it would have to be an infinite structure if the domain were infinite. Thus, we see that experiments with edge labels are clearly desirable for the specification of infinite-state probabilistic systems.

5.4.2 Adding Auxiliary Variables

Another way to extend the expressive power of experiments is augment them with auxiliary variables. These variables can be assigned a value along system edges, and they can appear in edge or vertex labels. This use of auxiliary variables is illustrated by the following example.

Example 5.7 (auxiliary variables in experiments) Consider the portion of experiment depicted in Figure 5.6. In this example, variables x and y are shared with the TPS to which the experiment refers, and variable z is an auxiliary variable of the experiment. This portion of experiment can be used to check whether the value of y is the same before and after the interval in which $x = 1$. ∎

Chapter 6

Verification of Long-Run Average Properties

This chapter presents the model-checking algorithms for operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ in the logics GPTL and GPTL*, along with the correctness proof for the algorithms. We present two algorithms: one that can be used on TPSs that correspond to Markov chains, and another that can be used on general TPSs that include nondeterminism. The model-checking algorithm for TPSs that correspond to Markov chains is an application of standard results in Markov chain theory.

The model-checking algorithm for general TPSs is based on a four-step reduction to linear programming. Given the product of a TPS and an experiment, we first decompose the product in its maximal end components. Second, we apply to each of these end components a transformation that ensures that the experiment is performed infinitely often with probability 1; the transformation does not affect the long-run average outcome of the experiment. Third, we reduce the problem of computing the limit points of $\mathcal{H}_n(\omega)$ to an optimization problem on semi-Markov decision processes. Finally, we show how the optimization problem can be solved by a reduction to linear programming.

The solution of the optimization problem is related to the computation of the average cost of a semi-Markov decision process. Even though technical difficulties prevent a direct use of results for this latter problem in our model-checking algorithm, the reduction provides us with information on the relation between different optimization problems for semi-Markov decision processes. The chapter is concluded with the proof of the equivalence of two optimization criteria for semi-Markov decision processes, and with an algorithm for computing the minimum average cost of semi-Markov processes under a very natural cost structure. These results close a long-standing open problem [Ros70b, Ros70a].

6.1 Model-Checking Algorithms for GPTL and GPTL*

In this section, we present algorithms to decide whether a TPS satisfies a specification written in GPTL or GPTL*. Since these logics are obtained by extending the logics pTL and pTL* [dA97], we need to examine only the cases corresponding to the two new operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$. The justification for the algorithms is somewhat involved, and will be presented in the remainder of the chapter.

Consider an MDP $\Pi = \Pi_0 \otimes \Psi = (S, A, p, r, w)$ resulting from the synchronous product of a TPS Π_0 with experiment Ψ. The model-checking problem consists in determining the truth values of the formulas $\bar{\mathrm{P}}_{\bowtie a}(\Psi)$ and $\bar{\mathrm{D}}_{\bowtie a}(\Psi)$ at all $s \in S$.

We present two model-checking algorithms, depending on whether the TPS Π_0 (and thus the product Π) is purely probabilistic, or whether it contains nondeterminism. In both cases, the first step of the algorithm consists in computing two new labelings R and W for Π_Ψ, to be used in place of r, w. These labelings are defined by

$$W(s,a) = \sum_{t \in S} p_{st}(a)\, w(s,t) \qquad\qquad R(s,a) = \sum_{t \in S} p_{st}(a)\, r(s,a,t) \ , \tag{6.1}$$

and denote by $\Pi' = (S, A, p, W, R)$ the resulting MDP.

6.1.1 Model Checking Purely Probabilistic Systems

The MDP $\Pi = (S, A, p, R, W)$ corresponds to a Markov chain if $|A(s)| = 1$ for all $s \in S$. In this case, we can write p_{st} instead of $p_{st}(a)$, so that the transition matrix is simply $P = [p_{st}]_{s,t \in S}$. Since the labels R and W do not depend on the action, we can consider them as column vectors $\boldsymbol{R} = [R(s)]_{s \in S}$ and $\boldsymbol{W} = [W(s)]_{s \in S}$. The *limiting matrix* $P^* = [p^*_{st}]_{s,t \in S}$ is defined by

$$P^* = \lim_{n \to \infty} \frac{1}{n} \sum_{k=0}^{n-1} P^n \ . \tag{6.2}$$

Element p^*_{st} of this matrix indicates the long-run average fraction of time spent in t when starting at state s. Our first result provides a model-checking algorithm for the case in which Π is an ergodic Markov chain (recall that an ergodic Markov chain is a chain in which all states belong to a single closed recurrent class [KSK66]).

Theorem 6.1 (model checking ergodic Markov chains) *Consider an MDP Π corresponding to an ergodic Markov chain. If $R(s) = W(s) = 0$ for all $s \in S$, then $\Xi_{\bowtie a}(\Psi)$ holds regardless of $\Xi \in \{\bar{\mathrm{P}}, \bar{\mathrm{D}}\}$, $\bowtie \in \{<, \leq, \geq, >\}$ and $a \in \mathbb{R}$. Otherwise, for any state s it is*

$$\liminf_{n \to \infty} \mathcal{H}_n(\omega) = \limsup_{n \to \infty} \mathcal{H}_n(\omega) = \frac{\boldsymbol{\pi R}}{\boldsymbol{\pi W}}$$

with probability 1, where $\boldsymbol{\pi} = [\pi_s]_{s \in S}$ *is the row vector for the steady-state distribution, defined by* $\pi_s = p^*_{ts}$*, for arbitrary* $t \in S$*. Consequently,*

$$s \models \Xi_{\bowtie a}(\Psi) \quad \textit{iff} \quad \frac{\boldsymbol{\pi} \boldsymbol{R}}{\boldsymbol{\pi} \boldsymbol{W}} \bowtie a \ .$$

Proof. We can write

$$\mathcal{H}_n(\omega) = \frac{\displaystyle\sum_{k=0}^{n-1} r(X_k, Y_k, X_{k+1})}{\displaystyle\sum_{k=0}^{n-1} w(X_k, X_{k+1})} = \frac{\displaystyle\frac{1}{n}\sum_{k=0}^{n-1} r(X_k, Y_k, X_{k+1})}{\displaystyle\frac{1}{n}\sum_{k=0}^{n-1} w(X_k, X_{k+1})} \ .$$

Since the chain is ergodic we know that the quantity

$$\frac{1}{n}\sum_{k=0}^{n-1} r(X_k, Y_k, X_{k+1})$$

as $n \to \infty$ converges in distribution to $\sum_{s,t \in S} \pi_s p_{st} r(s,t) = \sum_{s \in S} \pi_s R(s) = \boldsymbol{\pi} \boldsymbol{R}$ [Wil91, KT75]. A similar reasoning, applied to the denominator, concludes the proof. ∎

In a general Markov chain, all behaviors eventually enter with probability 1 a closed recurrent class. Such a closed recurrent class, considered in isolation, is an ergodic chain. Combining this observation with the previous theorem, we immediately get the following result, which yields a model-checking algorithm for general Markov chains.

Corollary 6.1 (model-checking general Markov chains) *If the product MDP* Π *corresponds to a Markov chain, let* $C_1, \ldots, C_n$ *be the closed recurrent classes of the chain, and let* $\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_n$ *be their steady-state distributions. Let also* $\boldsymbol{R}_1$*,* $\boldsymbol{W}_1$*, …,* $\boldsymbol{R}_n$*,* $\boldsymbol{W}_n$ *be the restrictions of the vectors* $\boldsymbol{R}$ *and* $\boldsymbol{W}$ *to classes* C_1*, …,* C_n*. Define*

$$\mathcal{L} = \Big\{ i \in [1..n] \ \Big| \ \exists s \in C_i \,.\, R(s) > 0 \vee W(s) > 0 \Big\} \ ;$$

it is easy to see that $\mathcal{L}$ *is the subset of the closed recurrent classes in which I holds with probability 1. It is*

$$s \models \Xi_{\bowtie a}(\Psi) \quad \textit{iff} \quad \bigwedge_{i \in \mathcal{L}} \frac{\boldsymbol{\pi}_i \boldsymbol{R}_i}{\boldsymbol{\pi}_i \boldsymbol{W}_i} \bowtie a \ ,$$

where Ξ *stands for either* $\bar{\mathrm{P}}$ *or* $\bar{\mathrm{D}}$*.*

6.1.2 Model Checking General Systems

If the TPS Π_0, and consequently the products Π and Π', contain nondeterminism, we propose a model-checking algorithm consisting of the following steps.

1. Compute the list $\{(S_1, A_1), \ldots, (S_n, A_n)\} = \mathit{maxEC}(S)$ of maximal end components of Π'. For each end component (S_i, A_i), $1 \le i \le n$, construct an MDP $\Pi_i = (S_i, A_i, p^i, R_i, W_i)$, where p^i, R_i, W_i are the restrictions of p, R, W to (S_i, A_i).

2. Compute the list
$$\mathcal{L} = \Big\{ i \in [1..n] \;\Big|\; \exists s \in S_i \,.\, \exists a \in A_i(s) \,.\, [R_i(s,a) > 0 \vee W_i(s,a) > 0] \Big\}$$
of sub-MDPs containing at least one state-action pair with positive R or W label.

3. Transform each $\Pi_i = (S_i, A_i, p^i, R_i, W_i)$, $i \in \mathcal{L}$, into a MDP $\widetilde{\Pi}_i = (\widetilde{S}_i, \widetilde{A}_i, \widetilde{p}^i, \widetilde{R}_i, \widetilde{W}_i)$ in which the predicate I holds with probability 1 over a behavior, for any state and for any policy. The transformation does not change the state space of Π_i, and it does not affect the long-run average outcome of experiments.

4. Compute the sets $K^- \subseteq \mathcal{L}$ and $K^+ \subseteq \mathcal{L}$ of end components where the minimum or maximum long-run average outcomes do not diverge to $-\infty$ or $+\infty$.

5. For each $i \in K^+$ (resp. $i \in K^-$), compute λ_i^+ (resp. λ_i^-).

6. Given a state s of Π, let $M_s = \{i \in \mathcal{L} \mid S_i \textit{ reachable in } \Pi' \textit{ from } s\}$ be the sub-MDPs reachable from s in Π. Then,
$$\bowtie \in \{\ge, >\}: \qquad s \models \Xi_{\bowtie a}(\Psi) \;\; \textit{iff} \;\; \bigwedge_{i \in M_s \cap K^-} \lambda_i^- \bowtie a$$
$$\bowtie \in \{\le, <\}: \qquad s \models \Xi_{\bowtie a}(\Psi) \;\; \textit{iff} \;\; (M_s \subseteq \mathcal{L}) \wedge \bigwedge_{i \in M_s} \lambda_i^+ \bowtie a \,,$$
where Ξ is one of $\bar{\mathrm{P}}$, $\bar{\mathrm{D}}$.

We now describe in detail steps 3, 4, and 5.

Step 3: Transforming the MDPs

For each $\Pi_i = (S_i, A_i, p^i, W^i, R^i)$, $1 \le i \le n$, we compute an MDP $\widetilde{\Pi}_i = (\widetilde{S}_i, \widetilde{A}_i, \widetilde{p}^i, \widetilde{R}^i, \widetilde{W}^i)$ in which predicate I holds with probability 1 on all behaviors. For each $1 \le i \le n$, $\widetilde{\Pi}_i$ is obtained from Π_i using the following algorithm.

Algorithm 6.1 (I-transformation)

Input: MDP $\Pi = (S, A, p, R, W)$.

Output: MDP $\widetilde{\Pi} = (\widetilde{S}, \widetilde{A}, \widetilde{p}, \widetilde{R}, \widetilde{W})$.

Method: For each $s \in S$, let $D(s) = \{a \in A(s) \mid R(s,a) = 0 \wedge W(s,a) = 0\}$, and let $\{(B_1, D_1), \ldots, (B_n, D_n)\} = maxEC(S, D)$. Define

$$\widetilde{S} = S \cup \{\widetilde{s}_1, \ldots, \widetilde{s}_n\} - \bigcup_{i=1}^{n} B_i \; .$$

The action sets are defined by:

$$\begin{aligned} &s \in S - \bigcup_{i=1}^{n} B_i : && \widetilde{A}(s) = \{\langle s, a\rangle \mid a \in A(s)\} \\ &1 \le i \le n : && \widetilde{A}(\widetilde{s}_i) = \Big\{\langle s, a\rangle \;\Big|\; s \in B_i \wedge a \in A(s) - D(s)\Big\} \; . \end{aligned}$$

For $s \in \widetilde{S}$, $t \in S - \bigcup_{i=1}^{n} B_i$ and $\langle u, a\rangle \in \widetilde{A}(s)$, the transition probabilities and R, W labelings are defined by

$$\begin{aligned} &\widetilde{p}_{st}(\langle u, a\rangle) = p_{ut}(a) && \widetilde{p}_{s,\widetilde{s}_i}(\langle u, a\rangle) = \sum_{t \in B_i} p_{ut}(a) \\ &\widetilde{R}(s, \langle u, a\rangle) = R(u, a) && \widetilde{W}(s, \langle u, a\rangle) = W(u, a) \; . \end{aligned}$$ ∎

Step 4: Convergence of Long-Run Average Outcomes

The sets K^- and K^+ are computed as follows. If Ψ is a P-experiment, then $K^- = K^+ = \mathcal{L}$. If Ψ is a D-experiment, we use the following algorithm.

Algorithm 6.2 (computation of convergent components, D-experiment)

Input: The list $\mathcal{L}$ of MDPs.

Output: Sets K^- and K^+.

Method: Initially, set $K^- = K^+ = \mathcal{L}$. For each $i \in \mathcal{L}$, do the following steps:

- For $s \in S_i$, let $B(s) = \{a \in \widehat{A}_i(s) \mid \widetilde{W}_i(s,a) = 0\}$ be the set of actions having $\widetilde{W}_i = 0$. Let $(C_1, D_1), \ldots, (C_m, D_m)$ be the maximal end components in $(\widetilde{S}_i, B)$. If there are $j \in [1..m]$, $s \in C_j$ and $a \in D_j(s)$ such that $\widetilde{R}_i(s,a) > 0$, then remove i from K^+.
- Remove i from K^- iff both of the following conditions hold:
  - for all $s \in \widetilde{S}_i$ and $a \in \widehat{A}_i(s)$, $\widetilde{W}_i(s,a) = 0$;
  - there are $s \in \widetilde{S}_i$ and $a \in \widehat{A}_i(s)$ such that $\widetilde{R}_i(s,a) > 0$. ∎

Step 5: Computation of λ^+ and λ^- in each End Component

If $i \in K^-$ (resp. $i \in K^+$), the value of λ^- (resp. λ^+) can be computed with the following algorithms.

Algorithm 6.3 (computation of λ^-) If $i \in K^-$, consider the following linear programming problem, with variables λ, $\{h_s\}_{s \in \widetilde{S}_i}$:

Maximize λ subject to

$$h_s \leq \widetilde{R}_i(s,a) - \lambda \widetilde{W}_i(s,a) + \sum_{t \in \widetilde{S}_i} \widetilde{p}^i_{st}(a)\, h_t \qquad s \in \widetilde{S}_i, a \in \widetilde{A}_i(s) .$$

Let λ_i^- be the single optimal solution of the problem. ∎

Algorithm 6.4 (computation of λ^-) If $i \in K^+$, consider the linear programming problem:

Minimize λ subject to

$$h_s \geq \widetilde{R}_i(s,a) - \lambda \widetilde{W}_i(s,a) + \sum_{t \in \widetilde{S}_i} \widetilde{p}^i_{st}(a)\, h_t \qquad s \in \widetilde{S}_i, a \in \widetilde{A}_i(s) .$$

Let λ_i^+ be the single optimal solution of the problem. ∎

This concludes the presentation of the algorithm for the model checking of operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$. The correctness proof of this algorithm is fairly involved, and will be presented in the following sections of this chapter.

6.1.3 Complexity of GPTL and GPTL* Model Checking

The complexity of the model-checking algorithms will be measured with respect to the size of the TPS and the size of the experiment. The size of the TPS is as in Definition 4.2; the size of an experiment is defined as follows.

Definition 6.1 (size of experiments) Given an experiment $\Psi = (V, E, E_r, V_{in}, \lambda)$, we define the size $|\Psi|$ of Ψ by $|\Psi| \stackrel{def}{=} |V|$. ∎

In our measures of complexity, we do not take into account the cost of checking whether $s \models \lambda(v)$ for each TPS state s and experiment vertex v. In fact, as previously stated, we are not interested in the specific details of the languages used to write the experiment labels, which influence this cost. Rather, we are interested in the computational complexity of the portion of algorithm that follows the construction of the product. The size of an experiment influences this complexity by giving rise to larger product MDPs.

The following theorem describes the complexity of the model-checking algorithms for GPTL and GPTL*.

Theorem 6.2 (complexity of GPTL, GPTL* model checking) *Given a TPS Π, the following assertions hold:*

- *Checking whether Π is non-Zeno has polynomial time-complexity in $||\Pi||$.*
- *Model checking a GPTL formula ϕ has time-complexity linear in $|\phi|$ and polynomial in $|\Pi|$.*
- *Model checking a GPTL* formula ϕ has time-complexity doubly exponential in $|\phi|$ and polynomial in $|\Pi|$.*

A more detailed result is as follows.

Theorem 6.3 (complexity of GPTL, GPTL* model checking II) *Assume that the truth of all state subformulas of ϕ has already been evaluated at all states. The following assertions hold:*

- *The model checking of $\mathrm{A}\phi$, $\mathrm{E}\phi$ has the following time-complexity:*

  GPTL: *Polynomial in $||\Pi||$ and independent of ϕ.*

  GPTL*: *Polynomial in $||\Pi||$ and exponential in $|\phi|$.*

- *The model checking of $\mathrm{P}_{\bowtie a}\phi$ has the following time-complexity:*

  GPTL: *polynomial in $|\Pi|$ and independent of $|\phi|$.*

  GPTL*: *polynomial in $|\Pi|$ and doubly exponential in $|\phi|$.*

- *The model checking of $\mathrm{D}_{\bowtie a}\phi$ has time-complexity polynomial in $|\Pi|$ and independent of ϕ.*
- *The model checking of $\bar{\mathrm{P}}_{\bowtie a}\Phi$ for an experiment Ψ has time-complexity polynomial in both $|\Pi|$ and $|\Phi|$.*

Proof. The lower bounds are a consequence of the results of Courcoubetis and Yannakakis [CY90]; the upper bounds are derived from an analysis of the proposed model-checking algorithms. ∎

6.2 Towards the Correctness Proof: Relation Between r, w and R, W (‡)

The rest of this chapter, except the last section, is devoted to the correctness proof of the model-checking algorithm for GPTL and GPTL*, and can be skipped if desired. The last section contains some results on optimization problems for semi-Markov decision processes.

The first step in the correctness proof of the model-checking algorithm for GPTL and GPTL* consists in justifying the use of the labels R and W in place of r and w in the algorithm. To this end, let I be the predicate defined as in Definition 5.8, and define the two predicated I_{rw}, I_{RW} as follows:

$$\begin{aligned}
\omega \models I_{rw} \quad \textit{iff} \quad & \stackrel{\infty}{\exists} k \,.\, [r(X_k, Y_k, X_{k+1}) > 0 \vee w(X_k, X_{k+1}) > 0] \\
\omega \models I_{RW} \quad \textit{iff} \quad & \stackrel{\infty}{\exists} k \,.\, [R(X_k, Y_k) > 0 \vee W(X_k, Y_k) > 0] \,.
\end{aligned}$$

Moreover, define

$$\mathcal{H}_n^{RW}(\omega) = \frac{\displaystyle\sum_{k=0}^{n-1} R(X_k, Y_k)}{\displaystyle\sum_{k=0}^{n-1} W(X_k, Y_k)} \,.$$

To justify the use of the new labelings R, W in the model-checking algorithms, we will show that we can adopt the following alternative definition to (5.12) for the semantics of $\bar{\mathrm{P}}$, $\bar{\mathrm{D}}$.

Definition 6.2 (definition of $\bar{\mathrm{P}}$, $\bar{\mathrm{D}}$ based on W, R) Consider a product MDP $\Pi' = (S, A, p, R, W)$. Then, $s \models \bar{\mathrm{P}}_{\bowtie a}\Psi$ (resp. $s \models \bar{\mathrm{D}}_{\bowtie a}\Psi$) holds if, for all policies η,

$$\mathrm{Pr}^{\eta}_{\langle s,v\rangle}\left(I_{RW} \;\rightarrow\; \liminf_{n\to\infty} \mathcal{H}_n^{RW} \bowtie a\right) = 1 \,. \tag{6.3}$$

The definition of $s \models \bar{\mathrm{P}}_{\bowtie a}\Psi$ (resp. $s \models \bar{\mathrm{D}}_{\bowtie a}\Psi$) for $\bowtie \in \{\leq, <\}$ is analogous. ∎

We show the equivalence of Definitions 5.8 and 6.2 through a sequence of two lemmas.

Lemma 6.1 (almost everywhere equivalence of I, I_{rw}, I_{RW}) *The following statements hold.*

- $\omega \models I$ *iff* $\omega \models I_{rw}$, *i.e. the predicates* I *and* I_{rw} *are equivalent.*
- *For any policy* η *and any* s, $\mathrm{Pr}^{\eta}_{s}(I_{rw} \equiv I_{RW}) = 1$.

Proof. Consider the first statement. If Ψ is a P-experiment, then $r(s,a,t) > 0$ only when $w(s,t) > 0$, since the outcomes are associated only to reset edges. Thus, $\sum_{k=0}^{\infty} w(X_k, X_{k+1})$ diverges exactly when I_{rw} holds. If Ψ is a D-experiment, the result is immediate.

Consider now the second statement. Let (B, C) be any end component of Π, fix any $s \in S$, and let $\Omega_s^{(B,C)} = \{\omega \in \Omega_s \mid \textit{inft}(\omega) = \textit{sa}(B,C)\}$ be the set of behaviors that eventually follow (B, C). We will prove that $\mathrm{Pr}^{\eta}_{s}(I_{rw} \equiv I_{RW} \mid \omega \in \Omega_s^{(B,C)}) = 1$; the result then follows from Theorem 3.2 and from the fact that there are finitely many end components.

Assume first that there is a state-action pair $(s,a) \in \textit{sa}(B,C)$ such that $R(s,a) > 0 \vee W(s,a) > 0$. Clearly, I_{RW} holds for all behaviors in $\Omega_s^{(B,C)}$, since $\omega \in \Omega_s^{(B,C)}$ takes infinitely many times the state-action pair s, a. Each time the state-action pair s, a occurs, the successor state t is chosen independently at random according to the probability $p_{st}(a)$; thus by definition of R, W it is immediate to see that I_{rw} must hold with probability 1 on $\Omega_s^{(B,C)}$.

Conversely, assume that $R(s,a) = W(s,a) = 0$ for all $(s,a) \in sa(B,C)$. Then, I_{RW} does not hold for any behavior in $\Omega_s^{(B,C)}$. For I_{rw}, there are two cases.

- *Ψ is a P-experiment.* Since w is non-negative, from $W(s,a) = 0$ for all $(s,a) \in sa(B,C)$ follows that $w(s,t) = 0$ for all $s \in B$, $t \in \bigcup_{a \in A(s)} \mathrm{Succ}(s,a)$. Since $r \neq 0$ only when $w > 0$, a similar result follows for r. This indicates that I_{rw} does not hold on $\Omega_s^{(B,C)}$.

- *Ψ is a D-experiment.* Reasoning as before, we obtain $w(s,t) = 0$ for all $s \in B$, $t \in \bigcup_{a \in A(s)} \mathrm{Succ}(s,a)$. Since $r \geq 0$, a similar result follows for r. This indicates that I_{rw} does not hold on $\Omega_s^{(B,C)}$. ∎

Lemma 6.2 *Consider a product MDP $\Pi'' = (S, A, p, r, w, R, W)$ with both the r, w labelings and the derived R, W ones. For any state $s \in S$ and for any policy η,*

$$\Pr^{\eta}_{\langle s,v \rangle}\Big(I_{RW} \;\rightarrow\; \liminf_{n\to\infty} \mathcal{H}_n(\omega) = \liminf_{n\to\infty} \mathcal{H}_n^{RW}(\omega)\Big) = 1\,.$$

A similar relation holds for lim sup.

Proof. Once again, we partition the set of behaviors Ω_s into the sets $\Omega_s^{(B,C)} = \{\omega \in \Omega_s \mid \mathit{inft}(\omega) = sa(B,C)\}$ of behaviors that follow end component (B,C), for every end component (B,C), and we prove the result separately for every (B,C).

If $R(s,a) = W(s,a) = 0$ for all state-action pairs $(s,a) \in sa(B,C)$, then I_{RW} does not hold for any behavior of $\Omega_s^{(B,C)}$. Otherwise, assume that there is at least one $(s,a) \in sa(B,C)$ such that $R(s,a) > 0 \vee W(s,a) > 0$. In this case, I_{RW} holds for all behaviors in $\Omega_s^{(B,C)}$. Define the random variables N^n_{sat} and $M^n_{t|s,a}$ by

$$N^n_{sat} = \Big|\{i \leq n \mid X_i = s \wedge Y_i = a \wedge X_{i+1} = t\}\Big| \qquad\qquad M^n_{t|s,a} = \frac{N^n_{sat}}{N^n_{sa}}\,.$$

In words, N^n_{sat} is the number of sequences s, a, t contained in the first n positions of a behavior, and $M^n_{t|s,a}$ is the relative number of times in which state-action pair s, a leads to t in the first n positions of a behavior. The definition of N^n_{sa} is as in (3.4). For $\omega \in \Omega_s^{(B,C)}$ we can write:

$$\liminf_{n\to\infty} \frac{\displaystyle\sum_{k=0}^{n-1} r(X_k, Y_k, X_{k+1})}{\displaystyle\sum_{k=0}^{n-1} w(X_k, X_{k+1})} = \liminf_{n\to\infty} \frac{\displaystyle\sum_{s,t\in S}\sum_{a\in A(s)} N^n_{sat}\, r(s,a,t)}{\displaystyle\sum_{s,t\in S}\sum_{a\in A(s)} N^n_{sat}\, w(s,t)} \tag{6.4}$$

$$= \liminf_{n\to\infty} \frac{\displaystyle\sum_{s,t\in B}\sum_{a\in C(s)} N^n_{sat}\, r(s,a,t)}{\displaystyle\sum_{s,t\in B}\sum_{a\in C(s)} N^n_{sat}\, w(s,t)} \tag{6.5}$$

$$= \liminf_{n\to\infty} \frac{\displaystyle\sum_{s,t\in B}\sum_{a\in C(s)} N_{sa}^n\, M_{t|s,a}^n\, r(s,a,t)}{\displaystyle\sum_{s,t\in B}\sum_{a\in C(s)} N_{sa}^n\, M_{t|s,a}^n\, w(s,t)} . \tag{6.6}$$

The step from (6.4) to (6.5) is justified by the fact that at least one of the numerator or denominator of (6.4) diverges with probability 1, so that the lim inf of the ratio depends with probability 1 on the portion of behavior once the end component (B, C) has been entered. Since $N_{sa}^n \to \infty$ for every $(s, a) \in sa(B, C)$, and since the successor state t of a state-action pair s, a is chosen independently each time according to the probability $p_{st}(a)$, by the strong law of large numbers we know that as $n \to \infty$, $M_{t|s,a}^n \to p_{st}(a)$ in distribution. Therefore, with probability 1 the limit (6.6) is equal to

$$\liminf_{n\to\infty} \frac{\displaystyle\sum_{s,t\in B}\sum_{a\in C(s)} N_{sa}^n\, p_{st}(a)\, r(s,a,t)}{\displaystyle\sum_{s,t\in B}\sum_{a\in C(s)} N_{sa}^n\, p_{st}(a)\, w(s,t)} = \liminf_{n\to\infty} \frac{\displaystyle\sum_{s\in B}\sum_{a\in C(s)} N_{sa}^n\, R(s,a)}{\displaystyle\sum_{s\in B}\sum_{a\in C(s)} N_{sa}^n\, W(s,a)} = \liminf_{n\to\infty} \frac{\displaystyle\sum_{k=0}^{n-1} R(X_k, Y_k)}{\displaystyle\sum_{k=0}^{n-1} W(X_k, Y_k)} ,$$

which concludes the proof. ∎

The following corollary, consequence of the two preceding lemmas, summarizes the equivalence of the two definitions of $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$.

Corollary 6.2 (use of R, W instead of r, w) *Definitions 5.8 and 6.2 are equivalent, in the sense that the sets of formulas that hold at the various states of the product MDP are the same under both definitions.*

Using the result of this corollary, from this point on we base the definition of the logic on R and W, using Definition 6.2. To simplify the notation, we will write I and $\mathcal{H}$ in place of I_{RW} and $\mathcal{H}^{RW}$; no confusion should arise, since the MDPs considered in the following are labeled only by R and W.

6.3 Decomposition in End Components

The intuitive justification of Step 1, which decomposes Π' in maximal end components, is based on the observation that Definition 6.2 for the semantics of the long-run average operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ depends entirely on the portion of behaviors towards infinity. By Theorem 3.2 we know that behaviors with probability 1 eventually follow end components. Thus, only the portion of behaviors in their final end components matters, and we can study separately the behaviors depending on the maximal end components in which they are eventually confined.

Step 2 is then justified by Lemma 6.1. In fact, if all state-action pairs of an end component have $R = W = 0$, then I does not hold for any behavior that follows the component forever, and the end component does not need to be further studied to determine the truth-values of $\bar{\mathrm{P}}$, $\bar{\mathrm{D}}$.

While the above reasoning might be fairly convincing, a rigorous justification of the two steps is somewhat involved. We chose to present the argument in detail, in order to introduce in a relatively simple setting some proof techniques that will be used later in the chapter. To state the theorem, let

$$\mathcal{L} = \Big\{ i \in \mathcal{L} \mid \forall s \in S_i \, . \, s \models_i \Xi_{\bowtie a}(\Psi) \Big\} \tag{6.7}$$

be the set of end components all whose states satisfy the formula $\Xi_{\bowtie a}(\Psi)$ being model checked. In (6.7), the subscript i of $\models_i$ is a reminder that the truth value of $\Xi_{\bowtie a}(\Psi)$ is evaluated on the sub-MDP (S_i, A_i) rather than the complete MDP. Let M_s be defined as in Step 6 of the model-checking algorithm.

Theorem 6.4 (correctness of the Decomposition Step) *For every state $s \in S$,*

$$s \models \Xi_{\bowtie a}(\Psi) \quad \textit{iff} \quad M_s \cap \mathcal{L} \subseteq \mathcal{L} \, .$$

Proof. We will assume that $\bowtie$ is equal to $\geq$; the arguments for the other cases are analogous.

In one direction, assume that there is $i \in M_s \cap \mathcal{L}$ and $t \in S_i$ such that $t \not\models_i \Xi_{\geq a}(\Psi)$. This means that from t there is a policy η for Π_i such that $\Pr_t^{\eta}(I \to \liminf_{n\to\infty} \mathcal{H}_n(\omega) < a) > 0$. Since $i \in M_s$, we can construct a policy η' for s in Π' as follows: η' tries to reach t, and upon reaching t imitates η. Formally, for any finite sequence of states $s_0, \ldots, s_k$ and $a \in A(s_k)$, the conditional probability $Q_{\eta'}(a \mid s_0, \ldots, s_k)$ is defined as follows:

- if $s_j \neq t$ for all $0 \leq j \leq k$, then $Q_{\eta'}(a \mid s_0, \ldots, s_k) = |A(s_k)|^{-1}$;
- if there is $1 \leq j \leq k$ such that $s_j = t$, then $Q_{\eta'}(a \mid s_0, \ldots, s_k) = Q_{\eta}(a \mid s_l, \ldots, s_k)$, where $l = \min\{j \mid s_j = t\}$.

Under policy η', a behavior from s reaches t with positive probability. Since η' simulates η from t onwards, we conclude $\Pr_s^{\eta'}(I \to \liminf_{n\to\infty} \mathcal{H}_n(\omega) < a) > 0$, concluding the argument.

In the other direction, assume that for all $i \in M_s \cap \mathcal{L}$ and $t \in S_i$ it is $t \models_i \Xi_{\bowtie a}(\Psi)$. Let $\alpha = s_0 a_0 s_1 \cdots s_k$ be a finite behavior prefix such that $s_k \in \bigcup_{i \in M_s \cap \mathcal{L}} S_i$. Let j_α be index of the MDP where α ends, i.e. the unique $j \in M_s \cap \mathcal{L}$ such that $s_k \in S_j$. Define the event $\vec{\alpha}$ as follows:

$$X_i = s_i \textit{ for } 0 \leq i \leq k, \; Y_i = a_i \textit{ for } 0 \leq i < k, \textit{ and } (X_i, Y_i) \in sa(S_{j_\alpha}, A_{j_\alpha}) \textit{ for } i \geq k.$$

Intuitively, event $\vec{\alpha}$ states that a behavior first follows α, and then stays forever in the maximal end component in which α ends. It is not difficult to see that event $\vec{\alpha}$ is measurable.

Since the set of all possible prefixes α of the above form is enumerable, to show that $\Pr_s^{\eta}(I \wedge \liminf_{n\to\infty} \mathcal{H}_n(\omega) < a) = 0$ it suffices to show that, for all α,

$$\Pr_s^{\eta}\Big(I \wedge \liminf_{n\to\infty} \mathcal{H}_n(\omega) < a \;\Big|\; \vec{\alpha} \Big) = 0 \, . \tag{6.8}$$

The reason why only the end components in $M_s \cap \mathcal{L}$ have to be considered is that, by Theorem 3.2, a behavior is eventually confined to an end component with probability 1. If the behavior is confined to an end component (S_i, A_i) with $i \in \{1, \ldots, n\} - \mathcal{L}$, then I does not hold on the behavior, so the end components does not need to be considered.

Assume towards the contradiction that

$$\Pr_s^\eta\Big(I \wedge \liminf_{n\to\infty} \mathcal{H}_n(\omega) < a \;\Big|\; \vec{\alpha}\Big) > 0 \tag{6.9}$$

for a specific α, and let $j = j_\alpha$ to simplify the notation. Define the *semi-policy* $\tilde{\eta}$ by

$$Q_{\tilde{\eta}}(a \mid \text{last}(\alpha)\sigma) = Q_\eta(a \mid (\natural\alpha)\sigma) \; ,$$

where σ is a finite sequence of states of S_j, last(α) denotes the last state of α, $\natural\alpha$ denotes the sequence of states obtained by removing the actions from α, and $a \in A_j(\text{last}(\sigma))$. Note that we use the convention of denoting by $\beta_1\beta_2$ the concatenation of two sequences β_1, β_2. We say that this policy is a *semi-policy* since it is possible that, for some $t \in S_j$ and σ,

$$\sum_{a\in A_j(\text{last}(\sigma))} Q_{\tilde{\eta}}(a \mid \text{last}(\alpha)\sigma) < 1 \; . \tag{6.10}$$

This is due to the fact that, after following α, some behaviors might leave the end component (S_j, A_j). We can interpret this "missing probability" as the probability that a behaviors ends prematurely; this is a point of view often adopted in the theory of stochastic processes and Markov chains (see, for example, Kemeny, Snell, and Knapp [KSK66]). With this interpretation, we can associate to $\tilde{\eta}$ a sub-probability space defined on the infinite behaviors. From the correspondence between η and $\tilde{\eta}$ and from (6.9) we have that

$$\Pr_{\text{last}(\alpha)}^{\tilde{\eta}}\Big(I \wedge \liminf_{n\to\infty} \mathcal{H}_n(\omega) < a\Big) = \Pr_s^\eta\Big(I \wedge \liminf_{n\to\infty} \mathcal{H}_n(\omega) < a \;\Big|\; \vec{\alpha}\Big) > 0 \; .$$

We can then re-normalize $\tilde{\eta}$, yielding a policy η' defined by

$$Q_{\eta'}(a \mid \text{last}(\alpha)\sigma) = \frac{Q_{\tilde{\eta}}(a \mid \text{last}(\alpha)\sigma)}{\displaystyle\sum_{a\in A_j(\text{last}(\sigma))} Q_{\tilde{\eta}}(a \mid \text{last}(\alpha)\sigma)} \; ,$$

for all $a \in A_j(\text{last}(\sigma))$ and all finite sequences σ of states of S_j. After this re-normalization step, we have that

$$\Pr_{\text{last}(\alpha)}^{\eta'}\Big(I \wedge \liminf_{n\to\infty} \mathcal{H}_n(\omega) < a\Big) \geq \Pr_{\text{last}(\alpha)}^{\tilde{\eta}}\Big(I \wedge \liminf_{n\to\infty} \mathcal{H}_n(\omega) < a\Big) > 0 \; .$$

This finally yields the desired contradiction, since the existence of such an η' indicates that $\mathrm{last}(\alpha) \not\models_j \Xi_{\geq a}(\Psi)$. ∎

6.4 MDP Transformation

Let Π be a strongly connected MDP. In the remainder of this chapter, instead of considering the MDP $\widetilde{\Pi}$ obtained by applying Algorithm 6.1 to Π, we will study the alternative MDP $\widehat{\Pi}$ obtained by applying the following algorithm.

Algorithm 6.5 (I-transformation, version II)

Input: MDP $\Pi = (S, A, p, R, W)$.

Output: MDP $\widetilde{\Pi} = (S, \widetilde{A}, \widetilde{p}, \widetilde{R}, \widetilde{W})$.

Method: For each $s \in S$, let $D(s) = \{a \in A(s) \mid R(s,a) = 0 \wedge W(s,a) = 0\}$, and let $\{(B_1, D_1), \ldots, (B_n, D_n)\} = maxEC(S, D)$. Define

$$\widetilde{A}(s) = \begin{cases} \{\langle s, a\rangle \mid a \in A(s)\} & \text{if } s \notin \bigcup_{i=1}^n B_i; \\ \{\langle t, a\rangle \mid t \in B_i \wedge a \in A(t) - D_i(t)\} & \text{if } s \in B_i,\ 1 \leq i \leq n. \end{cases} \tag{6.11}$$

and

$$\widetilde{p}_{su}(\langle t, a\rangle) = p_{tu}(a) \qquad \widetilde{R}(s, \langle t, a\rangle) = R(t,a) \qquad \widetilde{W}(s, \langle t, a\rangle) = W(t,a) \tag{6.12}$$

for all $s, u \in S$ and $\langle t, a\rangle \in \widetilde{A}(s)$. ∎

The rationale for this substitution is that it is easier for technical reasons to reason about the correspondence between Π and $\widehat{\Pi}$ than about the one between Π and $\widetilde{\Pi}$. The following theorem justifies the study of the new algorithm.

Theorem 6.5 (alternative algorithm for I-transformation) *In Section 6.1.2, the quantities computed in Steps 4, 5 and 6 do not change, if Algorithm 6.5 is used instead of Algorithm 6.1 in Step 3.*

Proof. Consider two states $s, t \in S$ of the MDP $\widehat{\Pi}$. By construction, if $s, t \in B_i$ for some $1 \leq i \leq n$, then $\widehat{A}(s) = \widehat{A}(t)$, and for any other state $u \in S$ and $a \in \widehat{A}(s)$ it is

$$\widehat{p}_{su}(a) = \widehat{p}_{tu}(a) \qquad \widehat{R}(s,a) = \widehat{R}(t,a) \qquad \widehat{W}(s,a) = \widehat{W}(t,a)\ .$$

This indicates that s, t are in fact identical copies. The MDP $\widetilde{\Pi}$ is obtained from $\widehat{\Pi}$ by merging all the identical copies corresponding to B_i into a single representative state $\widetilde{s}_i$, letting

$$\widetilde{p}_{u,\widetilde{s}_i}(a) = \sum_{s \in B_i} \widehat{p}_{us}(a)$$

for every $a \in \widetilde{A}(s)$. From these considerations, we see that neither the graph-theoretic conditions checked in Step 4 nor the solution of the linear programming problem of Step 5 are affected by the substitution of $\widetilde{\Pi}$ with $\widehat{\Pi}$. ∎

Showing the correctness of Algorithm 6.5. Let $B = \bigcup_{i=1}^{n} B_i$, $D(s) = \bigcup_{i=1}^{n} D_i(s)$. For any behavior ω, we define $\mathcal{H}_\bullet(\omega) \subseteq \mathbb{R}$ to be the set of accumulation points of the sequence $\mathcal{H}_n(\omega)$ for $n \to \infty$. Of course,

$$\liminf_{n\to\infty} \mathcal{H}_n(\omega) = \min \mathcal{H}_\bullet(\omega) \qquad\qquad \limsup_{n\to\infty} \mathcal{H}_n(\omega) = \max \mathcal{H}_\bullet(\omega) \ .$$

To show the correctness of the algorithm, we will construct a relationship between the behaviors and policies of Π and the behaviors and policies of $\widehat{\Pi}$. The aim, ultimately, is to relate the probability distribution of $\mathcal{H}_\bullet(\omega)$ in Π and $\widehat{\Pi}$. We begin by relating the behaviors first.

6.4.1 Relating the Behaviors of Π and $\widehat{\Pi}$

First, observe that a behavior of $\widehat{\Pi}$ satisfies I with probability 1. This is stated in the following lemma.

Lemma 6.3 *For any $s \in S$ and any policy $\widehat{\eta}$ for $\widehat{\Pi}$, $\Pr_s^{\widehat{\eta}}(I) = 1$. Thus, for any $s \in S$ and $\widehat{\eta}$ at least one of the expectations*

$$\mathrm{E}_s^{\widehat{\eta}}\Big\{\sum_{k=0}^{n-1} \widehat{R}(X_k, Y_k)\Big\} \qquad\qquad \mathrm{E}_s^{\widehat{\eta}}\Big\{\sum_{k=0}^{n-1} \widehat{W}(X_k, Y_k)\Big\}$$

diverges as $n \to \infty$.

Proof. The first statement follows from the fact that $\widehat{\Pi}$, by construction, does not contain any end component consisting entirely of state-action pairs having $\widehat{R} = \widehat{W} = 0$. The result then follows from the fact that, with probability 1, the set of state-action pairs repeated infinitely often along a behavior is an end component (Theorem 3.2). The second statement is an immediate consequence of the first. ∎

On the other hand, the probability that a behavior of Π satisfies I is not necessarily 1. The purpose of the transformation effected by Algorithm 6.5 is precisely to ensure that I holds with probability 1. The MDPs having this property are said to be *proper.*

Definition 6.3 (proper MDP) An MDP is *proper* if Lemma 6.3 holds for it. ∎

We now construct a relation between the behaviors of Π and those of $\widehat{\Pi}$. The relation is constructed in such a way that, if ω of Π and $\widehat{\omega}$ of $\widehat{\Pi}$ are related, written $\omega \sim \widehat{\omega}$, then $\mathcal{H}_\bullet(\omega) = \widehat{\mathcal{H}}_\bullet(\widehat{\omega})$, where $\widehat{\mathcal{H}}_\bullet$ is computed using the labelings $\widehat{R}$, $\widehat{W}$. Since I holds on $\widehat{\Pi}$ with probability 1, but I may hold on $\widehat{\Pi}$ with probability smaller than 1 for some initial states and policies, some behaviors of Π that do not satisfy I will not be related to any behavior of $\widehat{\Pi}$. On the other hand, all behaviors of $\widehat{\Pi}$ will be related to some behavior of Π.

The intuition behind the definition of the relation is that a behavior $\omega \in \Omega_s$ of Π is related to $\widehat{\omega} \in \widehat{\Omega}_s$ of $\widehat{\Pi}$ if ω and $\widehat{\omega}$ coincide outside of B, and if whenever ω enters B at s and leaves it afterwards by state-action pair t, a, behavior $\widehat{\omega}$ at s takes the action $\langle t, a\rangle$.

Given two behaviors, ω for Π and $\widehat{\omega}$ for $\widehat{\Pi}$, we say that $\omega \sim \widehat{\omega}$ if ω and $\widehat{\omega}$ can be written as infinite sequences of the form:

$$\begin{array}{rcllll} \widehat{\omega} & = & \sigma_0 s_0\langle t_0, a_0\rangle & & \sigma_1 s_1 \langle t_1, a_1\rangle, & \dots \\ \omega & = & \underbrace{\sigma_0}_{\notin B} \underbrace{s_0 \alpha_0 t_0 a_0}_{\in B} & & \underbrace{\sigma_1}_{\notin B} \underbrace{s_1 \alpha_1 t_1 a_1}_{\in B} & \dots \end{array} \tag{6.13}$$

where, for all $i \geq 0$:

- $s_i \in B$.
- $\{\sigma_i\}_{i\in\mathbb{N}}$ denotes a possibly empty sequence $s_0, a_0, \dots, s_m, a_m$ with $m \geq 0$ and $s_k \notin B$ for $0 \leq k \leq m$.[1]
- $\{\alpha_i\}_{i\in\mathbb{N}}$ denotes either the empty sequence ϵ, or a sequence $a_0, s_1, a_1, \dots, s_m, a_m$ with $m > 0$ and $s_k \in B$, $a_k \in D(s_k)$, $1 \leq k \leq m$. If $\alpha = \epsilon$, we consider a sequence $s\alpha t$ to be an abbreviation for the single state s, and we require that $s = t$ holds.

6.4.2 Constructing Corresponding Policies

We now show how to construct a policy $\widehat{\eta}$ for $\widehat{\Pi}$ given a policy η for Π. The idea is that $\widehat{\eta}$ simulates η outside of B, and "shortcuts" the portions of behaviors done inside B under η. The definition relies on an infinite set of clauses $\{\mathcal{A}_i\}_{i\geq 0}$: clause $\mathcal{A}_i$ gives $\widetilde{Q}_{\widehat{\eta}}(a \mid \beta)$ for the case in which β contains i sequences in B. We write $\widetilde{Q}_{\widehat{\eta}}$ instead of $Q_{\widehat{\eta}}$ since $\widetilde{Q}_{\widehat{\eta}}$ is not a proper conditional probability, as $\sum_a \widetilde{Q}_{\widehat{\eta}}(a \mid \beta)$ can be smaller than 1 in some cases; we will discuss this point later on.

We first list $\mathcal{A}_0$ and $\mathcal{A}_1$, to give the idea behind the construction of $\widehat{\eta}$. In the following clauses, $u_0, u_1, u_2 \notin B$.

- $\mathcal{A}_0$:
$$\widetilde{Q}_{\widehat{\eta}}(a \mid \sigma_0 u_0) = Q_\eta(a \mid \sigma_0 u_0) \tag{6.14}$$

[1] Of course, different σ_i may refer to different sequences, so that we should define more properly σ_i : $s_0^i, a_0^i, \dots, s_m^i, a_{m_i}^i$. We have retained the simpler notation for the sake of readability, hoping that no confusion will arise.

- $\mathcal{A}_1$:

$$\widetilde{Q}_{\widehat{\eta}}(\langle t, a\rangle \mid \sigma_1 s_1) = \frac{\sum_{\alpha_1} \Pr^{\eta}(\sigma_1 s_1 \alpha_1 t\, a)}{\Pr^{\eta}(\sigma_1 s_1)} \tag{6.15}$$

$$\widetilde{Q}_{\widehat{\eta}}(a \mid \sigma_1 s_1 \langle t_1, a_1\rangle \sigma_2 u_2) = \frac{\sum_{\alpha_1} \Pr^{\eta}(\sigma_1 s_1\ \alpha_1 t_1 a_1\ \sigma_2 u_2 a)}{\sum_{\alpha_1} \Pr^{\eta}(\sigma_1 s_1\ \alpha_1 t_1 a_1\ \sigma_2 u_2)} \tag{6.16}$$

To define $\mathcal{A}_i$ for $i \geq 0$, we introduce the abbreviations

$$\beta_i := \bigodot_{k=1}^{i} (\sigma_k s_k \alpha_k t_k a_k) \qquad \widehat{\beta}_i := \bigodot_{k=1}^{i} (\sigma_k s_k \langle t_k, a_k\rangle)$$

for two related behavior segments of Π, $\widehat{\Pi}$, where $\odot$ is the sequence concatenation operator. Clause $\mathcal{A}_i$, for $i \geq 0$, is then given by:

$$\widetilde{Q}_{\widehat{\eta}}(\langle t_i, a_i\rangle \mid \widehat{\beta}_{i-1} \sigma_i s_i) = \frac{\displaystyle\sum_{\alpha_1 \cdots \alpha_i} \Pr^{\eta}(\beta_{i-1} \sigma_i s_i \alpha_i t_i a_i)}{\displaystyle\sum_{\alpha_1 \cdots \alpha_{i-1}} \Pr^{\eta}(\beta_{i-1} \sigma_i s_i)} \tag{6.17}$$

$$\widetilde{Q}_{\widehat{\eta}}(a \mid \widehat{\beta}_i \sigma_{i+1} u_{i+1}) = \frac{\displaystyle\sum_{\alpha_1 \cdots \alpha_i} \Pr^{\eta}(\beta_i \sigma_{i+1} u_{i+1} a)}{\displaystyle\sum_{\alpha_1 \cdots \alpha_i} \Pr^{\eta}(\beta_i \sigma_{i+1} u_{i+1})} \tag{6.18}$$

where $u_{i+1} \notin B$. We note that the summations in the above equations refer also to the occurrences of α-sequences within the abbreviations β_i, $\widehat{\beta}_i$.

Since the sequences appearing in the above clauses consist of both states and actions, they seem to define, rather than policies, *extended policies* that choose the next action with a probability that depends not only on the past sequence of states, but also on the past sequence of actions. To show that this is not the case, we will prove that the probability with which the next action is chosen does not depend on the past sequence of actions.

We prove independence for a clause of type (6.17), for $i > 0$; the reasoning for (6.18) and (6.14) is similar (in fact, simpler).

There are two cases. First, consider the occurrences of actions contained in α-sequences of $\beta_{i-1}\sigma_i s_i \alpha_i t_i a_i$ or $\beta_{i-1}\sigma_i s_i$. Then, since we sum over all α-sequences, the ratio (6.17) does not depend on these actions.

Second, consider an occurrence of action in $\beta_{i-1}\sigma_i s_i$ that is not part of an α-sequence. Denote by $(\beta_{i-1}\sigma_i s_i)[a]$ the result of replacing that occurrence with action a. For any two actions a, b, it is easy to see that

$$\frac{\sum_{\alpha_1\cdots\alpha_i} \Pr^{\eta}\Big((\beta_{i-1}\sigma_i s_i)[a]\alpha_i t_i a_i\Big)}{\sum_{\alpha_1\cdots\alpha_{i-1}} \Pr^{\eta}(\beta_{i-1}\sigma_i s_i)[a]} = \frac{\sum_{\alpha_1\cdots\alpha_i} \Pr^{\eta}\Big((\beta_{i-1}\sigma_i s_i)[b]\alpha_i t_i a_i\Big)}{\sum_{\alpha_1\cdots\alpha_{i-1}} \Pr^{\eta}(\beta_{i-1}\sigma_i s_i)[b]} ,$$

showing that definition (6.17) does not depend on the sequence of actions in it.

The policy $\widehat{\eta}$ defined by these clauses is a *semi-policy,* since for $s \in B$ and a generic past β it is possible that

$$\sum_{\langle t,a\rangle \in \widehat{A}(s)} \widetilde{Q}_{\widehat{\eta}}(\langle t,a\rangle \mid \beta s) < 1 \ . \tag{6.19}$$

In fact, if a behavior in Π has a non-zero probability of following forever (B, D) once entered, the summation on the right hand side of (6.15) will be smaller than 1. We can interpret the "missing probability" in (6.19) as the probability that the behavior in $\widehat{\Pi}$ stops once reached s. The MDP $\widehat{\Pi}$, under semi-policy $\widehat{\eta}$, is thus a sub-stochastic process.

6.4.3 Model Checking on Π and $\widehat{\Pi}$

Using the relationship developed between Π and $\widehat{\Pi}$, we can finally relate the model-checking problem on these two MDPs. First, we show that the counterexamples to the specification are preserved by the transformation from Π to $\widehat{\Pi}$.

To prove this result, we formalize our relation between behaviors in the two MDPs. Given a behavior prefix

$$\rho = \widehat{\beta}_i \sigma_i u_i \tag{6.20}$$

of $\widehat{\Pi}$ for $i \geq 0$ and $u_i \in S$, define the set of related behaviors $\mu(\rho)$ of Π by

$$\mu(\rho) = \bigcup_{\alpha_1\cdots\alpha_i} \beta_i \sigma_i u_i \ .$$

The following lemma relates the probability measures of ρ and $\mu(\rho)$.

Lemma 6.4 *For every behavior prefix ρ of $\widehat{\Pi}$, it is*

$$\Pr^{\widehat{\eta}}(\rho) = \Pr^{\eta}(\mu(\rho)) \ . \tag{6.21}$$

Proof. The result is proved by induction on i in (6.20). The base case follows immediately from (6.14). To prove the induction case, we generalize μ to encompass behavior prefixes that end with actions:

$$\mu(\widehat{\beta}_i) = \bigcup_{\alpha_1 \cdots \alpha_i} \beta_i \tag{6.22}$$

$$\mu(\widehat{\beta}_i \sigma u a) = \bigcup_{\alpha_1 \cdots \alpha_i} \beta_i \sigma u a \ , \tag{6.23}$$

where $u_i \notin B$. There are two induction cases, corresponding to (6.22) and (6.23).

The case corresponding to (6.22) is as follows. For $i > 0$, we would like to prove

$$\mathrm{Pr}^{\widehat{\eta}}(\widehat{\beta}_{i-1} \sigma_i s_i \langle t_i, a_i \rangle) = \mathrm{Pr}^{\eta}\Big(\bigcup_{\alpha_1 \cdots \alpha_i} \beta_{i-1} \sigma_i s_i \alpha_i t_i a_i \Big) \ . \tag{6.24}$$

We have

$$\mathrm{Pr}^{\widehat{\eta}}(\widehat{\beta}_{i-1} \sigma_i s_i \langle t_i, a_i \rangle) = \widetilde{Q}_{\widehat{\eta}}(\langle t_i, a_i \rangle \mid \widehat{\beta}_{i-1} \sigma_i s_i) \, \mathrm{Pr}^{\widehat{\eta}}(\widehat{\beta}_{i-1} \sigma_i s_i) \tag{6.25}$$

$$\mathrm{Pr}^{\eta}\Big(\bigcup_{\alpha_1 \cdots \alpha_i} \beta_{i-1} \sigma_i s_i \alpha_i t_i a_i \Big) = \sum_{\alpha_1 \cdots \alpha_n} \mathrm{Pr}^{\eta}(\beta_{i-1} \sigma_i s_i \alpha_i t_i a_i) \ . \tag{6.26}$$

By induction hypothesis,

$$\mathrm{Pr}^{\widehat{\eta}}(\widehat{\beta}_{i-1} \sigma_i s_i) = \sum_{\alpha_1 \cdots \alpha_{i-1}} \mathrm{Pr}^{\eta}(\beta_{i-1} \sigma_i s_i) \ ,$$

and the result (6.24) follows from (6.25), (6.26) and (6.17). From (6.24), since

$$\widehat{p}_{s_i, u}(\langle t_i, a_i \rangle) = p_{t_i, u}(a)$$

we have

$$\mathrm{Pr}^{\widehat{\eta}}(\widehat{\beta}_{i-1} \sigma_i s_i \langle t_i, a_i \rangle u) = \mathrm{Pr}^{\eta}\Big(\bigcup_{\alpha_1 \cdots \alpha_i} \beta_{i-1} \sigma_i s_i \alpha_i t_i a_i u \Big) \ ,$$

which concludes the first induction case. The other case, corresponding to (6.23), can be proved similarly using (6.18). ∎

Corollary 6.3 *Consider a state $s \in S$, a policy η for Π, and the related semi-policy $\widehat{\eta}$ for $\widehat{\Pi}$. Let $\mathcal{L} \subseteq \mathbb{R}$ be a subset of real numbers. Then,*

$$\mathrm{Pr}_s^{\eta}\Big(I \wedge \mathcal{H}_{\bullet}(\omega) \cap \mathcal{L} \neq \emptyset \Big) \ = \ \mathrm{Pr}_s^{\widehat{\eta}}\Big(\widehat{\mathcal{H}}_{\bullet}(\omega) \cap \mathcal{L} \neq \emptyset \Big) \ . \tag{6.27}$$

Moreover, if $\widehat{\eta}'$ is the policy obtained from the semi-policy $\widehat{\eta}$ by re-normalizing to 1 the summation *of the conditional probabilities in (6.19), then*

$$\Pr_s^{\eta}\Big(I \wedge \mathcal{H}_\bullet(\omega) \cap \mathcal{L} \neq \emptyset\Big) \ \leq\ \Pr_s^{\widehat{\eta}'}\Big(\mathcal{H}_\bullet(\omega) \cap \mathcal{L} \neq \emptyset\Big)\ . \tag{6.28}$$

Proof. The relation μ on behavior prefixes induces a relation $\vec{\mu}$ on behaviors: two behaviors are related iff all their prefixes are. Since the end components eliminated by Algorithm 6.5 are composed of state-action pairs with both $R = 0$ and $W = 0$, behaviors related by $\vec{\mu}$ share the same set of limit points for the sequence $\mathcal{H}_n$. Precisely, if $\omega \in \vec{\mu}(\widehat{\omega})$, then $\mathcal{H}_\bullet(\omega) = \widehat{\mathcal{H}}_\bullet(\widehat{\omega})$. Moreover, I holds for a behavior ω of Π precisely if there is $\widehat{\omega}$ of $\widehat{\Pi}$ such that $\omega \in \vec{\mu}(\widehat{\omega})$. Equation (6.27) follows from these observations and from the previous lemma.

Relation (6.28) is a consequence of the fact that, after the re-normalization step, it is

$$\Pr_s^{\widehat{\eta}}(\omega \in G) \leq \Pr_s^{\widehat{\eta}'}(\omega \in G)$$

for any measurable set of behaviors $G \in \mathcal{B}_s$. ∎

With the help of this lemma, we can relate the model-checking problem on Π and $\widehat{\Pi}$ as follows.

Theorem 6.6 (model checking on I-transformed MDPs) *For MDPs Π and $\widehat{\Pi}$, it is:*

$$\exists\eta\,.\,\Pr_s^{\eta}\Big(I \rightarrow \liminf_{n\to\infty} \mathcal{H}_n(\omega) \bowtie a\Big) > 0 \quad \textit{iff} \quad \exists\widehat{\eta}\,.\,\Pr_s^{\widehat{\eta}}\Big(\liminf_{n\to\infty} \widehat{\mathcal{H}}_n(\omega) \bowtie a\Big) > 0 \tag{6.29}$$

for every $a \in \mathbb{R}$, $\bowtie \in \{\leq, <\}$ and $s \in S$. A similar relation holds for $\liminf$ *and $\bowtie \in \{\geq, >\}$.*

Proof. The left-to-right direction of the equivalence in (6.29) is a consequence of Corollary 6.2, Lemma 6.3, and Corollary 6.3. We postpone a proof of the other direction, since the result will follow more easily after an analysis of the model-checking problem on $\widehat{\Pi}$. ∎

6.5 Reduction to Semi-Markov Optimization Problems

We now turn our attention to the model checking problem on the proper MDP obtained by using Algorithm 6.5. We begin by studying the probability distribution of $\mathcal{H}_\bullet$ on Π. Our intent is to show that $\mathcal{H}_\bullet$ belongs with probability 1 to a set that can be characterized in terms of the state-action frequencies arising from deterministic policies. This connection will enable us to reduce the model-checking problem to maximization and minimization problems that are related to optimization problems studied in the theory of semi-Markov decision processes.

6.5.1 The Set $\bar{U}$ of State-Action Frequencies

Let $\chi_\Pi = \{\xi_1, \ldots, \xi_m\}$ be the set of state-action pairs of Π, and define for $n \geq 0$ the vector

$$\boldsymbol{v}_n = \frac{1}{n}\Big[N^n_{\xi_1}, \ldots, N^n_{\xi_m}\Big] ,$$

where $N^n_{\xi_1}, \ldots, N^n_{\xi_m}$ are the random variables denoting the number of times a behavior has taken the state-action pairs $\xi_1, \ldots, \xi_n$ before position n. Let $\boldsymbol{v}_\bullet$ be the set of accumulation points of the sequence $\{\boldsymbol{v}_n\}_{n\geq 0}$.

Given a policy η and a state s, define also

$$\boldsymbol{u}^\eta_{s,n} = \frac{1}{n}\Big[\mathrm{E}^\eta_s\{N^n_{\xi_1}\}, \ldots, \mathrm{E}^\eta_s\{N^n_{\xi_m}\}\Big] . \tag{6.30}$$

The elements of this vector are the expected state-action frequencies up to position n, measured from state s under policy η. Let $\boldsymbol{u}^\eta_{s,\bullet}$ be the set of accumulation points of the sequence $\{\boldsymbol{u}^\eta_{s,n}\}_{n\geq 0}$.

Define $U = \bigcup_{s\in S, \eta\in\boldsymbol{\eta}_D} \boldsymbol{u}^\eta_{s,\bullet}$ to be the set of accumulation points corresponding to deterministic policies. Finally, let $\bar{U}$ be the convex hull of U. Notice that $\bar{U}$, considered in the extended real space completed with the infinity points, is a closed set, since there is only a finite number of deterministic policies.

The following theorem, due to Derman [Der64] (see also [Der70, Chapter 7]), relates $\bar{U}$ to the probability distribution of $\boldsymbol{v}_\bullet$.

Theorem 6.7 (accumulation points of state-action frequencies [Derman, 1964]) *For any state s and any policy η,*

$$\Pr{}^\eta_s(\boldsymbol{v}_\bullet \subseteq \bar{U}) = 1 .$$

To use this result, we have to relate the accumulation points of the sequence $\{\mathcal{H}_n\}_{n\geq 0}$ to the ones of the sequence $\{\boldsymbol{v}_n\}_{n\geq 0}$. First, we need a technical result about the points $\boldsymbol{u} \in \bar{U}$.

Lemma 6.5 *If $\boldsymbol{u} \in \bar{U}$, then there is $1 \leq i \leq m$ such that $u_i > 0$ and either $R(\xi_i) > 0$ or $W(\xi_i) > 0$.*

Proof. Consider a deterministic policy $\eta \in \boldsymbol{\eta}_D$. Note that

$$\boldsymbol{u}^\eta_{s,\bullet} = \Big\{[x_{\xi_1}(s,\eta), \ldots, x_{\xi_m}(s,\eta)]\Big\} , \tag{6.31}$$

where $x_\xi(s,\eta)$ is the frequency of state-action pair ξ starting at state s under policy η (see Section 3.3.3). The closed recurrent classes under η correspond to end components of the MDP. By Algorithm 6.5 (or since the MDP is proper), each end component contains a state-action pair ξ such that $R(\xi) > 0 \vee W(\xi) > 0$. Thus, it is

$$V(s,\eta) = \sum_{i=1}^{m} x_{\xi_i}(s,\eta)\, W(\xi_i) + \sum_{i=1}^{m} x_{\xi_i}(s,\eta)\, R(\xi_i)\, \delta_{R(\xi_i)>0} > 0 ,$$

where $\delta_{R(\xi)>0}$ is equal to 1 if $R(\xi) > 0$, and is equal to 0 otherwise. Define

$$V^- = \min_{\eta \in \boldsymbol{\eta}_D, s \in S} V(s, \eta) ,$$

since there are only finitely many $s \in S$ and $\eta \in \boldsymbol{\eta}_D$, it is $V^- > 0$. For any $\boldsymbol{u} \in \bar{U}$ define

$$V(\boldsymbol{u}) = \sum_{i=1}^{m} u_i(s,\eta)\, W(\xi_i) + \sum_{i=1}^{m} u_i(s,\eta)\, R(\xi_i)\, \delta_{R(\xi_i)>0} .$$

Since $\boldsymbol{u} \in \bar{U}$ and since $\bar{U}$ is the convex hull determined by all initial states and deterministic policies, it is $V(\boldsymbol{u}) > V^-$, and this leads immediately to the result. ∎

To relate the accumulation points $\mathcal{H}_\bullet$ and $\boldsymbol{v}_\bullet$, given $\boldsymbol{v}$ define

$$f(\boldsymbol{v}) = \frac{\sum_{i=1}^{m} R(\xi_i) v_i}{\sum_{i=1}^{m} W(\xi_i) v_i} .$$

The following lemma relates $\mathcal{H}_\bullet$ to $\boldsymbol{v}_\bullet$ through the function f.

Lemma 6.6 *Consider a behavior ω such that $\boldsymbol{v}_\bullet \subseteq \bar{U}$. For any $h \in \mathcal{H}_\bullet$, there is $\boldsymbol{v} \in \boldsymbol{v}_\bullet$ such that $h = f(\boldsymbol{v})$.*

Proof. Let $h \in \mathcal{H}_\bullet$ be a limit point of the sequence $\{\mathcal{H}_n\}_{n\geq 0}$. Then, there is a subsequence $\{\mathcal{H}_{n_j}\}_{j\geq 0}$ that has limit h, i.e. such that $\lim_{j\to\infty} \mathcal{H}_{n_j} = h$. Consider the corresponding subsequence $\{\boldsymbol{v}_{n_j}\}_{j\geq 0}$, and let $\boldsymbol{v}$ be a limit point of this sequence. Clearly, $\boldsymbol{v} \in \boldsymbol{v}_\bullet$; it remains to be shown that $f(\boldsymbol{v})$ is well defined, and that $h = f(\boldsymbol{v})$.

To show that $f(\boldsymbol{v})$ is well defined, we must show that the the numerator and the denominator of the fraction defining $f(\boldsymbol{v})$ cannot be both 0. From Lemma 6.5, there is $1 \leq l \leq m$ such that $v_l > 0$ and $R(\xi_l) > 0 \vee W(\xi_l) > 0$. We reason by cases.

- If $W(\xi_l) > 0$, then the denominator cannot be 0.

- If $R(\xi_l) > 0$, the numerator could be 0 only if R were negative for other state-action pairs. This can happen only for P-experiments. By definition of r, w for P-experiments, $R(\xi_l) > 0$ implies $W(\xi_l) > 0$, indicating that the denominator is different from 0.

In either case, the numerator and denominator cannot be both 0, as desired.

To show that $h = f(\boldsymbol{v})$, let $\{\boldsymbol{v}_{k_i}\}_{i\geq 0}$ be a subsequence of $\{\boldsymbol{v}_{n_j}\}_{j\geq 0}$ such that $\lim_{i\to\infty} \boldsymbol{v}_{k_i} = \boldsymbol{v}$.

Clearly, it is also $\lim_{i\to\infty} \mathcal{H}_{k_i} = h$. It is

$$\mathcal{H}_n = \frac{\sum_{k=0}^{n-1} R(X_k, Y_k)}{\sum_{k=0}^{n-1} W(X_k, Y_k)} = \frac{\frac{1}{n}\sum_{l=1}^{m} N^n_{\xi_l}\, R(\xi_l)}{\frac{1}{n}\sum_{l=1}^{m} N^n_{\xi_l}\, W(\xi_l)} = f(\boldsymbol{v}_n) \ ,$$

so that $\mathcal{H}_{k_i} = f(\boldsymbol{v}_{k_i})$. As $f(\boldsymbol{v})$ is well defined, if $|f(\boldsymbol{v})| < \infty$ the result follows from the continuity of $f(\boldsymbol{v})$ in $\boldsymbol{v}$; if $f(\boldsymbol{v}) = \pm\infty$, the result $h = f(\boldsymbol{v})$ follows from the continuity of f when considered as an extended function $\mathbb{R}^m \mapsto \mathbb{R} \cup \{\infty\}$. ∎

Consider the set $f(\bar{U}) \stackrel{def}{=} \{h \in \mathbb{R} \cup \{\pm\infty\} \mid \exists \boldsymbol{u} \in \bar{U} \,.\, h = f(\boldsymbol{u})\}$. We have the following corollary.

Corollary 6.4 *For every state s and policy η,* $\Pr^{\eta}_{s}(\mathcal{H}_{\bullet} \subseteq f(\bar{U})) = 1$.

Proof. Immediate consequence of Theorem 6.7 and Lemma 6.6. ∎

Using this corollary, we can finally relate our model-checking problem to the value of f over the set $\bar{U}$.

Corollary 6.5 *If*

$$\min_{\boldsymbol{u}\in\bar{U}} f(\boldsymbol{u}) \bowtie a \ ,$$

with $\bowtie \in \{\geq, >\}$, *then for every state s and policy η,* $\Pr^{\eta}_{s}(\liminf_{n\to\infty} \mathcal{H}_n \bowtie a) = 1$. *Similarly, if*

$$\max_{\boldsymbol{u}\in\bar{U}} f(\boldsymbol{u}) \bowtie a \ ,$$

with $\bowtie \in \{\leq, <\}$, *then for every state s and policy η,* $\Pr^{\eta}_{s}(\limsup_{n\to\infty} \mathcal{H}_n \bowtie a) = 1$.

Proof: immediate. Note that the use of max and min in the statement of the corollary is justified, since $\bar{U}$ is a closed set. ∎

6.5.2 Reduction to Optimization Problem

By Corollary 6.5, the model-checking problem of the operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ is related to the problem of finding the minimum and maximum values of $f(\boldsymbol{u})$ for $\boldsymbol{u} \in \bar{U}$. These values can be found by solving an optimization problem on the MDP, as we will now show.

For a deterministic (or in fact, Markovian) policy η, we have already remarked that

$$\boldsymbol{u}^{\eta}_{s,\bullet} = \Big\{ [x_{\xi_1}(s,\eta), \ldots, x_{\xi_m}(s,\eta)] \Big\} \ ,$$

so that the vector $\boldsymbol{u}^{\eta}_{s,\bullet}$ consists of the state-action frequencies corresponding to policy η from state s. A vector $\boldsymbol{u} \in \bar{U}$ can thus be written as

$$\boldsymbol{u} = \sum_{s \in S, \eta \in \boldsymbol{\eta}_D} c_{s,\eta}\, \boldsymbol{u}^{\eta}_{s,\bullet} \tag{6.32}$$

for some set of coefficients $\{c_{s,\eta}\}_{s \in S, \eta \in \boldsymbol{\eta}_D}$, where $0 \leq c_{s,\eta} \leq 1$ for all $s \in S, \eta \in \boldsymbol{\eta}_D$, and $\sum_{s \in S, \eta \in \boldsymbol{\eta}_D} c_{s,\eta} = 1$.

We would like to show that the vector $\boldsymbol{u}$ corresponds to the state-action frequencies of some policy $\eta(\boldsymbol{u})$, where we write η as a function of $\boldsymbol{u}$ to emphasize the dependency. To this end, we first show that for each $s \in S$ we can replace a deterministic policy $\eta \in \boldsymbol{\eta}_D$ with a policy $\eta[s]$ such that $\boldsymbol{u}^{\eta[s]}_{t,\bullet} = \boldsymbol{u}^{\eta}_{s,\bullet}$ for all $t \in S$. In words, policy $\eta[s]$ starting from any state gives rise to the same $\boldsymbol{u}$-vector as policy η from s. The policy $\eta[s]$ is history-dependent, and for $n \geq 0$ and any sequence of states $s_0, \ldots, s_n$, the conditional probability $Q_{\eta[s]}(a \mid s_0, \ldots, s_n)$ is defined as follows, for all $a \in A(s_n)$:

- if $s = s_i$ for some $0 \leq i \leq n$, then $Q_{\eta[s]}(a \mid s_0, \ldots, s_n) = Q_{\eta}(a \mid s_n)$;

- if $s \neq s_i$ for all $0 \leq i \leq n$, then $Q_{\eta[s]}(a \mid s_0, \ldots, s_n) = |A(s_n)|^{-1}$.

Since the MDP is strongly connected, state s is reached with probability 1 and within a finite expected number of steps from any state. From this, it is easy to see that $\boldsymbol{u}^{\eta[s]}_{t,\bullet} = \boldsymbol{u}^{\eta}_{s,\bullet}$ for all $t \in S$, as desired. We can thus write

$$\boldsymbol{u} = \sum_{s \in S, \eta \in \boldsymbol{\eta}_D} c_{s,\eta}\, \boldsymbol{u}^{\eta[s]}_{t,\bullet} \tag{6.33}$$

for any $t \in S$. The advantage of this form, compared to (6.32), is that we can assume that the state-frequencies of the policies on the right-hand side are obtained when starting from a fixed state t, equal for all the vectors $\boldsymbol{u}^{\eta[s]}_{t,\bullet}$ being combined.

Once the dependency from the initial state is eliminated, to obtain the policy $\eta(\boldsymbol{u})$ corresponding to $\boldsymbol{u}$ we must somehow "mix" the policies according to (6.33). This "mix" can obtained as indicated by Derman [Der70, Chapter 7, Theorem 1]; see also Derman and Strauch [DS66] and Strauch and Veinott [SV66].

Theorem 6.8 (state-action frequencies correspond to policies) *For every $\boldsymbol{u} \in \bar{U}$ and $t \in S$, there is a policy $\eta(\boldsymbol{u})$ such that $\boldsymbol{u}^{\eta(\boldsymbol{u})}_{t,\bullet} = \{\boldsymbol{u}\}$.*

Proof. The result follows from (6.33) and from [Der70, Chapter 7, Theorem 1]. ∎

Consider now an arbitrary vector $\boldsymbol{u} = [x_{\xi_1}, \ldots, x_{\xi_m}]$. For policy $\eta(\boldsymbol{u})$ and any $s \in S$ we can write:

$$f(\boldsymbol{u}) = \frac{\sum_{i=1}^{m} x_{\xi_i} R(\xi_i)}{\sum_{i=1}^{m} x_{\xi_i} W(\xi_i)} = \lim_{n\to\infty} \frac{\sum_{i=1}^{m} \frac{1}{n} \mathrm{E}_s^{\eta(\boldsymbol{u})}\{N_{\xi_i}^n\} R(\xi_i)}{\sum_{i=1}^{m} \frac{1}{n} \mathrm{E}_s^{\eta(\boldsymbol{u})}\{N_{\xi_i}^n\} W(\xi_i)}$$

$$= \lim_{n\to\infty} \frac{\mathrm{E}_s^{\eta(\boldsymbol{u})}\Big\{\sum_{i=1}^{m} N_{\xi_i}^n R(\xi_i)\Big\}}{\mathrm{E}_s^{\eta(\boldsymbol{u})}\Big\{\sum_{i=1}^{m} N_{\xi_i}^n W(\xi_i)\Big\}} = \lim_{n\to\infty} \frac{\mathrm{E}_s^{\eta(\boldsymbol{u})}\Big\{\sum_{k=0}^{n-1} R(X_k, Y_k)\Big\}}{\mathrm{E}_s^{\eta(\boldsymbol{u})}\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\}} .$$

Thus, if we introduce the quantities

$$J_s^{\eta -} = \liminf_{n\to\infty} \frac{\mathrm{E}_s^{\eta}\Big\{\sum_{k=0}^{n-1} R(X_k, Y_k)\Big\}}{\mathrm{E}_s^{\eta}\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\}} \qquad J_s^{\eta +} = \limsup_{n\to\infty} \frac{\mathrm{E}_s^{\eta}\Big\{\sum_{k=0}^{n-1} R(X_k, Y_k)\Big\}}{\mathrm{E}_s^{\eta}\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\}} \tag{6.34}$$

for every policy η and $s \in S$, we have that $f(\boldsymbol{u}) = J_s^{\eta(\boldsymbol{u})-} = J_s^{\eta(\boldsymbol{u})+}$ for every $\boldsymbol{u} \in \bar{U}$ and $s \in S$. Define

$$J_s^- = \inf_{\eta \in \boldsymbol{\eta}} J_s^{\eta -} \qquad J_s^+ = \sup_{\eta \in \boldsymbol{\eta}} J_s^{\eta +} .$$

First, we show that the values of J_s^- and J_s^+, do not depend on the state s of a strongly connected, proper MDP $\Pi = (S, A, p, R, W)$.

Lemma 6.7 *In a strongly connected, proper MDP, the values of J_s^-, J_s^+ do not depend on s.*

Proof. Consider any policy η and state s. Given another state t, we can construct a history-dependent policy η_s for t as follows. For $n \geq 0$, any sequence of states $s_0 \cdots s_n$, and $a \in A(s_n)$, the conditional probability $Q_{\eta_s}(a \mid s_0 \cdots s_n)$ is defined as follows:

- if $s \neq s_i$ for all $0 \leq i \leq n$, then $Q_{\eta[s]}(a \mid s_0 \cdots s_n) = |A(s_n)|^{-1}$;
- if $s = s_i$ for some $0 \leq i \leq n$, then $Q_{\eta_s}(a \mid s_0 \cdots s_n) = Q_\eta(a \mid s_k \cdots s_n)$, where $k = \min\{i \mid s_i = s\}$.

Until s is reached, η_s chooses at every state $u \neq s$ an action from $A(u)$ uniformly at random. Since the state space is connected, s will be reached in finite expected time. After reaching s, η_s behaves exactly like η, "forgetting" the past up to s. Since s is reached in a finite expected number of steps, $J_s^\eta = J_t^{\eta_s}$. Since η is arbitrary, we have $J_s^- \geq J_t^-$. From the arbitrariness of s, t we can conclude $J_s^- = J_t^-$. An analogous reasoning proves the result for J_s^+. ∎

As J_s^- does not depend from s, we will denote the common value by J^-. Putting together the above arguments, we obtain the following results.

Lemma 6.8 *For each $\boldsymbol{u} \in \bar{U}$ and each state s, there is a policy $\eta(s, \boldsymbol{u})$ such that $f(\boldsymbol{u}) = J_s^{\eta(s,\boldsymbol{u})-} = J_s^{\eta(s,\boldsymbol{u})+}$.*

Proof. Consequence of Theorem 6.8 and of the previous considerations on $f(\boldsymbol{u})$ and J. ∎

Theorem 6.9 (model-checking and optimization of J) *In a strongly connected and proper MDP, for any s and any η it is*

$$\Pr_s^\eta\Big(\liminf_{n\to\infty} \mathcal{H}_n \geq J^-\Big) = 1 \qquad \Pr_s^\eta\Big(\limsup_{n\to\infty} \mathcal{H}_n \leq J^+\Big) = 1 \ .$$

Thus, the following statements hold for all $s \in S$:

$$\bowtie \in \{\geq, >\}: \qquad J^- \bowtie a \quad \textit{iff} \quad \forall \eta \,.\, \Pr_s^\eta(\liminf_{n\to\infty} \mathcal{H}_n \bowtie a) = 1 \tag{6.35}$$

$$\bowtie \in \{\leq, <\}: \qquad J^+ \bowtie a \quad \textit{iff} \quad \forall \eta \,.\, \Pr_s^\eta(\limsup_{n\to\infty} \mathcal{H}_n \bowtie a) = 1 \ . \tag{6.36}$$

Proof. The first statement is a consequence of Corollary 6.5 and of the previous analysis. In the left-to-right directions, the equivalences (6.35) and (6.36) are a consequence of the first statement. The proof of the right-to-left direction is postponed. ∎

This theorem indicates that to solve the model-checking problem for operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$, it suffices to compute the maximum and minimum values of J in the MDP. This is the problem that we solve in the following section.

We conclude the section with a lemma that is the counterpart for J of Lemma 6.6.

Lemma 6.9 *For any s and policy η, define $J_{s,\bullet}^\eta$ to be the set of accumulation points of the sequence*

$$\frac{\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} R(X_k, Y_k)\Big\}}{\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\}}$$

for $n \to \infty$. Then, for every $\gamma \in J_{s,\bullet}^\eta$ there is $\boldsymbol{u} \in \bar{U}$ such that $\gamma = f(\boldsymbol{u})$.

Proof. First, note that

$$\frac{\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} R(X_k, Y_k)\Big\}}{\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\}} = \frac{\mathrm{E}_s^\eta\Big\{\sum_{i=1}^{m} N_{\xi_i}^n R(\xi_i)\Big\}}{\mathrm{E}_s^\eta\Big\{\sum_{i=1}^{m} N_{\xi_i}^n W(\xi_i)\Big\}} .$$

Since $\gamma \in J_{s,\bullet}^\eta$, there is a subsequence $\{n_k\}_{k\geq 0}$ of $0, 1, 2, \ldots$ such that

$$\gamma = \lim_{k\to\infty} \frac{\mathrm{E}_s^\eta\Big\{\sum_{i=1}^{m} N_{\xi_i}^{n_k} R(\xi_i)\Big\}}{\mathrm{E}_s^\eta\Big\{\sum_{i=1}^{m} N_{\xi_i}^{n_k} W(\xi_i)\Big\}} = \lim_{k\to\infty} \frac{\sum_{i=1}^{m} \frac{1}{n_k} \mathrm{E}_s^\eta\{N_{\xi_i}^{n_k}\} R(\xi_i)}{\sum_{i=1}^{m} \frac{1}{n_k} \mathrm{E}_s^\eta\{N_{\xi_i}^{n_k}\} W(\xi_i)} .$$

From the sequence $\{n_k\}_{k\geq 0}$ we can extract a subsequence $\{n_l\}_{l\geq 0}$ such that $\lim_{l\to\infty} \boldsymbol{u}_{s,n_l}^\eta = \boldsymbol{u}$, for some $\boldsymbol{u} \in \bar{U}$. Reasoning as in the proof of Lemma 6.6, using again Lemma 6.5, it can be proved that $\gamma = f(\boldsymbol{u})$, concluding the proof. ∎

6.6 Computing the Maximum and Minimum Values of J

6.6.1 Semi-Markov Decision Processes

The problem of computing the maximum and minimum values of J on an MDP is closely related to the problem of the computation of the maximum and minimum average reward of *semi-Markov decision processes.* Semi-Markov decision processes have been described in Jensen [Jen53], Howard [How60, How63], De Cani [DC64], Veinott [Vei69], Ross [Ros70b, Ros70a]; we will essentially follow the presentation of Bertsekas [Ber95]. A *semi-Markov decision process* is an MDP in which to each state s and action $a \in A(s)$ are associated two quantities:

- An *average permanence time* $W(s, a)$, subject to

$$\forall s \in S,\ \forall a \in A(s) : \qquad W(s, a) > 0 \tag{6.37}$$

  which represents the average time for which the process remains at s when action a is chosen.

- A *reward* $R(s, a)$ received when action a is chosen at state s. The reward $R(s, a)$ is generally related to $W(s, a)$ by $R(s, a) = \bar{R}(s, a)W(s, a)$, where $\bar{R}(s, a)$ is the *instantaneous reward* of action a at s.

Given a policy η, the *average rewards* $J_s^{\eta-}$, $J_s^{\eta+}$ from a state s under η are defined precisely as in (6.34). The optimization problem for the average reward of a semi-Markov decision process consists in computing J^-, J^+, and in computing the policies that lead to this minimum and maximum average rewards.

The aim of the previous subsections was to reduce the model-checking problem for operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ to the problem of the computation of the minimum and maximum average rewards of a semi-Markov decision problem. While Theorem 6.9 appears to be accomplishing such a reduction, the reduction is in fact incomplete. In fact, the traditional analysis of semi-Markov decision problems relies on assumption (6.37), which enables the use of the *uniformation* technique described by Jensen [Jen53], Howard [How60], Veinott [Vei69] and [Sch71] (an account of the technique can also be found in [Ber95]). Most of the known results for semi-Markov problems depend on this technique, and thus on assumption (6.37); this includes the results presented in Ross [Ros70b, Ros70a], Puterman [Put94] and Bertsekas [Ber95]. Thus, while we will be able to borrow from the usual development of the subject many concepts and ideas, we will have to follow a different path to get the desired results, and we will have to provide independent proofs of our statements.

In passing, we note that the definition (6.34) of the average reward of semi-Markov decision processes does not intuitively correspond to the concept of average reward of a policy η. In a later section, we will propose an alternative quantity H, defined by

$$H_s^{\eta -} = \mathrm{E}_s^\eta\Big\{\liminf_{n\to\infty} \mathcal{H}_n(\omega)\Big\} \qquad\qquad H_s^{\eta +} = \mathrm{E}_s^\eta\Big\{\limsup_{n\to\infty} \mathcal{H}_n(\omega)\Big\} \ . \tag{6.38}$$

We will show that this quantity reflects better the intuitive notion of long-term average reward of a semi-Markov process, and we prove that, if the MDP is strongly connected, then $H^- = J^-$ and $H^+ = J^+$, settling a long-standing question in the theory of Markov decision processes. We will also provide an algorithm for the computation of H_s^- on general (not necessarily strongly connected) semi-Markov decision processes.

6.6.2 Computing J for Markovian and Unichain Policies

Before dealing with the general case of history-dependent policies, we study the properties of J under Markovian policies. The results will be used in many subsequent proofs, and will also be of help in understanding the relationship between J and H.

Given a Markovian policy η, define the column vectors $\boldsymbol{R}_\eta = [R_s^\eta]_{s\in S}$ and $\boldsymbol{W}_\eta = [W_s^\eta]_{s\in S}$ by

$$R_s^\eta = \sum_{a\in A(s)} R(s,a)\, Q_\eta(a \mid s) \qquad\qquad W_s^\eta = \sum_{a\in A(s)} W(s,a)\, Q_\eta(a \mid s) \tag{6.39}$$

for all $s \in S$. Moreover, define the transition matrix $P_\eta = [p_{st}^\eta]_{s,t\in S}$ by

$$p_{st}^\eta = \sum_{a\in A(s)} p_{st}(a)\, Q_\eta(a \mid s) \tag{6.40}$$

for all $s, t \in S$. The *limiting matrix* $P^*_\eta = [p^{\eta,*}_{st}]_{s,t\in S}$ is defined by

$$P^*_\eta = \lim_{n\to\infty} \frac{1}{n} \sum_{k=0}^{n-1} P^k_\eta \ ,$$

similarly to (6.2): the element $p^{\eta,*}_{st}$ indicates the long-run average fraction of time spent in t when starting at state s. The following theorem computes $J^{\eta+}_s = J^{\eta-}_s$ for a Markovian policy η as a function of these vectors and matrices.

Theorem 6.10 (J under Markovian policies) *Given a Markovian policy η, it is*

$$J^{\eta+}_s = J^{\eta-}_s = \Big(\sum_{t\in S} p^{\eta,*}_{st} R^\eta_t\Big) \Big/ \Big(\sum_{t\in S} p^{\eta,*}_{st} W^\eta_t\Big) \tag{6.41}$$

for all $s \in S$.

Proof. It is

$$\frac{1}{n} \limsup_{n\to\infty} \mathrm{E}^\eta_s \Big\{ \sum_{k=0}^{n-1} R(X_k, Y_k) \Big\} = \frac{1}{n} \limsup_{n\to\infty} \sum_{k=0}^{n-1} (P^k_\eta \boldsymbol{R}_\eta)_s = (P^*_\eta \boldsymbol{R}_\eta)_s \ ,$$

where $(v)_s$ denotes the component s of vector v. An analogous expression holds for W. By comparing these expressions with (6.34) and using Lemma 6.3, we have $J^{\eta+}_s = (P^*_\eta \boldsymbol{R}_\eta)_s / (P^*_\eta \boldsymbol{W}_\eta)_s$, which is equivalent to (6.41). ∎

This theorem also indicates that if η is Markovian then $J^{\eta+}_s = J^{\eta-}_s$; we will denote this common value simply as J^η_s.

We now define *unichain policies,* for which the above relations assume a particularly simple form (see, for example, Bertsekas [Ber95]).

Definition 6.4 (unichain policies) A Markovian policy η is *unichain* if the Markov chain defined by the state space S and the transition matrix P_η has a single closed recurrent class. ∎

If η is unichain, then standard results on Markov chains ensure that $p^{\eta,*}_{ts} = p^{\eta,*}_{us}$ for all $s, t, u \in S$ (see, for example, Kemeny, Kendall, and Snell [KSK66]). This means that the long-run average fraction of time spent at a state does not depend on the initial state. We can thus define the *steady-state vector* $\boldsymbol{\pi}_\eta = [\pi^\eta_s]_{s\in S}$ by $\pi^\eta_s = p^{\eta,*}_{ts}$ for all $s \in S$, where $t \in S$ is arbitrary. If η is unichain, J^η_s does not depend from $s \in S$, and by Theorem 6.10 it is $J^\eta_s = (\boldsymbol{\pi}_\eta \boldsymbol{R}_\eta)/(\boldsymbol{\pi}_\eta \boldsymbol{W}_\eta)$.

6.6.3 Computing J^- and J^+

To compute the minimum and maximum values for J, it is convenient to introduce the concept of *optimal policies* for J.

Definition 6.5 (optimal policies) If a policy η is such that $J_s^{\eta+} = J^+$ for all states s, then we say that η is *optimal* for J^+. A similar definitions applies J^-. ∎

The first step in the computation of J^- and J^+ consists in checking whether these quantities are finite. This can be done with the following theorem, which parallels Algorithm 6.2.

Theorem 6.11 (convergence of J^+ and J^-) *Consider a proper MDP (S, A, p, W, R). The following assertions hold:*

1. *$-\infty < J^-$ and $-\infty < J^+$.*
2. *If Ψ is a P-experiment, then $J^+ < \infty$, $J^- < \infty$.*
3. *If Ψ is a D-experiment, the convergence of J^+ can be determined as follows. For $s \in S$, let $B(s) = \{a \in A(s) \mid W(s,a) = 0\}$ be the set of actions having $W = 0$. Let $(C_1, D_1), \dots, (C_m, D_m)$ be the maximal end components of (S, B). Then, $J^+ = \infty$ iff there are $j \in [1..m]$, $s \in C_j$ and $a \in D_j(s)$ such that $R(s,a) > 0$.*
4. *If Ψ is a D-experiment, $J^- = \infty$ iff both of the following conditions hold:*
   (a) for all $s \in S$ and $a \in A(s)$, $W(s,a) = 0$;
   (b) there are $s \in S$ and $a \in A(s)$ such that $R(s,a) > 0$. ∎

Moreover, for the cases in which J^+ (resp. J^-) diverge, there is an unichain policy η such that $J_s^\eta = +\infty$ (resp. $J_s^\eta = -\infty$) for all s.

Proof.

1. For a state-action pair $\xi \in \chi_\Pi$, it can be $R(\xi) < 0$ only if Ψ is a P-experiment, in which case $R(\xi) < 0$ implies $W(\xi) > 0$. Let $\chi_{>0} = \{\xi \in \chi_\Pi \mid W(\xi) > 0\}$; the result follows from the fact that the ratio $|R(\xi)/W(\xi)|$ is bounded on $\chi_{>0}$.

2. If Ψ is a P-experiment, the result follows as before from the observation that $|R(\xi)/W(\xi)|$ is bounded on $\chi_{>0}$.

3. First, assume there are $j \in [1..m]$, $s \in C_j$ and $a \in D_j(s)$ such that $R(s,a) > 0$. Consider the Markovian policy η defined as follows:

   - for $s \in C_j$, let $Q_\eta(a \mid s) = |D_j(s)|^{-1}$ if $a \in S_j(s)$ and $Q_\eta(a \mid s) = 0$ otherwise;
   - for $s \notin C_j$, let $Q_\eta(a \mid s) = |A(s)|^{-1}$.

   Again, η is unichain, since the MDP under η has C_j as its single closed recurrent class. Moreover, while in C_j only actions from D_j are taken, and for all $\xi \in sa(C_j, D_j)$ it is $W(\xi) = 0$.

Since for at least one $\xi \in sa(C_j, D_j)$ it is $R(\xi) > 0$, it is $J^\eta = (\boldsymbol{\pi}_\eta \boldsymbol{R}_\eta)/(\boldsymbol{\pi}_\eta \boldsymbol{W}_\eta) = \infty$, which implies $J^+ = \infty$.

Conversely, assume that for every end component (C, D), if there is $\xi \in sa(C, D)$ with $R(\xi) > 0$, then there also is $\zeta \in sa(C, D)$ such that $W(\zeta) > 0$. Consider any accumulation point $\boldsymbol{u} \in \boldsymbol{u}^\eta_{s,\bullet}$ corresponding to any state s and deterministic policy $\eta \in \boldsymbol{\eta}_D$. Since the closed recurrent classes of deterministic policies correspond to end components, the assumption insures that there is M such that $f(\boldsymbol{u}) < M$. Hence, $f(\boldsymbol{u}) < M$ for all $\boldsymbol{u} \in \bar{U}$. By Lemma 6.9, $J^+ = f(\boldsymbol{u})$ for some $\boldsymbol{u} \in \bar{U}$, and this implies $J^+ < M$, concluding the argument.

4. First, suppose that Conditions (4a) and (4b) hold. From Lemma 6.3 we know that along a behaviors there are with probability 1 infinitely many position k such that $R(X_k, Y_k) > 0$. Thus, for any state s and any policy η, $\lim_{n\to\infty} \mathrm{E}^\eta_s\{\sum_{k=0}^{n-1} R(X_k, Y_k)\} = \infty$ while $\lim_{n\to\infty} \mathrm{E}^\eta_s\{\sum_{k=0}^{n-1} W(X_k, Y_k)\} = 0$. As $J^{\eta-}_s = \infty$ for all s, η, we have $J^- = \infty$. The policy η_u that chooses at each state an action with uniform probability is unichain, and is such $J^{\eta_u} = \infty$.

   Conversely, assume that at least one of Conditions (4a) and (4b) does not hold. Since the MDP is proper, it cannot be the case that $R(\xi) = W(\xi) = 0$ for all state-action pairs $\xi \in \chi_\Pi$. Thus, it must be the case that there is at least one state-action pair $(t, b) \in \chi_\Pi$ with $W(t, b) > 0$. Let η be the Markovian policy that at every state s chooses an action from $A(s)$ with uniform probability. Since the MDP is strongly connected and proper, η is unichain, yielding $J^\eta = (\boldsymbol{\pi}_\eta \boldsymbol{R}_\eta)/(\boldsymbol{\pi}_\eta \boldsymbol{W}_\eta)$. Since $\pi_s > 0$ for all $s \in S$, and since $w^\eta_t > 0$, this ratio cannot be infinite. Hence, $J^{\eta-} < \infty$, and thus $J^- < \infty$. ∎

6.6.4 Computation of J^-, J^+ on PBC MDP

First, observe that by considering an MDP in which the sign of $R(\xi)$ has been reversed, for all $\xi \in \chi_\Pi$, the problem of computing J^+ on the original MDP can be reduced to the problem of computing J^- on the new MDP. This latter is the problem we will study in this subsection.

While the algorithms we present could work also for MDPs on which J^- is equal to $\pm\infty$, depending on the implementation details, we will present them only for *bounded* MDPs, which are defined as follows.

Definition 6.6 (bounded MDPs) We say that a strongly connected, proper MDP is *bounded* if $-\infty < J^- < +\infty$. ∎

To check whether an MDP is bounded, we can use Theorem 6.11 to determine whether J^- or J^+ are bounded. This is done before the reduction to the computation of J^-.

For the sake of conciseness, we will call a strongly connected, proper and bounded MDP an *CPB MDP*. For a CPB MDP, we will show that the value of J^- be computed by solving a system of equations known as the *Bellman equations* for semi-Markov processes. This approach has been described by Howard [How60], De Cani [DC64], Veinott [Vei69] and Ross [Ros70b]; a more recent account is provided by Bertsekas [Ber95, Volume II]. Precisely, we show that the system of equations

$$h_s = \min_{a \in A(s)} \left\{ R(s,a) - \lambda W(s,a) + \sum_{t \in S} p_{st}(a) h_t \right\} , \qquad s \in S \tag{BEQ}$$

known as the *Bellman equations* for semi-Markov decision processes, always have at least one solution in a CPB MDP, and that the value of λ at the solutions is equal to J^-. We first show existence of the solutions; then, we show that the value of λ at the solution is equal to J^-. Last, we will see how the value of J^- can be computed by solving a linear programming problem derived from these equations.

Theorem 6.12 (existence of solutions to the Bellman equations) *On a CPB MDP, the equations (BEQ) always have at least one solution.*

The standard proof of this result relies either on uniformation techniques (see Jensen [Jen53], Howard [How60] and Veinott [Vei69]) or on a connection with the stochastic shortest path problem, if all policies are unichain (see Bertsekas [Ber95, Volume II, § 5.3]). These two techniques cannot be applied to our case: uniformation requires that $W(s,a) > 0$ for all $s \in S$, $a \in A(s)$, which does not necessarily hold for our MDPs due to the definition of product of a TPS with an experiment. The connection with stochastic shortest path problems also does not help, since we have no guarantee that all Markovian policies are unichain in our MDPs. We base our proof on a connection with the average reward problem. This connection is in the same spirit of the one with the shortest path problem presented in [Ber95, Volume II, § 5.3], but it turns out to be of greater generality.

Proof. Define a new labeling g on the MDP by $g(s,a) = R(s,a) - J^- W(s,a)$ for all $s \in S$, $a \in A(s)$. The average g-reward at s for policy η, denoted by V_s, is defined by

$$V_s^\eta = \liminf_{n\to\infty} \frac{1}{n} \mathrm{E}_s^\eta \Big\{ \sum_{k=0}^{n-1} g(X_k, Y_k) \Big\} = \liminf_{n\to\infty} \frac{1}{n} \mathrm{E}_s^\eta \Big\{ \sum_{k=0}^{n-1} [R(X_k,Y_k) - J^- W(X_k,Y_k)] \Big\} . \tag{6.42}$$

Let $V_s^* = \inf_\eta V_s^\eta$; since classical results ensure that V_s^* does not depend on s (this can also be proved in a similarly way to Lemma 6.7), we can write simply V^*.

First, we show that $V^* = 0$. In fact, by definition of J^- we have

$$\inf_{s,\eta} \liminf_{n\to\infty} \left[\frac{\mathrm{E}_s^\eta \Big\{ \sum_{k=0}^{n-1} R(X_k,Y_k) \Big\}}{\mathrm{E}_s^\eta \Big\{ \sum_{k=0}^{n-1} W(X_k,Y_k) \Big\}} - J^- \right] = 0 . \tag{6.43}$$

Notice that $W(X_k, Y_k) \geq 0$ for all $k \geq 0$, and

$$\lim_{n\to\infty} \mathrm{E}_s^\eta \Big\{ \sum_{k=0}^{n-1} W(X_k, Y_k) \Big\} > 0 \tag{6.44}$$

for at least some policy η. In fact, if (6.44) did not hold, from (6.43) and Lemma 6.3 it would be $J^{\bar{}} = \infty$, contradicting the fact that the MDP is bounded. Thus, from (6.43) upon multiplication we obtain

$$\begin{aligned} 0 &= \inf_{s,\eta} \liminf_{n\to\infty} \Big[\frac{1}{n} \mathrm{E}_s^\eta \Big\{ \sum_{k=0}^{n-1} R(X_k, Y_k) \Big\} - J^{\bar{}}\, \frac{1}{n} \mathrm{E}_s^\eta \Big\{ \sum_{k=0}^{n-1} W(X_k, Y_k) \Big\} \Big] \\ &= \inf_{s,\eta} \liminf_{n\to\infty} \frac{1}{n} \mathrm{E}_s^\eta \left\{ \sum_{k=0}^{n-1} \Big[R(X_k, Y_k) - J^{\bar{}}\, W(X_k, Y_k) \Big] \right\}, \end{aligned}$$

which by comparison with (6.42) yields $V^* = 0$. From classical results about the average cost of an MDP, we know then that the Bellman equations

$$h_s = \min_{a \in A(s)} \left\{ g(s,a) + \sum_{t \in S} p_{st}(a) h_t \right\}, \qquad s \in S$$

have at least one solution $[h_s^*]_{s\in S}$ [Der70]. Substituting $W(s,a) - J^{\bar{}}\, R(s,a)$ for $g(s,a)$, we see that this equation is the same as (BEQ) with $\lambda = J^{\bar{}}$. This indicates that (BEQ) admits at least the solution $J^{\bar{}}, [h_s^*]_{s\in S}$. ∎

Theorem 6.13 (optimality of solutions) *If (BEQ) admits a solution $\lambda^*, [h_s^*]_{s\in S}$, then $\lambda^* = J^{\bar{}}$. Moreover, consider any policy η that chooses, at each $s \in S$, only actions that realize the minimum of the right hand side of (BEQ). Then, $J_s^{\eta-} = J^{\bar{}}$, for all $s \in S$.*

Proof. This proof is modeled after the one of [Ber95, Chapter 5, Proposition 3.1]; the only difference being that since W can be equal to 0, we must pay attention to the convergence (or lack thereof) of the limits of the expectations. Consider a policy $\eta \in \boldsymbol{\eta}_D$, and let $\boldsymbol{h}^* = [h_s^*]_{s\in S}$. From (BEQ) we have

$$\begin{aligned} \boldsymbol{h}^* &\leq \boldsymbol{R}_\eta - \lambda^* \boldsymbol{W}_\eta + P^\eta \boldsymbol{h}^* \qquad (6.45) \\ &\leq \boldsymbol{R}_\eta - \lambda^* \boldsymbol{W}_\eta + P^\eta (\boldsymbol{R}_\eta - \lambda^* \boldsymbol{W}_\eta + P^\eta \boldsymbol{h}^*) \\ &= \boldsymbol{R}_\eta + P_\eta \boldsymbol{R}_\eta - \lambda^* (\boldsymbol{W}_\eta + P_\eta \boldsymbol{W}_\eta) + P_\eta^2 \boldsymbol{h}^* \\ &\leq \cdots \\ &\leq \Big(\sum_{k=0}^{n-1} P_\eta^k \Big) \boldsymbol{R}_\eta - \lambda^* \Big(\sum_{k=0}^{n-1} P_\eta^k \Big) \boldsymbol{W}_\eta + P_\eta^n \boldsymbol{h}^* \end{aligned}$$

where the inequalities are interpreted componentwise. Reordering the terms, we have

$$\Big(\sum_{k=0}^{n-1} P_\eta^k\Big)\boldsymbol{R}_\eta - \lambda^*\Big(\sum_{k=0}^{n-1} P_\eta^k\Big)\boldsymbol{W}_\eta \geq \boldsymbol{h}^* - P_\eta^n \boldsymbol{h}^*$$

for $n \geq 0$. For a general (not necessarily Markovian) policy η, a similar analysis yields

$$\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} R(X_k, Y_k)\Big\} - \lambda^* \mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\} \geq h_s^* - \mathrm{E}_s^\eta\{h_{X_n}^*\} \tag{6.46}$$

for all $s \in S$. Since all policies (including η) are proper, we know that at least one of the two expectations on the left hand side of (6.46) diverges as $n \to \infty$. We consider the two cases separately.

1. If $\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\}$ diverges as $n \to \infty$, we can divide by it, obtaining

$$\frac{\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} R(X_k, Y_k)\Big\}}{\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\}} - \lambda^* \geq \frac{h_s^* - \mathrm{E}_s^\eta\{h_{X_n}^*\}}{\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\}} .$$

   Since $h_s^* - \mathrm{E}_s^\eta\{h_{X_n}^*\}$ is bounded as $n \to \infty$, the right hand side goes to 0 as $n \to \infty$, and we finally obtain

$$J_s^{\eta -} = \liminf_{n\to\infty} \frac{\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} R(X_k, Y_k)\Big\}}{\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\}} \geq \lambda^* . \tag{6.47}$$

2. If $\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\}$ does not diverge as $n \to \infty$, then $\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} R(X_k, Y_k)\Big\}$ must diverge, by Lemma 6.3. Thus, from

$$\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} R(X_k, Y_k)\Big\} \geq h_s^* - \mathrm{E}_s^\eta\{h_{X_n}^*\} + \lambda^* \mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\}$$

   and from

$$\lim_{n\to\infty} \mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} R(X_k, Y_k)\Big\} = \infty \qquad \lim_{n\to\infty} \mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\} < \infty$$

   we obtain (6.47) once again.

In both cases, we can conclude $J_s^{\eta -} \geq \lambda^*$.

To show that indeed $J^- = \lambda^*$, let η be a policy that chooses at each state $s \in S$ one of the actions that attain the minimum in (BEQ). The above reasoning can be repeated, yielding (6.45), (6.46) and (6.47) with the inequality replaced by equality. Thus, $J_s^\eta = \lambda^*$. Combining this result with $J_s^\eta \geq \lambda^*$ for all η, we obtain $J^- = \lambda^*$, as desired. This also proves the second assertion of the theorem. ∎

Given a solution $\lambda^*, [h_s^*]_{s\in S}$ of (BEQ), we say that a policy η *corresponds* to the solution if, at each state s, it chooses only actions of $A(s)$ that realize the minimum of the right hand side of (BEQ). Since this set of eligible actions depends only on the state s, we have the following corollary.

Corollary 6.6 (existence of deterministic optimal policies) *The following assertions hold:*

1. *There is at least one deterministic policy η that corresponds to a solution of (BEQ), and that satisfies $J_s^\eta = J^-$, for all $s \in S$.*
2. *Let $(B_1, D_1), \ldots, (B_n, D_n)$ be the end components considered in Algorithm 6.5. Then, there is a deterministic optimal policy that corresponds to a solution of (BEQ) and that satisfies $J_s^\eta = J^-$ for all $s \in S$, and additionally that chooses at every $s \in B_i$ the same state-action pair $\langle a_i, t_i\rangle \in A(s)$, for every $1 \leq i \leq n$.*

Proof. The first assertion follows from the previous remarks. The second one follows from the fact that, by construction, all states in B_i share the same actions, cost of the actions, and action destinations, so that (BEQ) behaves in the same way at all states of B_i, for $i \leq i \leq n$. ∎

The value of J^- can be determined by solving a linear programming problem derived from (BEQ), as explained for example in Puterman [Put94] or Bertsekas [Ber95].

Theorem 6.14 (linear programming solution of Bellman equations) *Consider the following linear programming problem, with variables λ, $\{h_s\}_{s\in S}$:*

Maximize λ subject to

$$h_s \leq R(s,a) - \lambda W(s,a) + \sum_{t\in S} p_{st}(a) h_t \qquad s \in S, a \in A(s) .$$

The following assertions hold:

1. *the problem admits at least one solution;*
2. *the variable λ assumes the same value λ^* at all solutions;*
3. *$J^- = \lambda^*$.*

Proof. Clearly, a solution to (BEQ) is a feasible solution of the linear programming problem. Conversely, if $\lambda^\circ, [h_s^\circ]_{s\in S}$ is a feasible solution of the linear programming problem, by reasoning in an analogous way to the proof of Theorem 6.13 we see that $J_s^\eta \geq \lambda^\circ$ for any s and any proper η, implying $J^- \geq \lambda^\circ$. ∎

6.7 The Correctness Proof

In order to complete the correctness proof of the model-checking algorithm for operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$, we need to complete the proofs of Theorem 6.6 and 6.9, and to put all the results together. We begin by completing the proofs of the two theorems.

Proof of Theorem 6.9 (second part). We prove the right-to-left direction of equivalence (6.35) for the case in which $\bowtie$ is $\geq$; the other cases can be proved similarly. In fact, rather than the right-to-left implication, we prove the counterpositive:

> *For all states s, if $J^{-} < a$ then there is η such that* $\Pr_s^\eta(\liminf_{n\to\infty} \mathcal{H}_n \geq a) < 1$.

We first consider the case in which $J^{-} > -\infty$. Assume thus $-\infty < J^{-} < a$. By Corollary 6.6, there is a deterministic policy η such that $J_s^\eta = J^{-}$. Let $C_1, \ldots C_m \subseteq S$ be the closed recurrent classes of the Markov chain corresponding to the MDP under η, and let $\boldsymbol{\pi}_1 = [\pi_t^1]_{t\in C_1}, \ldots, \boldsymbol{\pi}_m = [\pi_t^m]_{t\in C_1}$ be the steady state distributions of classes $C_1, \ldots, C_m$ when considered in isolation. From Theorem 6.10, we have

$$J_s^\eta = \frac{\displaystyle\sum_{t\in S} p_{st}^{\eta,*} R_t^\eta}{\displaystyle\sum_{t\in S} p_{st}^{\eta,*} W_t^\eta} = \frac{\sum_{j=1}^m q_{sj} r_j}{\sum_{j=1}^m q_{sj} w_j} , \tag{6.48}$$

where

$$q_{sj} = \sum_{t\in C_j} p_{st}^{\eta,*} \qquad r_j = \sum_{t\in C_j} \pi_t^j R_t^\eta \qquad w_j = \sum_{t\in C_j} \pi_t^j W_t^\eta$$

for all $1 \leq j \leq m$. Since η is optimal for J, it must be

$$\frac{r_1}{w_1} = \cdots = \frac{r_m}{w_m} = J . \tag{6.49}$$

To see this, assume towards the contradiction that there are $1 \leq i, j \leq m$ with $r_i/w_i < r_j/w_j$. Then, for $s \in C_i$, $t \in C_j$, it would be $J_s^\eta = r_i/w_i < r_j/w_j = J_s^\eta$, contradicting the fact that J_s^η does not depend on s, since η is optimal.

Consider the behaviors that eventually reach C_j, with $1 \leq j \leq m$. For these behaviors, with probability 1 it is

$$\lim_{n\to\infty} \frac{\displaystyle\sum_{k=0}^{n-1} R(X_k, Y_k)}{\displaystyle\sum_{k=0}^{n-1} W(X_k, Y_k)} = \frac{r_j}{w_j} .$$

Thus, since all behaviors enter one of $C_1, \ldots, C_m$ with probability 1,

$$\Pr_s^\eta\Big(\lim_{n\to\infty} \mathcal{H}_n = J^{-}\Big) = 1 , \tag{6.50}$$

and the result then follows from the fact that $J^- < a$.

If $J^- = -\infty$, by Theorem 6.11 there is an unichain policy η such that $J_s^\eta = -\infty$. Reasoning as above, it is easy to see that (6.50) holds also in this case. ∎

Proof of Theorem 6.6 (second part). We must prove the right-to-left direction of (6.29), i.e.

$$\textit{if} \quad \exists\widehat{\eta} \,.\, \Pr_s^{\widehat{\eta}}\Big(\liminf_{n\to\infty} \widehat{\mathcal{H}}_n(\omega) \bowtie a\Big) > 0 \quad \textit{then} \quad \exists\eta \,.\, \Pr_s^{\eta}\Big(I \to \liminf_{n\to\infty} \mathcal{H}_n(\omega) \bowtie a\Big) > 0$$

where $\bowtie \in \{\leq, <\}$; the other case is similar. To facilitate the proof of the subsequent Corollary 6.7 we do a bit of extra work in this proof. We show that if the left hand side of the above implication holds, not only we can find a policy η for the right hand side, but we can also choose η from the set of unichain policies.

Assume that there is a policy $\widehat{\eta}$ such that

$$\Pr_s^{\widehat{\eta}}\Big(\liminf_{n\to\infty} \widehat{\mathcal{H}}_n(\omega) \bowtie a\Big) > 0 \,.$$

From Corollary 6.4, it is $\liminf_{n\to\infty} \widehat{\mathcal{H}}_n(\omega) \in \bar{U}$ with probability 1. Using Lemma 6.8, we have therefore that $J^- \bowtie a$.

If $J^- = -\infty$, from the structure of Algorithm 6.5 and from the divergence criteria stated in Theorem 6.11 we easily see that we can directly construct an unichain policy η for Π such that

$$\Pr_s^{\eta}\Big(\lim_{n\to\infty} \mathcal{H}_n(\omega) = -\infty\Big) = 1 \,.$$

If $J^- > -\infty$, let η_d be the deterministic policy having the properties stated by Corollary 6.6, Part 2.

Our first step in constructing η consists in obtaining an unichain policy η_u for $\widehat{\Pi}$ such that $J^{\eta_u} = J^-$. If η_d is already unichain, we take simply $\eta_u = \eta_d$. Otherwise, η_d will give rise to some closed recurrent classes; let them be $C_1, \ldots, C_m$. By repeating the analysis of the previous proof, (6.49) applies also to these classes. We now construct a Markovian policy η_u in such a way that C_1 is its only recurrent class. Policy η_u is defined as follows:

- On $s \in C_1$, policy η_u coincides with η_d.
- On $s \notin C_1$, policy η_u chooses an action from $\widehat{A}(s)$ uniformly at random.

Since the MDP is strongly connected, η_u has C_1 as single closed recurrent class, and by comparison with (6.49) we see that $J_s^{\eta_u} = J^-$ for all s. Proceeding as in the previous proof, we can show that

$$\Pr_s^{\eta_u}\Big(\lim_{n\to\infty} \widehat{\mathcal{H}}_n(\omega) = J^-\Big) = 1 \,. \tag{6.51}$$

From η_u we can construct a policy η for Π as follows. Let $(B_1, D_1), \ldots, (B_n, D_n)$ be the end components considered in Algorithm 6.5. Note that for each $1 \leq i \leq n$, it is either $B_i \subseteq C_1$ or $B_i \cap C_1 = \emptyset$: this is a consequence of the fact that policy η_d chooses the same action at all states of B_i. If $B_i \subseteq C_1$, indicate by $\langle t_i, a_i \rangle$ the action chosen by η_u at all states of B_i, for $1 \leq i \leq n$. The Markovian policy η for Π is defined as follows.

- If $s \notin C_1$, policy η chooses an action from $A(s)$ uniformly at random.
- If $s \in C_1 - \bigcup_{i=1}^n B_i$, policy η chooses the same action chosen by η_u at s.
- If $s \in C_1 \cap B_i$, for $1 \leq i \leq n$, there are two cases:
  - if $s = t_i$, then η deterministically chooses action a_i;
  - if $s \neq t_i$, then η chooses an action from $D_i(s)$ uniformly at random.

From this definition, and from the fact that either $B_i \subseteq C_1$ or $B_i \cap C_1 = \emptyset$ for all $1 \leq i \leq n$ as earlier remarked, follows that η is also unichain, with C_1 as its only closed recurrent class.

Under policy η, outside C_1 a behavior follows a random walk. This random walk leads the behavior to C_1 with probability 1, since the MDP is strongly connected.

Once in C_1, policies η and η_u coincide in $C_1 - \bigcup_{i=1}^n B_i$. When a state $s \in B_i$ is reached, $1 \leq i \leq m$, policy η_u chooses deterministically action $\langle t_i, a_i \rangle \in \widehat{A}(s)$. If $s = t_i$, then η can simulate η_u directly by choosing action a_i. If $s \neq t$, however, this is not possible. In this case, η starts a random walk in a Markov chain that has (B_i, D_i) as closed recurrent class. The random walk will reach with probability 1 state t_i: once at t_i, η can choose action a_i, and the simulation between η and η_u continues.

Thus, both the behaviors of $\widehat{\Pi}$ under η_u and Π under η enter C_1 with probability 1. Once in C_1, the behaviors of Π simulate those of $\widehat{\Pi}$ in the manner indicated above. Thus, from (6.51) follows

$$\Pr_s^{\eta}\left(\lim_{n\to\infty} \mathcal{H}_n(\omega) = J^-\right) = 1$$

which leads to the desired conclusion.

A more formal proof of this statement can be reached by formalizing the simulation relation between η and $\widehat{\eta}$. This can be done similarly to Section 6.4, constructing a relation μ between behavior prefixes and proving results similar to Lemma 6.4 and Corollary 6.3 for it. ∎

We can finally state and prove the correctness theorem for the model-checking algorithm.

Theorem 6.15 (correctness of model-checking algorithm for $\bar{\mathrm{P}}$, $\bar{\mathrm{D}}$) *The algorithm for the model-checking of operators $\bar{\mathrm{P}}$, $\bar{\mathrm{D}}$ presented in Section 6.1.2 is correct.*

Proof. The result is a consequence of Corollary 6.2 and Theorems 6.4, 6.5, 6.6, 6.9, 6.11 and 6.14. ∎

We conclude this section by stating a corollary that will be needed in Chapter 8 to justify the model-checking algorithms for fair probabilistic systems.

Corollary 6.7 (existence of unichain optimal policies) *Let $\mathcal{L}$, K^- and K^+ be the lists of sub-MDPs defined in Section 6.1.2, and consider a sub-MDP $(S_i, A_i, p^i, R_i, W_i)$. The following assertions hold.*

1. *If $i \in K^-$ (resp. $i \in K^+$), let λ_i^- (resp. λ_i^+) be the outcome computed as in Section 6.1.2. Then, there is an unichain policy η for Π_i such that $(\boldsymbol{\pi}_\eta \boldsymbol{R}_\eta)/(\boldsymbol{\pi}_\eta \boldsymbol{W}_\eta) = \lambda_i^-$ (resp. $= \lambda_i^+$), so that*
$$\Pr_s^\eta\Big(\lim_{n\to\infty} \mathcal{H}_n(\omega) = \lambda_i^-\Big) = 1$$
for all $s \in S_i$ (and resp. for λ^+).

2. *If $i \in \mathcal{L} - K^-$ there is an unichain policy η for Π_i such that*
$$\Pr_s^\eta\Big(\lim_{n\to\infty} \mathcal{H}_n(\omega) = -\infty\Big) = 1 \tag{6.52}$$
for all $s \in S_i$. If $i \in \mathcal{L} - K^+$, a result similar to (6.52) holds, where $-\infty$ is replaced by $+\infty$.

Proof. This corollary follows immediately from an analysis of the proofs of Theorems 6.9 and 6.6. ∎

6.8 Average Reward of Semi-Markov Decision Processes

As mentioned earlier, a *semi-Markov decision process* (SMDP) $\Pi = (S, A, p, R, W)$ consists of a Markov decision process (S, A, p), together with two additional labelings R and W. For a state-action pair $(s, a) \in \chi_\Pi$, $R(s, a)$ represents the reward received when action a is selected at s, and $W(s, a) > 0$ represents the time spent at s when a is selected.

As for ordinary Markov decision processes, several optimization problems can be formulated for semi-Markov decision processes; among them, the minimization or maximization of total, discounted or average reward. Ross [Ros70a] and Bertsekas [Ber95] provide an account of the subject; further references were given in Section 6.6.1. In this section we study the optimization of the average reward of SMDPs. We consider two optimization criteria: the first is the classical one, and corresponds to the quantity J considered in the previous sections; the second is a new quantity H, whose definition is related to the expected value of the limit points of the sequence $\{\mathcal{H}_n\}_{n\geq 0}$.

We will show that the first criterion does not have a strong semantical basis: in fact, the value of J for a fixed policy does not immediately correspond to quantities of interest in the SMDP. Nonetheless, the literature on semi-Markov decision processes is uniformly based on this criterion, since it is amenable to a mathematical analysis that is well understood.

The second criterion, based on the sequence $\{\mathcal{H}_n\}_{n\geq 0}$, is semantically sound, but has so far resisted the attempts to formulate optimization methods for it. The problem stated in Ross [Ros70b, Ros70a] of proving the equivalence of these two optimization criteria has not been solved so far.

Using the results of the previous sections, we will prove that the two optimization criteria are indeed equivalent on strongly connected SMDPs, and we will provide an algorithm for the optimization of the average reward of general SMDPs using the second criterion. This will close the long-standing open question of the optimization criteria and methods for the average reward of SMDPs.

6.8.1 The two Criteria: J and H

In the rest of the section, we will consider the problem of the minimization of the average reward of an SMDP. The dual problem, concerning the maximization of the average reward, can be reduced to this problem by considering an SMDP in which the sign of the labeling R has been reversed.

Given an SMDP $\Pi = (S, A, p, R, W)$, there are two quantities that can be claimed to capture the intuitive idea of "average reward": J and H. The quantity J is defined as in (6.34):

$$J_s^{\eta-} = \liminf_{n\to\infty} \frac{\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} R(X_k, Y_k)\Big\}}{\mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} W(X_k, Y_k)\Big\}} .$$

The quantity H is defined by

$$H_s^{\eta-} = \mathrm{E}_s^\eta\left\{\liminf_{n\to\infty} \frac{\sum_{k=0}^{n-1} R(X_k, Y_k)}{\sum_{k=0}^{n-1} W(X_k, Y_k)}\right\} = \mathrm{E}_s^\eta\Big\{\liminf_{n\to\infty}\ \mathcal{H}_n(\omega)\Big\} .$$

To see that H represents the intuitive concept of average reward, note that $\liminf_{n\to\infty} \mathcal{H}_n(\omega)$ represents the average reward of a single behavior. The expectation of this quantity corresponds thus to the intuitive concept of average reward of policy η. In Example 6.1 we will further justify the fact that H, and not J, corresponds to intuitive concept of average reward.

The main results of this section are a proof that

$$\min_\eta J_s^{\eta-} = \min_\eta H_s^{\eta-} \tag{6.53}$$

on strongly connected SMDPs, and a method for the computation of $\min_\eta H_s^{\eta-}$ on general, not necessarily strongly-connected, SMDPs. To achieve these results, we begin by studying the properties of H. The aim is to gather more information about H to justify its adoption as the preferred optimization criterion. Moreover, we will show that for an arbitrary policy η and state s, the quantities $J_s^{\eta-}$ and $H_s^{\eta-}$ are not equal, showing that (6.53) is not trivial.

6.8.2 Martingale Property of H

A *martingale* for a stochastic process is a quantity such that its current value is equal to the expectation of its future value; the precise definition can be found, for example, in [Wil91, KSK66]. By showing that H is a martingale, we will obtain a relation between the value of H at a state and at the successor states.

More precisely, the next theorem states that $H_s^{\eta+}$ is equal to the expected value of $H_{X_1}^{\eta'+}$, where η' is the policy corresponding to η after state s. Due to the strong resemblance between this result and the definition of martingale, this result is called the *martingale property of H.*

To state the theorem, given a strategy η denote by $\eta[s_0, \ldots, s_n]$ the modified strategy defined by

$$Q_{\eta[s_0,\ldots,s_n]}(a \mid t_0 \cdots t_m) = Q_\eta(a \mid s_0 \cdots s_n t_0 \cdots t_m) \; .$$

Strategy $\eta[s_0, \ldots, s_n]$ behaves thus as η after the behavior prefix $s_0 \cdots s_n$ has been traversed.

Theorem 6.16 (martingale property of H) *For any policy η, it is*

$$\mathrm{E}^\eta\Big\{H_{X_{n+1}}^{\eta[X_0,\ldots,X_n]-} \;\Big|\; X_0 \cdots X_n\Big\} = H_{X_n}^{\eta[X_0\cdots X_{n-1}]-} \; . \tag{6.54}$$

In particular, if η is a Markovian policy, it is

$$H_{X_n}^{\eta-} = \mathrm{E}^\eta\Big\{H_{X_{n+1}}^{\eta-} \;\Big|\; X_0 \cdots X_n\Big\} \; ,$$

and taking $n = 0$, we can write

$$H_s^{\eta-} = \sum_{t\in S} p_{st}^\eta H_t^{\eta-} \; . \quad \blacksquare \tag{6.55}$$

Proof. To avoid cumbersome notation, we prove only the simpler (6.55). We have

$$H_s^{\eta-} = \mathrm{E}_s^\eta \left\{ \liminf_{n\to\infty} \frac{\sum_{k=0}^{n-1} R(X_k, Y_k)}{\sum_{k=0}^{n-1} W(X_k, Y_k)} \right\}$$

$$= \sum_{t\in S} \mathrm{E}_s^\eta \left\{ \liminf_{n\to\infty} \frac{\sum_{k=0}^{n-1} R(X_k, Y_k)}{\sum_{k=0}^{n-1} W(X_k, Y_k)} \;\middle|\; X_1 = t \right\} p_{st}^\eta$$

$$= \sum_{t \in S} \mathrm{E}_s^\eta \left\{ \liminf_{n\to\infty} \frac{R(X_0,Y_0) + \sum_{k=1}^{n-1} R(X_k,Y_k)}{W(X_0,X_1) + \sum_{k=1}^{n-1} W(X_k,Y_k)} \;\middle|\; X_1 = t \right\} p_{st}^\eta$$

$$= \sum_{t \in S} \mathrm{E}_s^\eta \left\{ \liminf_{n\to\infty} \frac{\sum_{k=1}^{n-1} R(X_k,Y_k)}{\sum_{k=1}^{n-1} W(X_k,Y_k)} \;\middle|\; X_1 = t \right\} p_{st}^\eta \tag{6.56}$$

$$= \sum_{t \in S} H_t^{\eta+} p_{st}^\eta$$

where (6.56) is justified by the fact that $\sum_{k=0}^{n-1} W(X_k,Y_k) = \infty$, since $W(\xi) > 0$ for all $\xi \in \chi_\Pi$. ∎

The interest of the above theorem lies in the fact that it ensures the existence of a policy η that maximizes *simultaneously* $H_s^{\eta-}$ at all $s \in S$. While the formal proof of this will be given later, the intuition is as follows. Consider a state s with possible successors $t_1, \ldots, t_n$. Since $H_s^{\eta-}$ is equal to the expected value of $H_{X_1}^{\eta'-}$, and $X_1 = t_i$ for some $1 \le i \le n$, maximizing the value of H at $t_1, \ldots, t_n$ will help towards the maximization of $H_s^{\eta-}$ as well.

6.8.3 Computing H on Markov Chains

First, we consider the case in which the SMDP corresponds to an ergodic Markov chain, i.e. to a Markov chain whose states form a single, closed recurrent class.

Theorem 6.17 (H on ergodic Markov chains) *If Π is an SMDP corresponding to an ergodic Markov chain, then for any state s it is $H_s^- = (\boldsymbol{\pi R})/(\boldsymbol{\pi W})$.*

Proof. We have to show that

$$\mathrm{E}_s^\eta \left\{ \liminf_{n\to\infty} \frac{\sum_{k=0}^{n-1} R(X_k,Y_k)}{\sum_{k=0}^{n-1} W(X_k,Y_k)} \right\} = \frac{\boldsymbol{\pi R}}{\boldsymbol{\pi W}}$$

for all $s \in S$. Since the chain is ergodic we know that

$$\lim_{n\to\infty} \frac{1}{n} \sum_{k=0}^{n-1} R(X_k,Y_k) = \boldsymbol{\pi R}$$

with probability 1. A similar reasoning, applied to the denominator, concludes the proof. ∎

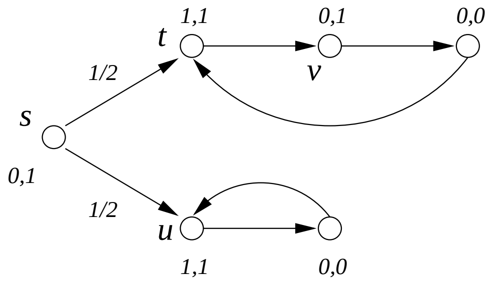

Figure 6.1: A non-ergodic Markov chain. Each state t of the chain is labeled with r_t, w_t.

Combining this result with the martingale property of H, we get the following theorem, which enables the computation of H on a generic Markov chain.

Theorem 6.18 (computation of H on general Markov chains) *Consider an SMDP $\Pi = (S, A, p, R, W)$ corresponding to a Markov chain, and let $C_1, \ldots, C_n$ be the closed recurrent classes of the chain. For each j, with $1 \le j \le n$, let $\boldsymbol{\pi}_j$ be the steady-state distribution of class C_j when considered in isolation.*

- *If $s \in C_i$, for $1 \le i \le n$, then $H_s = (\boldsymbol{\pi}_i \boldsymbol{R})/(\boldsymbol{\pi}_i \boldsymbol{W})$.*
- *If $s \notin \bigcup_{i=1}^n C_i$, then*

$$H_s = \sum_{i=1}^{n} \sum_{t \in C_i} H_t p^*_{st} \ . \tag{6.57}$$

Since $p^*_{st} = 0$ for $t \notin \bigcup_{i=1}^n C_i$, the above relations enable the computation of H on all states of the chain.

Proof. The first result is a restatement of the previous theorem; the second result is a consequence of the first and of the martingale property of H. ∎

6.8.4 Relation Between H and J on Connected SMDPs

First, we present an example that demonstrates that H^-_s and J^-_s are not necessarily equal.

Example 6.1 Consider the Markov chain depicted in Figure 6.1. We have $H_t = 1/2$, $H_u = 1$, $p^*_{st} = p^*_{sv} = 1/6$, $p^*_{su} = 1/4$, so that

$$H_s = \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4} \qquad J_s = \frac{1/6 + 1/4}{1/6 + 1/6 + 1/4} = \frac{5}{7} \ .$$

Intuitively, the difference between the values of J_s and H_s in this example is due to the fact that time advances at different speeds in the two recurrent classes: in one class, time advances on average of 2/3 every transition; in the other, time advances of 1/2 on average every transition. The value of J is affected by this, since the expectations in its definition (6.34) are taken over a horizon defined by the number n of steps taken, rather than the amount of time elapsed. The value of H, on the other hand, is not affected, since the expectation is taken after taking the limit over each behavior. Thus, this example not only demonstrates that H and J can be different, but it also indicates that H is the preferred quantity to be used in the formulation of optimization problems, as it reflects more directly the intuition behind the concept of "long-run average reward". ∎

As remarked in the above example, $J_s^{\eta-}$ and $H_s^{\eta-}$ may be different if the SMDP, under η, is not ergodic. If the SMDP is not strongly connected, also J_s^- and H_s^- can be different, as demonstrated by the above example. However, the next theorem states that J_s^- and H_s^- coincide on strongly connected SMDPs, settling a long-lasting open question. Intuitively, this result is due to the fact that there are optimal policies for J_s^- and H_s^- that are unichain, and the values of $J_s^{\eta-}$ and $H_s^{\eta-}$ coincide on unichain policies, as a comparison between Theorems 6.10 and 6.18 indicates. The proof of this fact, however, rests on the analysis of Sections 6.5 and 6.6.

Theorem 6.19 (equality of J and H on strongly connected SMDPs) *If Π is a strongly connected SMDP, then $J_s^- = H_s^-$ for all states s. Moreover, these quantities do not depend on the state s, so that we can write simply J^- and H^-.*

Proof. From Corollary 6.4, Lemma 6.8 and the definition of $\bar{U}$, we have easily that $H_s^- \geq J_s^-$. To obtain the reverse inequality, we reason as in the second part of the proof of Theorem 6.9. Precisely, if $J_s^- = a$, we can construct a policy η such that $H_s^{\eta-} = a$, and this implies $H_s^- \leq J_s^-$, concluding the proof. ∎

6.8.5 Computing H^- on General SMDP

As stated by Theorem 6.19, on a strongly connected SMDP the quantity H_s^- does not depend on the state s, and it is equal to J^-. These two results do not hold for general SMDPs, as demonstrated in Example 6.1. For a general SMDP $\Pi = (S, A, p, R, W)$, the value of H_s^- at a state $s \in S$ can be computed with the following algorithm. The method we use to go from the solution on a strongly connected SMDP to that on a general SMDP is related to the technique used to solve the *optimization problem for many events* in Courcoubetis and Yannakakis [CY90].

Algorithm 6.6 (computation of H_s^- on general SMDP)

Input: An SMDP $\Pi = (S, A, p, R, W)$ and a state $s_0 \in S$.

Output: H_s^-.

Method: Perform the following steps.

1. Compute the list $\{(S_1, A_1), \ldots, (S_n, A_n)\} = maxEC(S)$ of maximal end components of Π. For each end component (S_i, A_i), $1 \le i \le n$, construct an MDP $\Pi_i = (S_i, A_i, p^i, R_i, W_i)$, where p^i, R_i, W_i are the restrictions of p, R, W to (S_i, A_i).
2. For each Π_i, compute $J_i^- = H_i^-$ using Theorem 6.14.
3. From Π, construct an MDP $\Pi' = (S', A', p')$ having state space $S' = S \cup \{t_1, \ldots, t_n\}$. For every $1 \le i \le n$, state t_i has only one action that leads deterministically to t_i. The actions and transition structure for the states in S are unchanged, except that for $1 \le i \le n$, to each state $s \in S_i$ we add a new action a_i that leads deterministically to t_i.
4. Solve the SSP problem (S', A', p', U, c, g), where the set of destination states is $U = \{t_1, \ldots, t_n\}$, the cost function c is identically 0, and the terminal cost function is defined by $g(t_i) = J_i^-$, for $1 \le i \le n$. Denote the solution by $[v_s^*]_{s \in S'}$.

Output: $v_{s_0}^*$. ∎

Note that the SSP problem constructed by the algorithm does not necessarily satisfy SSP Assumption 2 (see Section 3.4). In the next chapter, we will provide a solution method for *non-negative SSP problems* that can also be applied to this instance of SSP problem. The correctness of this algorithm can be stated and proved as follows.

Theorem 6.20 *Given an SMDP $\Pi = (S, A, p, R, W)$ and a state $s_0 \in S$, let $v_{s_0}^*$ be as computed by Algorithm 6.6. Then, $H_{s_0}^- = v_{s_0}^*$.*

Proof. Consider the SSP problem constructed by the algorithm.

In one direction, we reason as follows. For $1 \le i \le n$, the choice of action a_i at state $s \in S_i$ represents the decision of following, from that point on, the optimal policy for the SMDP Π_i, which yields $H_i^- = J_i^-$. Notice that for $1 \le i \le n$ and $s \in S_i$, state-action pairs (s, a_i) is not part of any end component. From the argument in the proof of Theorem 7.1 in the next chapter, we see that there is a Markovian policy η such that $\boldsymbol{v}^\eta = \boldsymbol{v}^*$, and such that for each $s \in S_i$, action a_i is chosen with probability either 0 or 1, for all $1 \le i \le n$. Let $B_i \subseteq S_i$ be the set of states where a_i is chosen with probability 1 by η, for $1 \le i \le n$.

From the policy η we construct a history-dependent policy η' that, after the sequence $s_0 \cdots s_k$ of states, chooses the next action as follows:

- If $s_j \in \cup_{i=1}^n B_i$ for some $0 \le j \le k$, let $l = \min\{j \mid s_j \in \cup_{i=1}^n B_i\}$ be the position of the first entry in $\cup_{i=1}^n B_i$, and let m be defined by $s_l \in B_m$. Then, η' at s_k follows the Markovian optimal policy for Π_m.
- If $s_j \notin \cup_{i=1}^n B_i$ for all $0 \le j \le k$, then η' coincides with η.

Since η is SSP-proper, under η' with probability 1 a behavior eventually follows an optimal policy for some $\Pi_1, \ldots, \Pi_n$. Since the value of $\mathcal{H}$ is determined by the portion at infinity of the behaviors, from the structure of the SSP problem we see that $H_{s_0}^{\eta'} = v_{s_0}^{\eta} = v_{s_0}^*$. Thus,

$$H_{s_0}^{-} \leq v_{s_0}^* \ . \tag{6.58}$$

In the other direction, let η be a policy for Π. Since $\Pr_{s_0}^{\eta}(\exists i \in [1..n] \ . \ \mathit{inft}(\omega) \in sa(S_i, A_i)) = 1$, it is

$$H_{s_0}^{\eta -} \geq \sum_{i=1}^{n} H_i^{-} \Pr_{s_0}^{\eta}(\mathit{inft}(\omega) \in sa(S_i, A_i)) \ . \tag{6.59}$$

From the relation between Π and the SSP problem, we also have

$$\sum_{i=1}^{n} H_i^{-} \Pr_{s_0}^{\eta}(\mathit{inft}(\omega) \in sa(S_i, A_i)) \geq v_{s_0}^* \ . \tag{6.60}$$

Putting together (6.59) with (6.60), from the arbitrariness of η we have $H_{s_0}^{-} \geq v_{s_0}^*$, which together with (6.58) leads to the result. ∎

Chapter 7

Stochastic Shortest Path and Related Problems

As we have seen in Chapter 4, the model checking of operator D relies on the solution of stochastic shortest path (SSP) problems. The solutions to these problems provide upper and lower bounds for the expected time before a given state formula holds, and these bounds are then used to decide the truth-value of the specification.

In the next chapter we introduce *fairness* in our system model, and we present logics and algorithms for the specification and verification of probabilistic systems with fairness. To solve the model checking problem for the operator D in these logics, it will be necessary to consider instances of the SSP problem that do not satisfy the assumptions described in Section 3.4, and for which no solution methods have so far been presented.

In this chapter, we extend the solvable instances of the SSP problem to two new classes of instances: those in which costs are always non-negative, and those in which costs are always non-positive. The solutions we propose rely on our results on end components, and are related to our Algorithm 3.3 for the efficient computation of reachability probabilities.

These results on the SSP problem will also lead us to new algorithms for the *minimum expected total cost* problem. Even though reductions to linear programming for this problem have been known since Denardo [Den70], by combining the previous analysis with yet another application of end components, we will obtain a more direct, and possibly more efficient, reduction to linear programming for the case of non-negative costs. This result is not used in the verification algorithms presented in this dissertation, and has been included in the dissertation only due to its potential interest for dynamic programming and optimal control.

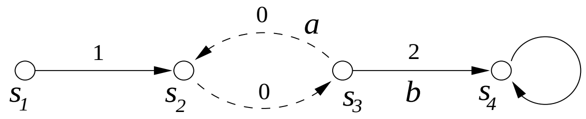

Figure 7.1: An instance of SSP. The set of destination states is $U = \{s_4\}$, and the terminal cost g is 0. All actions are deterministic; their names have been indicated only when there is more than one action available at a state. The actions are labeled with their cost c. The single end component has been indicated with dashed lines.

7.1 The Non-Negative SSP Problem

Consider an instance (S, A, p, U, c, g) of the stochastic shortest path problem. We replace SSP Assumptions 1 and 2 with the following assumptions:

SSP Assumption 3: There is at least one SSP-proper policy.

SSP Assumption 4: For all $s \in S$ and $a \in A(s)$, it is $c(s, a) \geq 0$.

An SSP problem that satisfies SSP Assumptions 3 and 4 is called a *non-negative SSP problem.* SSP Assumption 3 is very similar to SSP Assumption 1, and it can be shown in fact that they are equivalent. SSP Assumption 2, which intuitively stated that all SSP-improper policies must be "hindered" by v diverging for at least one state, has been replaced by SSP Assumption 4, which assumes the non-negativity of the costs.

7.1.1 Relation with Previously Known Solvable Cases

Replacing SSP Assumption 1 with SSP Assumption 3 is merely a matter of notation: in the case of finite Markov decision processes, which is the one that concerns us here, the two assumptions are easily shown to be equivalent. What differentiates the class of SSP instances considered here from the classes that have been discussed so far in the literature is the adoption of SSP Assumption 4 as the only additional assumption.

The class of instances that is most closely related to ours is the one discussed by Bertsekas [Ber87, §6.2, p. 255] and Bertsekas and Tsitsiklis [BT91], which consider the replacement of SSP Assumption 2 with both SSP Assumption 4 and the following assumption:

SSP Assumption 5: There is a proper policy $\eta_0 \in \boldsymbol{\eta}_P$ such that $\boldsymbol{v}^{\eta_0} = \inf_\eta \boldsymbol{v}^\eta$.

In words, SSP Assumption 5 states the existence of a policy that is at the same time proper and optimal from the point of view of expected cost. As noted also in [BT91], there are many instances of SSP in which SSP Assumptions 1 and 4 hold, but 5 does not: one such instance is described in the following example.

Example 7.1 Consider the instance of SSP depicted in Figure 7.1. Clearly, the optimal policy in terms of expected total cost to U is the policy η_1 that chooses always action a at s_3: for this policy, it is:

$$v_{s_1}^{\eta_1} = 1 \qquad v_{s_2}^{\eta_1} = 0 \qquad v_{s_3}^{\eta_1} = 0 \; .$$

On the other hand, for every proper policy $\eta \in \boldsymbol{\eta}_P$ it can be easily verified that

$$v_{s_1}^{\eta} = 3 \qquad v_{s_2}^{\eta} = 2 \qquad v_{s_3}^{\eta} = 2 \; . \qquad \blacksquare$$

7.1.2 Solution Methods

Denote with $\boldsymbol{v} = [v_s]_{s \in S-U}$ a vector of real numbers, and define as in (3.8) the functional L on the space of $\boldsymbol{v}$ by

$$[L\boldsymbol{v}]_s = \min_{a \in A(s)} \left[c(s,a) + \sum_{t \in S-U} p_{st}(a)\, v_t + \sum_{t \in U} p_{st}(a)\, g(t) \right] \qquad s \in S - U \; .$$

In non-negative SSP problems, the functional L can have more than one fixpoint; we will see that the solution of a non-negative SSP problem is the greatest fixpoint of L. The following example illustrates this point.

Example 7.2 Consider again the instance of SSP depicted in Figure 7.1, and write an instance of the column vector $\boldsymbol{v}$ as $[v_{s_1}\ v_{s_2}\ v_{s_3}\ v_{s_4}]^t$. For $x \geq 0$, any vector

$$\boldsymbol{v}(x) = [3 \quad 2 \quad 2 \quad 1]^t - x[1 \quad 1 \quad 1 \quad 0]^t$$

is a fixpoint of L. $\blacksquare$

We present two approaches to the solution of non-negative SSP problems.

The first approach relies on an algorithm that eliminates from the SSP instance the end components consisting of state-action pairs having cost 0. Once this is done, the problem can be solved in the standard way. This approach has two advantages. First, the size of the state space of the problem is reduced before linear programming methods are applied. Second, once the offending end components are removed, other solution methods such as policy iteration and value iteration can be used (for a description of these methods, see for example [Ber95]).

The second approach consists in reducing the SSP problem directly to linear programming: since the solution of the linear programming problem corresponds to the greatest fixpoint, it corresponds to the solution of the SSP problem.

The algorithm for the first approach is related to Algorithms 3.3 and Algorithm 6.5, and is presented below.

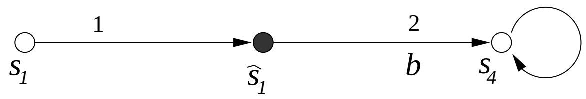

Figure 7.2: Result of applying Algorithm 7.1 to the instance of SSP depicted in Figure 7.1. The new state $\widehat{s}_1$ introduced by the algorithm is drawn as a filled circle.

Algorithm 7.1

Input: Instance $\Pi = (S, A, p, U, c, g)$ of SSP.

Output: Instance $\widehat{\Pi} = (\widehat{S}, \widehat{A}, \widehat{p}, \widehat{U}, \widehat{c}, \widehat{g})$ of SSP.

Method: For each $s \in S - U$, let $D(s) = \{a \in A(s) \mid \mathit{time}(s,a) = 0\}$, and let $\{(B_1, D_1), \ldots, (B_n, D_n)\} = \mathit{maxEC}(S - U, D)$. Define

$$\widehat{S} = S \cup \{\widehat{s}_1, \ldots, \widehat{s}_n\} - \bigcup_{i=1}^{n} B_i$$

and $\widehat{U} = U$. The action sets are defined by:

$$s \in S - \bigcup_{i=1}^{n} B_i : \qquad \widehat{A}(s) = \{\langle s, a\rangle \mid a \in A(s)\}$$

$$1 \le i \le n : \qquad \widehat{A}(\widehat{s}_i) = \Big\{\langle s, a\rangle \;\Big|\; s \in B_i \wedge a \in A(s) - D(s)\Big\} .$$

For $s \in \widehat{S}$, $t \in S - \bigcup_{i=1}^{n} B_i$ and $\langle u, a\rangle \in \widehat{A}(s)$, the transition probabilities are defined by

$$\widehat{p}_{st}(\langle u, a\rangle) = p_{ut}(a) \qquad \widehat{p}_{s,\widehat{s}_i}(\langle u, a\rangle) = \sum_{t \in B_i} p_{ut}(a) . \quad \blacksquare$$

For $s \in \widehat{S}$ and $\langle u, a\rangle \in \widehat{A}(s)$, the cost $\widehat{c}$ is defined by $\widehat{c}(s, \langle u, a\rangle) = c(u, a)$; for $s \in \widehat{U}$ it is $\widehat{g}(s) = g(s)$. $\blacksquare$

Example 7.3 The result of applying Algorithm 7.1 to the instance of SSP depicted in Figure 7.1 is illustrated in Figure 7.2. As we see, the (maximal) end component formed by states s_2, s_3 together with the 0-cost actions has been replaced by the single state $\hat{s}_1$.

A more complex example is presented in Figure 7.3. Algorithm 7.1 computes the end components $(B_1, D_1), (B_2, D_2)$ given by

$$(B_1, D_1) = \mathit{sub}\{(s_3, d), (s_4, k), (s_7, i)\} \qquad (B_2, D_2) = \mathit{sub}\{(s_5, f), (s_6, g)\}$$

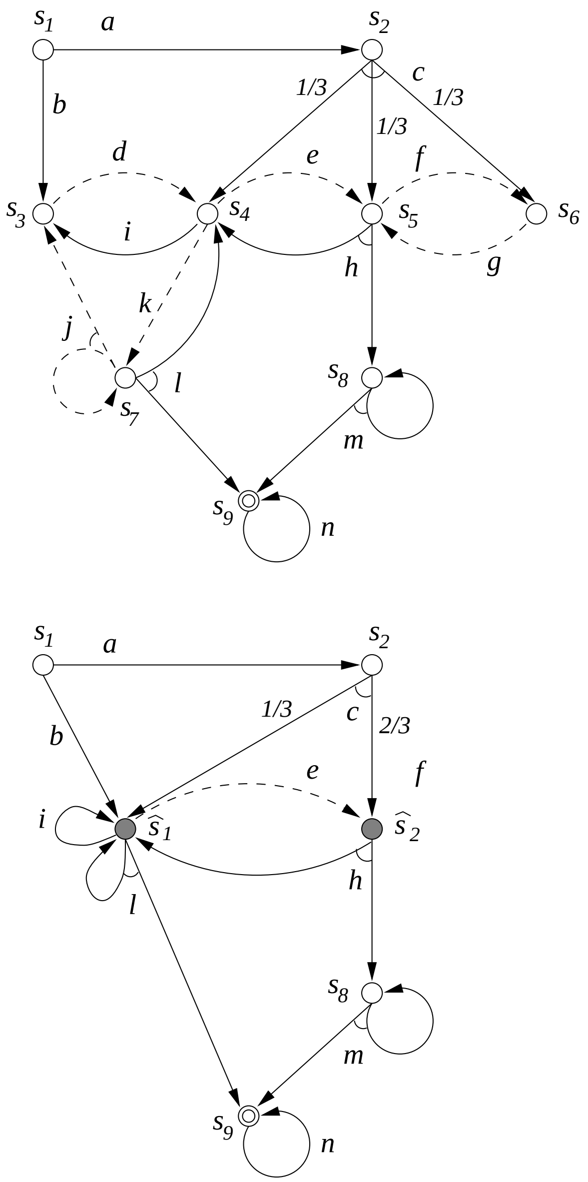

Figure 7.3: An instance of SSP (top), and the result of applying Algorithm 7.1 to it (bottom). The actions that have cost 0 have been represented by dashed edges; those that have positive cost by solid edges. State s_9 is the single destination state. To simplify the diagrams, we have indicated only the transition probability corresponding to action c, and we have omitted all costs. The new states introduced to replace the zero-cost end components are indicated by filled circles.

and replaces them with the two new states $\widehat{s}_1$ and $\widehat{s}_2$. This example illustrates also the potential reduction of the state-space of the system. ∎

The following theorem present the first approach to the solution of non-negative stochastic shortest path problem.

Theorem 7.1 (non-negative SSP solution, approach 1) *Consider an instance* Π *of SSP satisfying SSP Assumptions 3 and 4, and let* $\widehat{\Pi}$ *be the instance obtained from* Π *using Algorithm 7.1. Then,* $\widehat{\Pi}$ *satisfies SSP Assumptions 1 and 2, so that it can be solved using Theorem 3.4. Moreover, it is*

$$\begin{aligned} s \in S \bigcup_{i=1}^{n} B_i : \qquad & v_s^* = \widehat{v}_s^* \\ s \in B_i, 1 \leq i \leq n : \qquad & v_s^* = \widehat{v}_{\hat{s}_i}^* \ , \end{aligned}$$

where $\widehat{\boldsymbol{v}}^*$ *(resp.* $\boldsymbol{v}^*$*) is the solution of* $\widehat{\Pi}$ *(resp.* Π*).*

The second approach is given by the following theorem.

Theorem 7.2 (non-negative SSP solution, approach 2) *Consider an instance* Π *of SSP satisfying SSP Assumptions 3 and 4, and let* L *be the operator defined as in (3.8). Then, the solution* $\boldsymbol{v}^*$ *of the SSP problem is the largest fixpoint of operator* L*. Moreover, the linear programming problem (3.9) has* $\boldsymbol{v}^*$ *as unique solution.*

7.2 The Non-Positive SSP Problem

Consider an instance (S, A, p, U, c, g) of the stochastic shortest path problem. We replace SSP Assumptions 1 and 2 with the following assumptions:

SSP Assumption 3: There is at least one SSP-proper policy.

SSP Assumption 6: For all $s \in S$ and $a \in A(s)$, it is $c(s, a) \leq 0$. For all $s \in U$, it is $g(s) \geq 0$.

An SSP problem that satisfies SSP Assumptions 3 and 6 is called a *non-positive SSP problem.* SSP Assumption 6 is exactly like SSP Assumption 4, except that we require non-positivity of the costs instead of non-negativity.

Denote as usual the optimal cost vector by $\boldsymbol{v}^*$. Unlike in the non-negative case, it is possible that $v_s^* = -\infty$ for some state $s \in S - U$. Our first step towards the solution of non-positive SSP problems consists in determining the set of states on which v^* diverges to $-\infty$.

Lemma 7.1 (divergence of optimal cost of non-positive SSP) *Let*

$$C = \bigcup \Big\{ B' \ \Big| \ \exists (B, D) \in maxEC(S - U) \,.\, \exists (s, a) \in sa(B, D) \,.\, \Big[B' = B \wedge c(s, a) < 0 \Big] \Big\}$$

and let $\widehat{C}$ be the set of states that can reach C. Then, $v_s^* = -\infty$ iff $s \in \widehat{C}$.

Since states in $S - \widehat{C}$ cannot reach $\widehat{C}$, in light of the previous lemma we continue our study of non-positive SSP by adding one more assumption.

SSP Assumption 7: Every end component (B, D) of $S - U$ is such that $c(s, a) = 0$ for all $(s, a) \in sa(B, D)$.

Clearly, if the above assumption does not hold, Lemma 7.1 enables to eliminate the offending end components, along with the states that lead to them.

As in the previous section, the functional L defined by (3.8) can have more than one fixpoint, and the solution of the SSP problem corresponds to the largest fixpoint. This is due to the fact that there can be end components contained in $S - U$ consisting of state-action pairs that all have cost 0.

Again, there are two approaches to the solution of non-positive SSP problems: one consists in using Algorithm 7.1 to eliminate the offending end components, insuring that the functional on the resulting instance of SSP problem has only one solution. As before, this opens the way to the use of linear programming as well as other methods for the solution of the SSP instance. The second approach consists in reducing the problem to linear programming directly. The two approaches are summarized by the two following theorems.

Theorem 7.3 (non-positive SSP solution, approach 1) *Consider an instance Π of SSP satisfying SSP Assumptions 3, 6 and 7, and let $\widehat{\Pi}$ be the instance obtained from Π using Algorithm 7.1. Then, $\widehat{\Pi}$ satisfies SSP Assumptions 1 and 2, so that it can be solved using Theorem 3.4. Moreover, it is*

$$\begin{aligned} s \in S \bigcup_{i=1}^{n} B_i : \qquad & v_s^* = \widehat{v}_s^* \\ s \in B_i, 1 \le i \le n : \qquad & v_s^* = \widehat{v}_{\hat{s}_i}^* \ , \end{aligned}$$

where $\widehat{\boldsymbol{v}}^*$ (resp. $\boldsymbol{v}^*$) is the solution of $\widehat{\Pi}$ (resp. Π).

Theorem 7.4 (non-positive SSP solution, approach 2) *Consider an instance Π of SSP satisfying SSP Assumptions 3 and 4, and let L be the operator defined as in (3.8). Then, the solution $\boldsymbol{v}^*$ of the SSP problem is the largest fixpoint of operator L. Moreover, the linear programming problem (3.9) has $\boldsymbol{v}^*$ as unique solution.*

7.3 Proof of the Results for Non-Negative SSP (‡)

Consider an instance $\Pi = (S, A, p, U, c, g)$ of SSP problem that satisfies SSP Assumptions 3 and 4. To simplify the argument, we assume that once the set U of destination states is entered, it cannot be exited. Formally, this is expressed by the requirement $\mathrm{Succ}(s,a) \subseteq U$ for all $s \in U$ and $a \in A(s)$. It is immediate to see that this assumption does not affect the generality of the arguments.

As our first step towards the proof of Theorem 7.1, we show that a fixpoint of L provides a lower bound for the solution $\boldsymbol{v}^*$. The following theorem is the analogous of Theorem 3.9.

Theorem 7.5 (fixpoint is below solution) *If $\boldsymbol{v}^\bullet = L\boldsymbol{v}^\bullet$, then $v_s^\bullet \le v_s^\eta$ for every $\eta \in \boldsymbol{\eta}_P$ and every $s \in S - U$.*

Proof. From $\boldsymbol{v}^\bullet = L\boldsymbol{v}^\bullet$, it is

$$v_s^\bullet \le c(s,a) + \sum_{t \in S-U} p_{st}(a)\, v_t + \sum_{t \in U} p_{st}(a)\, g(t)$$

for all $s \in S - U$ and all $a \in A(s)$. Iterating, for a general policy η and $n \ge 0$ we obtain

$$v_s^\bullet \le \mathrm{E}_s^\eta\Big\{\sum_{k=0}^{n-1} c(X_k, Y_k)\Big\} + \sum_{t \in S-U} \mathrm{Pr}_s^\eta(X_n = t)\, v_t^\bullet + \sum_{t \in U}\sum_{k=1}^{n} \mathrm{Pr}_s^\eta(X_n = t \wedge T_U = k)\, g(t)\ . \tag{7.1}$$

If $\eta \in \boldsymbol{\eta}_P$, it is

$$\lim_{n\to\infty} \sum_{t \in S-U} \mathrm{Pr}_s^\eta(X_n = t) = 0\ .$$

Taking the limit $n \to \infty$ in (7.1), we have

$$v_s^\bullet \le \mathrm{E}_s^\eta\Big\{\sum_{k=0}^{T_U} c(X_k, Y_k)\Big\} + \mathrm{E}_s^\eta\Big\{g(X_{T_U})\Big\}\ ,$$

which finally yields $v_s^\bullet \le v_s^\eta$, as desired. ∎

We can now prove our main result.

Proof of Theorem 7.1. The proof of this theorem follows closely that of Theorem 3.8. Again, we break up the transformation effected by Algorithm 7.1 in two steps, and we consider an instance $(S, \widetilde{A}, \widetilde{p}, U, \widetilde{c}, \widetilde{g})$ of SSP problem obtained as follows. Let $(B_1, D_1), \ldots, (B_n, D_n)$ be as in Algorithm 7.1, and define

$$\widetilde{A}(s) = \begin{cases} \{\langle s,a\rangle \mid a \in A(s)\} & \text{if } s \notin \bigcup_{i=1}^n B_i; \\ \{\langle t,a\rangle \mid t \in B_i \wedge a \in A(t) - D_i(t)\} & \text{if } s \in B_i,\ 1 \le i \le n. \end{cases} \tag{7.2}$$

and

$$\widetilde{p}_{su}(\langle t,a\rangle) = p_{tu}(a) \qquad \widetilde{c}(s,\langle t,a\rangle) = c(t,a) \qquad \widetilde{g}(s) = g(s) \tag{7.3}$$

for all $s, u \in S$ and $\langle t,a\rangle \in \widetilde{A}(s)$.

Let $\widetilde{L}$ be the operator defined for $\widetilde{\Pi}$ in the same way as L for Π, and let $\boldsymbol{v}^\bullet = \widetilde{L}\boldsymbol{v}^\bullet$ be a fixpoint of $\widetilde{L}$.

Note that $\widetilde{\Pi}$ satisfies SSP Assumption 1 and 2. The fact that the first one holds is a consequence of SSP Assumption 3. To see that SSP Assumption 2 holds, note that for every end component (B, C) in $S - U$ there is $(s, a) \in sa(B, C)$ such that $c(s, a) > 0$: in fact, the end components consisting only of state-action pairs with $c = 0$ have been eliminated by the transformation that led to $\widetilde{\Pi}$. Thus, for a Markovian policy η there are only two cases:

- Under η, a behavior reaches U with probability 1 from all $s \in S - U$.
- Under η, there is an end component (B, C) with $B \subseteq S - U$ such that $\Pr_s^\eta(\mathit{inft}(\omega) = sa(B, C)) > 0$. Then, for these states it is also $v_s^\eta = \infty$, from the above considerations.

This shows that SSP Assumption 2 is satisfied. We can then prove the following facts.

Fact 1. If $\boldsymbol{v}^\bullet = \widetilde{L}\boldsymbol{v}^\bullet$ is a fixpoint of $\widetilde{L}$, and $s, t \in B_i$, for $1 \le i \le n$, then $v_s^\bullet = v_t^\bullet$.

Proved as in the proof of Theorem 3.8.

Fact 2. If $\boldsymbol{v} = \widetilde{L}\boldsymbol{v}$, then $\boldsymbol{v} = L\boldsymbol{v}$.

This fact is proved as in the proof of Theorem 3.8, noting for (3.20) that $c(s, a) = 0$, for $s \in B_i$, $a \in D_i(s)$ and $1 \le i \le n$, by construction of the end components.

Our third fact follows from Theorem 3.4, which can be applied since the SSP problem $\widetilde{\Pi}$ satisfies SSP-assumptions 1 and 2.

Fact 3. There is exactly one $\boldsymbol{v}^\bullet$ such that $\boldsymbol{v}^\bullet = \widetilde{L}\boldsymbol{v}^\bullet$.

The equation $\boldsymbol{v}^\bullet = \widetilde{L}\boldsymbol{v}^\bullet$, written in componentwise notation, is

$$v_s^\bullet = \min_{a \in \widetilde{A}} \Big[c(s, a) + \sum_{t \in S - U} \widetilde{p}_{st}(a)\, v_t^\bullet + \sum_{t \in U} \widetilde{p}_{st}(a)\, g(t) \Big] \qquad s \in S - U \,. \tag{7.4}$$

From the fixpoint $\boldsymbol{v}^\bullet$ of $\widetilde{L}$ we again construct a policy $\widetilde{\eta}$ for $\widetilde{\Pi}$. To this end, for each $s \in S - U$, define the set $A_{min}(s)$ by

$$A_{min}(s) = \arg\min_{a \in \widetilde{A}} \Big[c(s, a) + \sum_{t \in S - U} \widetilde{p}_{st}(a)\, v_t^\bullet + \sum_{t \in U} \widetilde{p}_{st}(a) g(t) \Big] \,. \tag{7.5}$$

At each $s \in S - U$, policy $\widetilde{\eta}$ chooses deterministically an arbitrary action selected from $A_{min}(s)$. For each $1 \le i \le n$, from Fact 1 it is $A_{min}(s) = A_{min}(t)$ for all $s, t \in B_i$. Thus, we can force $\widetilde{\eta}$ to choose the *same* action $\langle u_i, a_i \rangle$ at all states of B_i. By Theorem 3.4, policy $\widetilde{\eta}$ is SSP-proper, and it is $\boldsymbol{v}^{\widetilde{\eta}} = \boldsymbol{v}^\bullet$.

From $\widetilde{\eta}$, we construct a Markovian policy η for Π such that $\boldsymbol{v}^\eta = \boldsymbol{v}^{\widetilde{\eta}}$. At $s \in S - \bigcup_{i=1}^n B_i$, η behaves like $\widetilde{\eta}$. At $s \in B_i$, with $1 \le i \le n$, there are two cases:

- if $s = u_i$, then η chooses deterministically a_i;
- if $s \neq u_i$, then η chooses with uniform probability an action from the set $D_i(s)$.

From the definition of η, we see that η is SSP-proper. Proceeding as in the proof of Lemma 3.4, we have that the system of equations

$$v_s = c(s, \eta(s)) + \sum_{t \in S-U} \widetilde{p}_{st}(\eta(s))\, v_t + \sum_{t \in U} \widetilde{p}_{st}(\eta(s))\, g(t) \qquad s \in S - U \tag{7.6}$$

has $\boldsymbol{v}^\eta$ as unique solution. To show that $\boldsymbol{v}^{\widetilde{\eta}} = \boldsymbol{v}^\eta$, it suffices to show that $\boldsymbol{v}^{\widetilde{\eta}}$ is also a solution of (7.6). The proof proceeds by cases.

- If $s \in B_i$ for $1 \leq i \leq n$, and $s \neq t_i$ (where $\langle t_i, a_i \rangle$ is the action chosen by $\widetilde{\eta}$), then policy η chooses with uniform probability an action from $D_i(s)$. For $a \in D_i(s)$, it is $c(s,a) = 0$ and $\mathrm{Succ}(s,a) \subseteq B_i$. From Fact 1, $\boldsymbol{v}^{\widetilde{\eta}} = \boldsymbol{v}^\bullet$ is constant on B_i, so $v_s^{\widetilde{\eta}} = v_t^{\widetilde{\eta}}$ for $a \in D_i(s)$ and $t \in \mathrm{Succ}(s,a)$, and the result follows.
- If $s \in B_i$ and $s = t_i$, for $1 \leq i \leq n$, or if $s \in S_r - \bigcup_{i=1}^n B_i$, policies η and $\widetilde{\eta}$ coincide on s. The fact that $\boldsymbol{v}^{\widetilde{\eta}}$ satisfies (7.6) for s is then a consequence of $\boldsymbol{v}^{\widetilde{\eta}} = \boldsymbol{v}^\bullet$ and of (7.4), as well as of the fact that $\widetilde{\eta}$ chooses at s an action in $A_{min}(s)$.

Since $\boldsymbol{v}^{\widetilde{\eta}} = \boldsymbol{v}^\eta$, it is also $\boldsymbol{v}^\eta = \boldsymbol{v}^\bullet$, and from Fact 2 we have $\boldsymbol{v}^\eta = \boldsymbol{v}^\bullet = L\boldsymbol{v}^\eta$. By Theorem 7.5, policy η realizes the minimum cost, so that $\boldsymbol{v}^\eta = \boldsymbol{v}^*$.

The proof is then completed in analogy to that of Theorem 3.8, by noting that MDP $\widehat{\Pi}$ is obtained from $\widetilde{\Pi}$ by merging the states belonging to each B_i, for $1 \leq i \leq n$. ∎

Proof of Theorem 7.2. By Theorem 7.5, if $\boldsymbol{v}^\bullet = L\boldsymbol{v}^\bullet$, then $\boldsymbol{v}^\bullet \leq \boldsymbol{v}^\eta$ for every $\eta \in \boldsymbol{\eta}_P$, where the inequality is interpreted componentwise.

From the proof of Theorem 7.1, we know that there is a policy η for Π such that $\boldsymbol{v}^\eta = \boldsymbol{v}^*$. From this follows immediately that $\boldsymbol{v}^\bullet \leq \boldsymbol{v}^*$ for all fixpoints $\boldsymbol{v}^\bullet$ of L. We conclude that $\boldsymbol{v}^*$ is the maximal fixpoint of L. The result about the reduction to linear programming is an immediate consequence of this fact. ∎

7.4 Proof of the Results for Non-Positive SSP (‡)

First, we prove the following lemma, which is equivalent to one of the two directions of Lemma 7.1.

Lemma 7.2 *Assume that there is an end component (B, D) in $S - U$ such that $c(t,a) < 0$ for at least one state-action pair $(t,a) \in sa(B,D)$. Assume also that B is reachable from s. Then,* $v_s^* = \inf_{\eta \in \boldsymbol{\eta}_P} v_s^\eta = -\infty$.

Proof. Since (B, D) is an end component reachable from s, there is a policy η such that $Pr^{\eta}_{s}(\mathit{inft}(\omega) = sa(B, D)) > 0$. Since $c(t, a) < 0$ for at least one state-action pair $(t, a) \in sa(B, D)$, it is $v^{\eta}_{s} = -\infty$. The problem is that policy η is not SSP-proper, since under it a behavior from s is eventually confined to B with positive probability.

To overcome this difficulty, we construct a family of policies $\eta(x)$, for $0 \leq x \leq 1$, in such a way that as $x \to 0$ a behavior from s stays longer and longer in (B, D) before eventually reaching U.

To construct this family of policies, denote by $\widehat{B} \subseteq S - U$ the set of states that can reach B, and let η_m, η_c and η_p be three Markovian policies defines as follows:

- η_m is a policy defined on $\widehat{B} - B$ that maximizes the probability of reaching B.
- η_p is an SSP-proper Markovian policy. From SSP Assumption 3, we know that there is at least one such policy, not necessarily Markovian; it can be easily seen, using reachability arguments for U, that there is also at least one Markovian SSP-proper policy.
- η_c is the policy defined on B that chooses at each $s \in B$ an action from $D(s)$ with uniform probability.

Using these three policies, for each $0 \leq x < 1$ we define the Markovian policy $\eta(x)$ as follows:

$$Q_{\eta(x)}(a \mid s) = \begin{cases} (1-x)\, Q_{\eta_c}(a \mid s) + x\, Q_{\eta_p}(a \mid s) & \text{if } s \in B; \\ (1-x)\, Q_{\eta_m}(a \mid s) + x\, Q_{\eta_p}(a \mid s) & \text{if } s \in \widehat{B} - B; \\ Q_{\eta_p}(a \mid s) & \text{if } s \in S - \widehat{B}. \end{cases}$$

For every $0 < x < 1$, policy $\eta(x)$ is SSP-proper: in fact, it has non-zero probability of reaching U in at most $|S - U|$ steps from each state of $S - U$. Denote by $P(x)$ the matrix of the Markov chain corresponding to $\eta(x)$, for $0 \leq x < 1$, and abbreviate $v^{\eta(x)}_{s}$ by $v_s(x)$. By definition, it is

$$v_s(x) = \mathrm{E}^{\eta(x)}_{s}\left\{ \sum_{k=0}^{T_U - 1} c(X_k, Y_k) \right\} + \mathrm{E}^{\eta(x)}_{s}\left\{ g(X_{T_U}) \right\}. \tag{7.7}$$

Define two vectors $\boldsymbol{z}(x) = [z_s(x)]_{s \in S}$ and $\boldsymbol{u} = [u_s]_{s \in S}$ by

$$z_s(x) = \begin{cases} \mathrm{E}^{\eta(x)}_{s}\left\{ \sum_{k=0}^{T_U - 1} c(X_k, Y_k) \right\} & \text{if } s \in S - U; \\ 0 & \text{otherwise;} \end{cases}$$

$$u_s = \begin{cases} \dfrac{1}{2|D(s)|} \sum_{a \in D(s)} c(s, a) & \text{if } s \in B; \\ 0 & \text{otherwise.} \end{cases}$$

For all $s \in S$, it is $u_s \leq 0$. Since at least one state-action pair of (B, D) has $c < 0$, we know that $u_t < 0$ for at least one $t \in B$. Since $\boldsymbol{u}$ represents only part of the negative cost incurred by a policy that chooses in B with uniform probability the actions in D, for sufficiently small x it is

$$\boldsymbol{z}(x) \leq \sum_{k=0}^{\infty} P^k(x)\, \boldsymbol{u} \; . \tag{7.8}$$

Since B is a closed recurrent class of the Markov chain corresponding to $P(0)$, and since B is reachable with positive probability from s, we have

$$\left[\sum_{k=0}^{\infty} P^k(0)\, \boldsymbol{u}\right]_s = -\infty \; ,$$

where the notation $[\cdot]_s$ indicates that we are considering the component of the vector corresponding to s. From the continuity of $P(x)$ in $x = 0$ follows $\lim_{x \to 0} P^k(x) = P^k(0)$ for all $k \geq 0$; thus,

$$\lim_{x \to 0} \left[\sum_{k=0}^{\infty} P^k(x)\, \boldsymbol{u}\right]_s = -\infty \; ,$$

and hence from (7.8) and (7.7) we have $\lim_{x \to 0} v_s(x) = -\infty$, which concludes our argument. ∎

With this result, we can prove Theorem 7.3 and conclude the proof of Lemma 7.1.

Proof of Theorem 7.3. Consider an instance Π of SSP satisfying SSP Assumptions 3, 6 and 7, and let $\widehat{\Pi}$ be the instance obtained from Π using Algorithm 7.1. By SSP Assumptions 6 and 7, if (B, D) is an end component of $S - U$, it must be $c(s, a) = 0$ for all $(s, a) \in sa(B, D)$. Since these end components are eliminated by Algorithm 7.1, the resulting instance $\widehat{\Pi}$ will not contain any end component in $S - U$. Thus, all policies for $\widehat{\Pi}$ are proper, and SSP Assumptions 1 and 2 hold. The fact that the solutions of the SSP problem $\widehat{\Pi}$ also provide solutions for Π can then be proved by reasoning as in the proof of Theorem 7.1. ∎

Proof of Lemma 7.1. The first direction of the lemma was given in Lemma 7.2. In the other direction, assume that in the instance Π of SSP there is no end component (B, D) of $S - U$ having $c(s, a) < 0$ for some $(s, a) \in sa(B, D)$. The fact that $v_s > -\infty$ at every $s \in S - U$ follows from Theorem 7.3, and from the analysis of the SSP problem $\widehat{\Pi}$ given in Section 3.4. ∎

Theorem 7.4 can then be proved by repeating the argument used for Theorem 7.2.

7.5 The Minimum Expected Total Cost Problem

Even though the operator L can admit more than one fixpoint for non-negative and non-positive instances of SSP problems, these instances can still be solved by a direct translation into linear programming of the equation $\boldsymbol{v} = L\boldsymbol{v}$, as indicated by Theorems 7.2 and 7.4 (albeit a more efficient solution is provided by Theorems 7.1 and 7.3).

In this section, we will describe instead a problem for which the straightforward reduction to linear programming does not work: the *non-negative minimum expected total cost* (non-negative METC) problem. Strauch [Str66] discusses this problem, and Denardo [Den70] proposed reductions to linear programming that finds *bias-optimal* policies for the METC problem without requiring the non-negativity assumption. The proposal of [Den70] requires the solution of three nested linear programming problems.

For the case of the non-negative METC problem we propose an alternative method, which requires the solution of only one linear programming problem. Our method is based on a reduction from the non-negative METC problem to the SSP problem. Aside from linear programming, the reduction enables the use of value and policy iteration methods to solve non-negative METC problems.

The proposed reduction relies once more on the properties of end components. In fact, there is a correspondence between the method of [Den70] and our proposal: the solution of some linear programming problems in [Den70] corresponds to the removal of end components in our approach. A discussion of the relationship between the two methods, however, is beyond the scope of this dissertation.

The results developed in this subsection will not be used in the remainder of this dissertation, and have been included due to their general interest only.

An instance of the METC problem is defined as follows.

Definition 7.1 (minimum expected total cost problem) An instance of the *minimum expected total cost* problem consists of an MDP $\Pi = (S, A, p, c)$, in which c is a cost function assigning to each state-action pair $(s, a) \in \xi_\Pi$ a cost $c(s, a) \in \mathbb{R}$. ∎

The METC problem consists in minimizing the total expected cost, over the set of all policies. For a policy η, we define the *cost* u_s^η of η from $s \in S$ by

$$u_s^\eta = \sum_{k=0}^{\infty} c(X_k, Y_k) \ . \tag{7.9}$$

Note that the cost can be unbounded for some policies and initial states. The METC problem is formally defined as follows.

Definition 7.2 (minimum expected total cost problem) Given an instance $\Pi = (S, A, p, c)$ of METC, determine

$$u_s^* = \inf_\eta u_s^\eta$$

for all $s \in S$. ∎

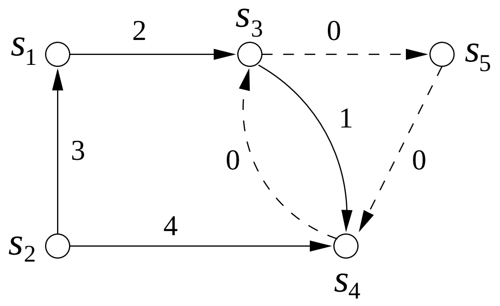

Figure 7.4: An instance of non-negative METC. The actions are all deterministic, and the edges are labeled with their cost. The end component composed by the states $\{s_3, s_4, s_5\}$ along with the zero-cost actions from these states is depicted using dashed lines.

We will provide a solution for the METC problem under the *non-negativity* assumption:

METC Non-Negativity Assumption: For all $(s, a) \in \xi_\Pi$, it is $c(s, a) \geq 0$.

There is also a reduction from non-positive METC to SSP, albeit the reduction is more involved. We will not discuss this reduction here, since the non-positive METC problem can also be solved efficiently with the methods discussed, for example, by Puterman [Put94].

Before presenting our reduction, let us gain some intuition into why the standard approach fails. The Bellman equation for the METC problem can be written as $\boldsymbol{u} = L\boldsymbol{u}$, where $\boldsymbol{u} = [u_s]_{s \in S}$ is the cost vector, and the functional L is defined by

$$[L\boldsymbol{u}]_s = \min_{a \in A(s)} \left[c(s, a) + \sum_{t \in S} p_{st}(a) u_t \right] . \tag{7.10}$$

The operator L may have more than one fixpoint. As discussed in [Put94], $\boldsymbol{u}^*$ is the least non-negative fixpoint of L. The following example illustrates the presence of more than one fixpoint.

Example 7.4 Consider the instance of non-negative METC depicted in Figure 7.4. The operator L defined in (7.10) admits infinitely many fixpoints: two such fixpoints are $\boldsymbol{u}^{(1)}$ and $\boldsymbol{u}^{(2)}$, where

$$u^{(1)}_{s_1} = 5, \quad u^{(1)}_{s_2} = 7, \quad u^{(1)}_{s_3} = 3, \quad u^{(1)}_{s_4} = 3, \quad u^{(1)}_{s_5} = 3$$
$$u^{(2)}_{s_1} = 2, \quad u^{(2)}_{s_2} = 4, \quad u^{(2)}_{s_3} = 0, \quad u^{(2)}_{s_4} = 0, \quad u^{(2)}_{s_5} = 0$$

For this instance of METC, all fixpoints of L can be written in the form $\boldsymbol{u}^{(1)} + x\mathbf{1}$, where $x \in \mathbb{R}$ and $\mathbf{1}$ is the vector composed of $|S|$ 1s. Vector $\boldsymbol{u}^{(2)}$ is the least non-negative fixpoint of L, so that by the arguments discussed in [Put94] it is $\boldsymbol{u}^{(2)} = \boldsymbol{u}^*$. ∎

The problem of finding the least non-negative fixpoint of L cannot be solved directly by linear programming. In fact, the linear programming problem that arises directly from $\boldsymbol{u} = L\boldsymbol{u}$ is:

Maximize $\sum_{s \in S} u_s$ *subject to:*

$$u_s \leq c(s,a) + \sum_{t \in S} p_{st}(a) u_t \ , \qquad\qquad s \in S, a \in A(s) \ .$$

However, instead of finding the least non-negative fixpoint of L, this linear programming problem would find the *largest* such fixpoint. As noted in Example 7.4, this largest fixpoint can be unbounded.

From the example, we also see that the presence of multiple fixpoints is connected with the existence of end components composed of state-action pairs having cost 0. Our analysis of the non-negative METC problem exploits the existence of such end components. Our first result characterizes the set of states on which $u^* < \infty$. Given an instance $\Pi = (S, A, p, c)$ of METC, define $D(s) = \{a \in A(s) \mid c(s,a) = 0\}$, and let

$$S_0 = \bigcup \Big\{ B \ \Big| \ (B,C) \in \mathit{maxEC}(S,D) \Big\} \ .$$

Define also

$$S_{<\infty} = \Big\{ s \in S \ \Big| \ \max_\eta \Pr_s^\eta(\mathit{reach}(S_0)) = 1 \Big\} \ .$$

Theorem 7.6 *The following assertions hold:*

- *If* $s \in S_0$, *then* $v_s^* = 0$.
- $v_s^* < \infty$ *iff* $s \in S_{<\infty}$.

Proof. The first assertion is an immediate consequence of our results on end components.

For the second assertion, assume first that $\max_\eta \Pr_s^\eta(\mathit{reach}(S_0)) < 1$, and consider an arbitrary policy η. Since the behaviors from s with probability 1 eventually follow and end component, there must be an end component (B, C) such that:

- there is $(t,a) \in sa(B,C)$ such that $c(t,a) > 0$;
- $\Pr_s^\eta(\mathit{inft}(\omega) = sa(B,C)) > 0$.

From these two statements, and from the fact that η is arbitrary, it follows immediately that $u_s^* = \infty$.

In the other direction, from our results on the maximum reachability probability problem we know that there is a Markovian policy η_m that maximizes the probability of reaching S_0. Thus, if $s \in S_{<\infty}$ it is $\max_\eta \Pr_s^{\eta_m}(\mathit{reach}(S_0)) = 1$. From η_m, reasoning as in the proof of Lemma 4.1 we can obtain another Markovian policy η_M, which coincides with η_m outside S_0, and that once in S_0 it follows forever one of the end components forming S_0. Let $P = [p_{st}]_{s,t \in S_{<\infty}}$, $\boldsymbol{u} = [u_s^{\eta_M}]_{s \in S_{<\infty}}$, and $\boldsymbol{c} = [c_s]_{s \in S_{<\infty}}$, where

$$p_{st} = \sum_{a \in A(s)} Q_{\eta_M}(a \mid s)\, p_{st}(a) \qquad\qquad c_s = \sum_{a \in A(s)} Q_{\eta_M}(a \mid s) c(s,a) \ .$$

Then, we can write $\boldsymbol{u} = \sum_{k=0}^{\infty} P^k \boldsymbol{c}$. Noting that $c_s = 0$ for $s \in S_0$, and that once S_0 is entered, a behavior never exits from S_0, we can write $\widehat{\boldsymbol{u}} = \sum_{k=0}^{\infty} \widehat{P}^k \widehat{\boldsymbol{c}}$, where $\widehat{\boldsymbol{u}}$, $\widehat{P}$ and $\widehat{\boldsymbol{c}}$ are the restrictions of $\boldsymbol{u}$, P and $\boldsymbol{c}$ to $S_{<\infty} - S_0$. This is equivalent to $\widehat{\boldsymbol{u}} = (I - \widehat{P})^{-1} \widehat{\boldsymbol{c}}$. Since $\widehat{P}$ is the matrix of a transient Markov chain, $\det(I - \widehat{P}) \neq 0$, and we conclude that $\widehat{\boldsymbol{u}}$ is finite. By definition of u_s^*, this implies that $u_s^* < \infty$ for $s \in S_{<\infty}$. ∎

It remains to compute the value of u_s^* at the states $s \in S_{<\infty} - S_0$. This can be done using a straightforward reduction to the stochastic shortest path problem, as stated by the following theorem.

Theorem 7.7 (reduction from non-negative METC to SSP) *Consider the instance* $\widehat{\Pi} = (S_{<\infty}, \widehat{A}, \widehat{p}, S_0, \widehat{c}, g)$ *of SSP problem, where* $\widehat{A}$ *and* $\widehat{p}$ *are the restrictions of* A *and* p *to* $S_{<\infty}$, $\widehat{c}$ *is the restriction of* c *to* $S_{<\infty} - S_0$, *and* g *is identically 0.*

Then, the SSP problem satisfies SSP Assumptions 1 and 2. Moreover, it is $u_s^* = v_s^*$ *for all* $s \in S_{<\infty} - S_0$, *where* v_s^* *denotes the solution of the SSP problem.*

Proof. The fact that SSP Assumption 1 holds is a direct consequence of the definition of $S_{<\infty}$. To see that SSP Assumption 2 holds, we can reason as in the proof of Theorem 7.6 to show that if η is SSP-improper, then $v_s^\eta = \infty$ for some $s \in S_{<\infty}$.

Denote by $\boldsymbol{u}$, $\boldsymbol{v}$ the restrictions to $S_{<\infty} - S_0$ of the vectors corresponding to the cost of the METC and SSP problems, respectively. As can be seen from the proof of Theorem 7.6, it is $\boldsymbol{u}^* = \inf_{\eta \in \boldsymbol{\eta}_P} \boldsymbol{u}^\eta$, i.e. the solution of the METC problem corresponds to an SSP-proper policy. For $\eta \in \boldsymbol{\eta}_P$, it is $\boldsymbol{u}^\eta \geq \boldsymbol{v}^\eta$, since in the definition of METC, also the cost incurred after leaving the end components in S_0 is taken into account. Hence,

$$\boldsymbol{u}^* = \inf_{\eta \in \boldsymbol{\eta}_P} \boldsymbol{u}^\eta \geq \inf_{\eta \in \boldsymbol{\eta}_P} \boldsymbol{v}^\eta = \boldsymbol{v}^* \ . \tag{7.11}$$

In the other direction, if η is SSP-proper we can obtain another SSP-proper policy η' that differs from η only on S_0, and such that η' follows forever one of the end components of S_0, once S_0 is entered. For this policy, it is $\boldsymbol{v}^\eta = \boldsymbol{v}^{\eta'} = \boldsymbol{u}^{\eta'}$. Hence,

$$\boldsymbol{u}^* = \inf_{\eta \in \boldsymbol{\eta}_P} \boldsymbol{u}^\eta \leq \inf_{\eta \in \boldsymbol{\eta}_P} \boldsymbol{v}^\eta = \boldsymbol{v}^* \ . \tag{7.12}$$

The result follows by combining (7.11) and (7.12). ∎

Chapter 8

Fairness

Fairness is a concept that has been widely used in the formal modeling and verification of systems; its traditional uses include the modeling of concurrency and of unbiased arbitration. In the context of probabilistic systems, fairness serves also the additional purposes of abstracting from the specific values of some transition probabilities, and of modeling transitions with an unspecified distribution of waiting times. The introduction of fairness in the system model requires the use of updated specification languages and model-checking algorithms, which will be described in this chapter.

The subject of fairness in probabilistic systems has been already discussed in the literature. Pnueli [Pnu83] and Pnueli and Zuck [PZ93] introduce the notions of *extreme fairness* and *α-fairness* to abstract from the precise values of probabilities. Hart, Sharir and Pnueli [HSP82] and Vardi [Var85] consider probabilistic systems in which the choice of actions at the states is subject to fairness requirements, and present algorithms for checking that temporal logic formulas hold with probability 1 over these systems. This chapter has been inspired by Baier and Kwiatkowska [KB96], which consider models very similar to Markov decision processes with fairness requirements, and present algorithms to compute the probability with which temporal logic formulas hold over these systems.

We return on the subject of fairness in probabilistic systems for three reasons. The first is to introduce a notion of fairness that is better suited to the study of infinite-state systems and to the modeling of transitions with unknown delay distributions. The second is to propose a modification to the definition of the logics GPTL and GPTL* that leads to simpler model-checking algorithms. The third reason is to provide model-checking algorithms for operators D, $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$, which have been introduced in this dissertation.

We begin our presentation by briefly recalling a few notions of fairness that have been presented in the literature, discussing the implications of each definition in the context of probabilistic systems. We then introduce our version of fairness, called *probabilistic fairness,* and we present the temporal logics FPTL and FPTL*, derived from GPTL and GPTL*. In these new logics, the semantics of the path quantifiers and of the probabilistic operators is modified to take into account the presence of fairness. Finally, we describe the model-checking algorithm for FPTL and FPTL*, followed by its correctness proof.

8.1 An Overview of Fairness

The concept of fairness in the context of formal system verification was introduced by Lehmann, Pnueli and Stavi [LPS81] for shared-variables programs, and by Queille and Sifakis [QS83] for transition systems. Its traditional uses include the modeling of concurrency and of un-biased arbitration; among the monographs devoted to the study of fairness, we recall Francez [Fra86] and Apt, Francez and Katz [AFK88].

Two notions of fairness are commonly considered: *justice,* also called *weak fairness,* and *compassion,* also called *strong fairness* (see Manna and Pnueli [MP91, MP95]). Informally, the meaning of justice and compassion can be described as follows. If τ is a just or compassionate transition, then every behavior of the system must obey the following constraints:

- *τ is just.* If τ is forever enabled, it must be taken infinitely often.
- *τ is compassionate.* If τ is infinitely often enabled, it must be taken infinitely often.

Justice is mainly used to model the progress of individual threads of concurrent computation. If all the transitions of a thread are just, then the execution of the thread must continue as long as at least one of its transitions is enabled. Compassion is used to model fair choice, i.e. to model probabilistic choice abstracting from the numerical values of the probabilities; it can be used in the modeling of arbiters and message-passing systems.

In this dissertation, we will focus on probabilistic notions of fairness related to compassion, and we will not propose methods for dealing with justice. In fact, compassion is better suited to model unspecified transition probabilities and unknown delay distributions, which are our motivations for introducing fairness. We also remark that justice can be often substituted by the stronger requirement of compassion, and that it would be possible to extend our definitions and algorithms to encompass also the notion of justice.

8.2 Probabilistic Fairness

In a fair transition system, the fairness requirements are represented by specifying which transitions are just and compassionate [MP91]. In the system model of timed probabilistic systems, the role of transitions is played by *actions.* Differently from transition systems, however, we do not specify fairness by a global set of fair actions: instead, we associate to each system state a set of actions that are fair at that state. This approach is consistent with our desire to de-emphasize action names: the

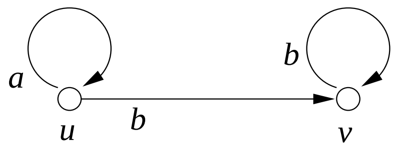

Figure 8.1: A fair Markov decision process $\Pi = (S, A, p, \mathcal{F})$, with $S = \{u, v\}$, $A(u) = \{a, b\}$, $A(v) = \{b\}$, $\mathcal{F}(u) = \{a, b\}$, $\mathcal{F}(b) = \emptyset$. The probability distributions of the state-action pairs are deterministic.

specification of a global set of fair actions would assume an implicit correspondence between actions at different states that have the same name. The definition of *fair Markov decision process* is as follows.

Definition 8.1 (fair Markov decision process) A *fair Markov decision process* (FMDP) $\Pi = (S, A, p, \mathcal{F})$ is a Markov decision process (S, A, p) with an additional labeling $\mathcal{F}$ that specifies for each $s \in S$ a subset $\mathcal{F}(s) \subseteq A(s)$ of *fair* actions at s. ∎

A *fair TPS* (FTPS) is obtained from a TPS by adding the fairness labeling $\mathcal{F}$, defined as for fair MDPs.

Definition 8.2 (fair TPS) A *fair TPS* $\Pi = (S, A, p, \mathcal{F}, S_{in}, time, \mathcal{I})$ consists of a TPS $(S, A, p, S_{in}, time, \mathcal{I})$ with an additional labeling $\mathcal{F}$ that is defined as for fair Markov decision processes. ∎

The fairness of an FTPS is enforced by defining a set of *fair policies.* Roughly, a policy is fair if there is a global non-zero lower bound to the probability with which the fair actions are chosen. To enforce the fairness requirements of a FMDP, we restrict our attention to fair policies in our definition of logic and system semantics. This approach is closely related to that of [KB96], even though our definition of fair policy is different; we will later present a comparison of the two definitions.

Definition 8.3 (fair policy) A policy η for an FMDP $\Pi = (S, A, p, \mathcal{F})$ is fair if there is $\epsilon > 0$ such that, for all $n \geq 0$, all sequences of states $s_0, \ldots, s_n$ and all $a \in \mathcal{F}(s_n)$ it is $Q_\eta(a \mid s_0 \cdots s_n) \geq \epsilon$. ∎

In the above definition, ϵ can depend on the policy η, but cannot depend on the past sequence $s_0 \cdots s_n$ of states, nor on the action $a \in \mathcal{F}(s_n)$. Thus, ϵ represents a lower bound for the probability with which η chooses any fair action throughout the behavior of the system. The following example justifies this definition.

Example 8.1 (past independence of ϵ) Consider the FMDP depicted in Figure 8.1, and assume that we adopt the following alternative definition of fair policy:

> *A policy η for $\Pi = (S, A, p, \mathcal{F})$ is fair if, for all $n \geq 0$, all sequences of states $s_0, \ldots, s_n$ and all $a \in \mathcal{F}(s_n)$, it is $Q_\eta(a \mid s_0 \cdots s_n) > 0$.*

Consider then the policy $\eta^\bullet$ that, after the sequence $s_0, \ldots, s_n$, chooses the next action as follows:

- if $s_n = v$, then $\eta^\bullet$ chooses deterministically action b;
- if $s_n = u$, then $\eta^\bullet$ chooses a with probability $p_n = \exp(-(2^{-n-1}) \log 2)$, and b with probability $1 - p_n$.

According to the modified definition, policy $\eta^\bullet$ is fair, since the probability of choosing a and b at u are always strictly positive. However, it is easy to check that a behavior that starts from $s_0 = u$ will remain forever at u with probability $1/2$. This clearly contradicts our intuitive notion of fair choice between a and b, since a non-zero fraction of behaviors never chooses b, even though such a choice is always possible.

To avoid this type of situation, Definition 8.3 requires that the lower bound to the probability of choosing a fair action is independent from the sequence of past states. ∎

To gain a better understanding of the properties of our definition of fairness, we compare it with other notions of fairness that have been proposed in the literature. To facilitate the comparison, we have reformulated all definitions in terms of Markov decision processes, instead of introducing *fair transition systems,* as in Manna and Pnueli [MP91], or *probabilistic automata,* as in Kwiatkowska and Baier [KB96].

8.2.1 Compassion: Fair Transition Systems Approach

Consider an MDP (S, A, p), and let *Acts* be the underlying set of actions. Adapting the definition of [MP91] to the notation of Markov decision processes, we represent *FTS-compassion* by a set $\mathcal{C} \subseteq \mathit{Acts}$ of actions. Given a behavior $\omega : s_0, a_0, s_1, a_1, s_2, \ldots$, we say that ω is *FTS-fair* if the following condition holds for every $a \in \mathcal{C}$:

> *if $a \in A(s_i)$ for infinitely many $i \geq 0$, then $a = a_i$ for infinitely many $i \geq 0$.*

Since the definition of [MP91] is not aimed at probabilistic systems, it does not specify how the fairness of behaviors should be reflected in the set of policies that are considered. Two alternatives appear to be the most natural, and have been explored by [KB96] in a similar setting:

- Only policies under which all behaviors are fair are considered.
- Only policies under which behaviors are fair with probability 1 are considered.

Clearly, the first alternative leads to a stronger definition of fairness than the second one. The following example illustrates why the notion of FTS-compassion is not appropriate for our purpose of abstracting from the value of transition probabilities, regardless of which one of the above alternatives is chosen.

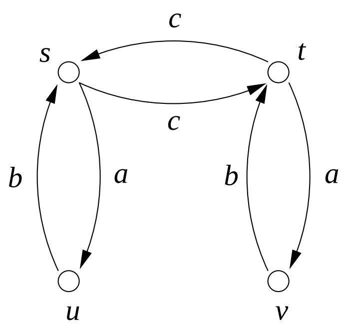

Figure 8.2: A Markov decision process $\Pi = (S, A, p)$ with $S = \{s, t, u, v\}$, and action sets as depicted. The probability distributions associated to all state-action sets are deterministic. To this MDP is associated a set of compassionate actions $\mathcal{C} = \{a, b, c\}$.

Example 8.2 Consider the MDP depicted in Figure 8.2, and consider the policy η defined by:

$$\begin{aligned} Q_\eta(c \mid s) &= 1 & Q_\eta(c \mid \sigma ubs) &= 1 \\ Q_\eta(c \mid \sigma t) &= 1 & Q_\eta(a \mid \sigma tcs) &= 1 \\ Q_\eta(b \mid \sigma u) &= 1 & Q_\eta(b \mid \sigma v) &= 1 \ , \end{aligned}$$

where σ denotes any (possibly empty) sequence of alternated states and actions, and all other conditional probabilities are 0. Intuitively, η chooses always b, b, c respectively at u, v, t; at s, it alternately chooses a or c. Under η, every behavior is compassionate with respect to all actions, since every action appears infinitely often along every behavior. Nonetheless, action a is never chosen at state t, indicating how this definition of compassion fails to capture fully the behavior of probabilistic choice. ∎

Extreme Fairness and α-Fairness

To remedy to the situation illustrated by the previous example, Pnueli [Pnu83] and Pnueli and Zuck [PZ93] introduce the notions of *extreme fairness* and *α-fairness.* Similar concerns also led to the concept of *uniform compassion* in Bjørner, Lerner and Manna [BLM97].

The notion of α-fairness relies on the use of *past temporal logic.* Given a behavior $\omega = s_0 a_0 s_1 a_1 s_2 \cdots$ and a past temporal formula ϕ, we say that $i \geq 0$ is a *ϕ-position* of ω if ϕ holds at position i of ω. We say that an action a is taken at position $i \geq 0$ of ω if $a_i = a$.

A behavior ω is α-fair with respect to ϕ if, whenever a fair action a is enabled at infinitely many ϕ-positions, a is chosen infinitely often from ϕ-positions. A behavior is α-fair if it is α-fair with respect to all past temporal formulas.

This definition prevents situations such as the one discussed in Example 8.2 from arising. We

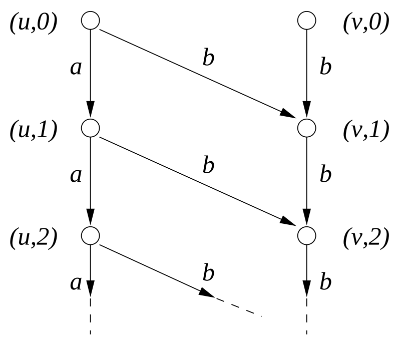

Figure 8.3: Markov decision process resulting from the "unrolling" of the process of Figure 8.1.

have preferred to base our results on the notion of *probabilistic fairness* rather than α-*fairness* because the definition of probabilistic fairness does not require the introduction of past temporal formulas, and because probabilistic fairness, being already stated in terms of probability, leads to a simpler analysis of the model-checking algorithms.

8.2.2 State-Based Fairness

Baier and Kwiatkowska [KB96] propose a different solution to the problem illustrated in Example 8.2. According to [KB96], a behavior ω is fair if, whenever a state s appears infinitely often along ω, all actions in $A(s)$ also appear infinitely often along ω. This notion of fairness has its roots in the *state-fairness* described in Pnueli [Pnu83] and Vardi [Var85]. As mentioned in [KB96], this definition can be easily generalized to the case in which to each state s is associated a set of fair actions $\mathcal{F}(s)$. A policy is *strictly fair* if under it all the behaviors are fair, and is *(almost) fair* if under it the behaviors are fair with probability 1.

The drawback of this definition is that it does not directly apply to infinite-state systems, as the following example indicates.

Example 8.3 Consider the FMDP depicted in Figure 8.3. This FMDP can be obtained by taking the product of the FMDP of Example 8.1 with an automaton over the natural numbers whose state increases by one at each transition. Since no state of this MDP appears more than once in any behavior, the fairness notion of [KB96] does not impose any constraint on the set of policies for it.

This example indicates that the definition of [KB96] is not invariant with respect to the product with infinite-state systems. ∎

The notion of probabilistic fairness retains most of the properties of the proposal of [KB96] while being readily extendible to infinite-state systems. Moreover, probabilistic fairness is suited to the representation of transitions with unknown delay distributions, as we will see in the next chapter.

8.3 Time Divergence and Admissible Policies

In Chapter 3 we introduced the concept of *non-Zeno* TPSs: informally, a TPS is non-Zeno if time diverges with probability 1 under any policy. In the previous chapters, we required that the TPSs under consideration be non-Zeno, thus ensuring time divergence regardless of the policy.

While this approach leads to simpler logics and verification algorithms, is not adequate for the study of system approximations. A TPS that is constructed as the approximation of a system may not be non-Zeno, since the substitution of probability with nondeterminism can lead to policies under which time diverges with probability less than 1.

For these reasons, we adopt in the following a more general approach, due to Segala [Seg95b]. This approach is based on the notion of *time-divergent policies* (called *admissible* in [Seg95b]): a policy is *time divergent* if time diverges with probability 1 under it. While the non-Zenoness requirement asks for all policies to be time divergent, the proposal of [Seg95b] is to consider the behavior of the system only under time-divergent policies.

The advantage of the approach based on non-Zenoness is its simplicity: since in a non-Zeno TPS all policies lead to time divergence, it is possible to define the meaning of the probabilistic operators by quantifying over the set of all policies, without distinguishing among time divergent and non-divergent ones. However, our approach to fairness has already led us to restrict the set of policies under consideration, so that adopting the approach of [Seg95b] does not complicate excessively the semantics of the logic or the presentation of the model-checking algorithms. For this reason, in the study of fair TPSs we adopt this approach, and we define time divergent policies as follows.

Definition 8.4 (time-divergent policy) Given an FTPS $(S, A, p, \mathcal{F}, S_{in}, time, \mathcal{I})$, we say that a policy η is *time divergent* for a state s if

$$\Pr_s^\eta\Big(\sum_{k=0}^{\infty} time(X_k, Y_k) = \infty\Big) = 1 . \quad \blacksquare$$

For each state s we define the set $Adm(s)$ of *admissible policies* for s as the set of policies that are both fair and time divergent. This is the set of policies that will be considered in the logics and verification algorithms for fair TPSs.

Definition 8.5 (admissible policy) We say that a policy η is *admissible* for a state s if η is both fair and time divergent for s. The set of admissible policies for a state s is denoted by $Adm(s)$. We say that a policy η is *admissible* if it is admissible for all states. $\blacksquare$

Finally, we substitute Assumption 3.1 with the weaker assumption stating that every state has at least one admissible policy.

Assumption 8.1 (admissibility) Given a fair TPS Π, for every state s of Π it is $Adm(s) \neq \emptyset$.

8.4 Probabilistic Logics for Fairness: FPTL and FPTL*

As a consequence of the introduction of fairness and of the new approach to time divergence, it is necessary to provide new definitions for the semantics of the probabilistic temporal logics. The new logics, called FPTL and FPTL* (from *Fair Probabilistic Temporal Logics*) are obtained from GPTL and GPTL* by modifying the semantics of the path quantifiers A, E and of the probabilistic operators P, D, $\bar{\text{P}}$, $\bar{\text{D}}$. The rationale for the modifications is as follows.

8.4.1 Path Quantifiers A, E

In GPTL or GPTL*, formula Aϕ holds at a state s if all behaviors originating from s satisfy ϕ. Thus, the path quantifiers A and E are defined as in CTL and CTL*.

This definition is not compatible with our approach to fairness. In fact, even if policy η is fair and time divergent for s, some of the behaviors arising under η may not be fair, in the sense that they may contain infinitely many states where a fair action is possible, and contain only finitely many occurrences of that action. The behavior of the system under such unfair behaviors is taken into account by the classical definition of the path quantifiers, preventing the use of these quantifiers in specifications in which the fairness of the system matters.

The problem can be traced to the fact that, according to our definition, fairness is a restriction on the probability distributions with which policies choose the actions, rather than a restriction on the sequence of actions along every behavior. Hence, we are not able to guarantee that all behaviors arising from a fair policy are fair in the classical sense: all we can guarantee is that they are fair with probability 1. To remedy to this situation, we redefine the semantics of the path quantifiers A and E in FPTL and FPTL*. According to the our new definitions, formula Aϕ holds at a state s if, under any admissible policy, ϕ holds with probability 1; the path quantifier E is then defined as the dual of A, as usual.

8.4.2 Operators P, D, $\bar{\text{P}}$, $\bar{\text{D}}$.

To simplify the model-checking algorithms, we adopt a new semantics for the probabilistic operators P, D, $\bar{\text{P}}$, and $\bar{\text{D}}$. We motivate this choice by presenting the argument relative to the operator P; similar arguments apply to the other operators.

Given a state s and a sequence formula ϕ, denote by

$$P(s,\phi) = \left\{ x \in [0,1] \;\middle|\; \exists \eta \in Adm(s) \,.\, \Pr_s^{\eta}(\omega \models \phi) = x \right\}$$

the set of values that the probability of ϕ can assume under admissible policies. An analysis of the algorithms presented in Baier and Kwiatkowska [KB96] (which are based on a different notion of fairness) reveals that determining $\sup P(s,\phi)$ and $\inf P(s,\phi)$ can be done in time polynomial in the size of the TPS. Determining whether $P(s, \phi)$ is open or closed at the endpoints, i.e. whether $\inf P(s, \phi) \in P(s, \phi)$ and $\sup P(s, \phi) \in P(s, \phi)$, requires more involved algorithms.

If $\sup P(s, \phi) = a$, determining whether $P(s, \phi)$ is open or closed at the endpoints enables to distinguish between the case in which $\mathrm{P}_{<a}\phi$ holds at s, and the case in which only $\mathrm{P}_{\leq a}\phi$ holds. We note that the need to distinguish among these cases does not arise in the analysis of non-fair TPSs, since the sets under consideration are closed at the endpoints in the absence of fairness. This is a consequence of the existence of optimal policies that minimize or maximize the quantities measured by P, D, $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$, a result that has been proved in the previous chapters.

Even in the case of fair TPSs, however, the practical usefulness of such a fine distinction is open to question, especially in view of fact that the transition probabilities of a system are known in many cases only in an empirical fashion. In the logics FPTL and FPTL* we do not distinguish between the cases in which $P(s, \phi)$ is open or closed at the endpoints, and we base our definitions on the supremum and infimum of the set $P(s, \phi)$ instead. This leads to simpler model-checking algorithms.

8.4.3 Semantics of FPTL and FPTL*

To define the semantics of FPTL and FPTL*, we need to provide new definitions for the product of a fair TPS and experiment, and for threshold outcomes. The product is defined as follows.

Definition 8.6 (product of fair TPS and experiments) Given a fair TPS $\Pi = (S, A, p, \mathcal{F}, S_{in}, \mathit{time}, \mathcal{I})$ and an experiment $\Psi = (V, E, E_r, V_{in}, \lambda)$, their *product FMDP* $\Pi_\Psi = \Pi \otimes \Psi = (\widehat{S}, \widehat{A}, \widehat{p}, \widehat{\mathcal{F}}, \widehat{\mathit{time}}, r, w)$ is defined as in Definition 5.5, with the addition that

$$\widehat{\mathcal{F}}(\langle s, v\rangle) = \mathcal{F}(s) \qquad\qquad \widehat{\mathit{time}}(\langle s, v\rangle, a) = \mathit{time}(s, a)$$

for all $\langle s, v\rangle \in \widehat{S}$ and $a \in \widehat{A}(\langle s, v\rangle)$. ∎

Admissible threshold outcomes represent the minimum and maximum values for the long-run average outcome of an experiment that can be attained with positive probability under an admissible policy: they are the corresponding concept to threshold outcomes in presence of fairness (see Definition 5.9).

Definition 8.7 (admissible threshold outcomes) Consider a fair TPS Π, an experiment Ψ and their product FMDP Π_Ψ. For each state $\langle s, v\rangle$ of Π_Ψ, we define the *maximum* and *minimum threshold outcomes* $\bar{\mathrm{FT}}^+_{\langle s,v\rangle}$ and $\bar{\mathrm{T}}^-_{\langle s,v\rangle}$ by

$$\bar{\mathrm{FT}}^+_{\langle s,v\rangle} = \sup\Big\{a \in \mathbb{R} \;\Big|\; \exists \eta \in \mathit{Adm}(\langle s, v\rangle) \,.\, \Pr^\eta_{\langle s,v\rangle}\Big(I \wedge \limsup_{n\to\infty} \mathcal{H}_n(\omega) \geq a\Big) > 0\Big\}$$

$$\bar{\mathrm{FT}}^-_{\langle s,v\rangle} = \inf\Big\{a \in \mathbb{R} \;\Big|\; \exists \eta \in \mathit{Adm}(\langle s, v\rangle) \,.\, \Pr^\eta_{\langle s,v\rangle}\Big(I \wedge \liminf_{n\to\infty} \mathcal{H}_n(\omega) \leq a\Big) > 0\Big\} \, ,$$

	GPTL, GPTL*	FPTL, FPTL*
$s \models \mathrm{A}\phi$	$\forall\omega \in \Omega_s \,.\, \omega \models \phi$	$\forall\eta \in \mathit{Adm}(s) \,.\, \mathrm{Pr}_s^\eta(\omega \models \phi) = 1$
$s \models \mathrm{E}\phi$	$\exists\omega \in \Omega_s \,.\, \omega \models \phi$	$\exists\eta \in \mathit{Adm}(s) \,.\, \mathrm{Pr}_s^\eta(\omega \models \phi) > 0$
$s \models \mathrm{P}_{\bowtie a}\phi$	$\forall\eta \,.\, \mathrm{Pr}_s^\eta(\omega \models \phi) \bowtie a$	$\left[\inf_{\eta \in \mathit{Adm}(s)} \mathrm{Pr}_s^\eta(\omega \models \phi)\right] \bowtie a$
$s \models \mathrm{D}_{\bowtie a}\psi$	$\forall\eta \,.\, \mathrm{E}_s^\eta\{\mathit{timeto}(\psi)\} \bowtie a$	$\left[\inf_{\eta \in \mathit{Adm}(s)} \mathrm{E}_s^\eta\{\mathit{timeto}(\psi)\}\right] \bowtie a$
$s \models \bar{\mathrm{P}}_{\bowtie a}\Psi$	$\forall\eta \,.\, \mathrm{Pr}_{\langle s, V_{in}\rangle}^\eta\big(I \to (\limsup_{n\to\infty} \mathcal{H}_n \bowtie a)\big)$	$\bar{\mathrm{FT}}_s^-(\Psi) \bowtie a$
$s \models \bar{\mathrm{D}}_{\bowtie a}\Psi$	$\forall\eta \,.\, \mathrm{Pr}_{\langle s, V_{in}\rangle}^\eta\big(I \to (\limsup_{n\to\infty} \mathcal{H}_n \bowtie a)\big)$	$\bar{\mathrm{FT}}_s^-(\Psi) \bowtie a$

Table 8.1: Semantics of the path quantifiers and probabilistic operators in GPTL, GPTL* and FPTL, FPTL*. The satisfaction relations in the left column holds if and only if the statements in the two other columns hold. In the table, ϕ denotes a sequence formula, ψ a state formula, and Ψ an experiment. The formulas I and $\mathcal{H}$ appearing in the rows for $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ refer to the product with experiment Ψ, and V_{in} is the initial vertex of Φ. The symbol a denotes either a non-negative real number (operators D, $\bar{\mathrm{D}}$) or a real number in the interval $[0, 1]$ (operators P, $\bar{\mathrm{P}}$). The symbol $\bowtie$ denotes one of $\geq, >$: the definitions for $\leq, <$ are analogous.

with the convention that, if $\mathrm{Pr}_{\langle s,v\rangle}^\eta(I) = 0$ for all $\eta \in \mathit{Adm}(\langle s, v\rangle)$, then $\bar{\mathrm{FT}}_{\langle s,v\rangle}^+ = -\infty$, $\bar{\mathrm{FT}}_{\langle s,v\rangle}^- = +\infty$. For a state s of Π we define then

$$\bar{\mathrm{FT}}_s^+(\Psi) = \bar{\mathrm{FT}}_{v_{in}(s)}^+ \qquad \bar{\mathrm{FT}}_s^-(\Psi) = \bar{\mathrm{FT}}_{v_{in}(s)}^- . \qquad \blacksquare$$

Table 8.1 presents the definitions of the path quantifiers and probabilistic operators in FPTL and FPTL*. The table also gives the corresponding definitions in GPTL and GPTL*, for comparison. The table uses the abbreviation

$$\mathit{timeto}(\psi) = \sum_{k=0}^{T_\psi - 1} \mathit{time}(X_k, Y_k) \,. \tag{8.1}$$

As usual, we say that a fair TPS satisfies a FPTL or FPTL* specification $\phi \in \mathit{Stat}$ if all initial states of the TPS satisfy ϕ. The following definition is the exact counterpart of Definition 4.1

Definition 8.8 (validity over a fair TPS) Given a fair TPS $\Pi = (S, A, p, \mathcal{F}, S_{in}, \mathit{time}, \mathcal{I})$ and an FPTL or FPTL* specification $\phi \in \mathit{Stat}$, we say that Π *satisfies* ϕ, written $\Pi \models \phi$, iff $s \models \phi$ for all $s \in S_{in}$. $\blacksquare$

The following results states that operators A and E are the dual of each other in FPTL and FPTL*. The result for GPTL and GPTL* was a consequence of the well-known result for CTL and CTL*.

Theorem 8.1 (duality of path operators in FPTL, FPTL*) *Given a state s and a sequence formula ϕ of FPTL or FPTL*, it is $s \models \mathrm{A}\phi$ iff $s \not\models \mathrm{E}\neg\phi$, where the satisfaction relation $\models$ is as defined for FPTL and FPTL*.*

8.5 Model-Checking Algorithms for FPTL and FPTL*

As usual, the model-checking problem for FPTL and FPTL* consists in determining the truth-value of a given state formula ϕ at all states of the system. As described in Chapter 4, this problem can be solved by recursively evaluating the values of the state subformulas of ϕ at all states of the system. Since the only difference between FPTL, FPTL* and GPTL, GPTL* lies in the definition of the path quantifiers and the probabilistic operators, we need to provide model-checking algorithms only for these operators. We present the model-checking algorithms for FPTL* only, since FPTL is a subset of FPTL*.

We present the model-checking algorithms in this section, and their correctness proof in the next. Before we proceed to stating the algorithms, we need some preliminary notions.

8.5.1 Admissible End Components

The concept of end components played a major role in the design and analysis of the model-checking algorithms for GPTL and GPTL*. As explained in Chapter 3, this concept provided a characterization of the set of state-action pairs that could be repeated forever along a behavior with positive probability. With the introduction of fairness in the system model and with the adoptions of the approach of [Seg95b] to time divergence, we must revise this notion, and introduce *admissible end components* (AECs). Admissible end components play the same role in the study of fair MDPs as end components in the study of ordinary MDPs; they are defined as follows.

Definition 8.9 (admissible end components) Given a fair MDP $\Pi = (S, A, p, \mathcal{F})$, an end component (B, C) of Π is *admissible end component* (AEC) if it satisfies the following two conditions:

- *(fairness)* for all $s \in B$ it is $\mathcal{F}(s) \subseteq C(s)$;
- *(time divergence)* there is $(s, a) \in sa(B, C)$ such that $time(s, a) > 0$.

Maximality and containment for AECs are defined as for end components. ∎

Again, we introduce a shorthand to denote the set of AECs contained in a given sub-MDP.

Definition 8.10 (*maxAEC*) Given a sub-MDP (B, C) of a fair MDP, we denote by $maxAEC(B, C) = \{(D_1, E_1), \ldots, (D_n, e_n)\}$ the set consisting of the maximal admissible end components of (B, C). Moreover, we abbreviate $maxAEC(B, A_{\setminus B})$ by $maxAEC(B)$. ∎

Given a sub-MDP (C, D), the set of maximal admissible end components of (C, D) can be computed using the following algorithm.

Algorithm 8.1 (computation of maximal admissible end components)

Input: A sub-MDP (C, D).

Output: The set $\mathcal{L} = maxAEC(C, D)$.

Initialization: $\mathcal{J} := \{(C, D)\}$.

Repeat the following steps:

- Select $(B, E) \in \mathcal{J}$.
- For all $s \in B$, let $E'(s) := \{a \in E(s) \mid \mathrm{Succ}(s, a) \subseteq B\}$, and let

$$E''(s) := \begin{cases} E'(s) & \text{if } \mathcal{F}(s) \subseteq E'(s); \\ \emptyset & \text{otherwise.} \end{cases}$$

- Let $B_1, \ldots, B_n$ be the strongly connected components of the graph $(B, \rho_{(B,E'')})$, and let $E_i(s) = E''(s)$ for all $i \in [1..n]$ and $s \in B_i$.
- Replace $(B, E) \in \mathcal{J}$ with $(B_1, E_1), \ldots, (B_n, E_n)$.

Until: $\mathcal{J}$ cannot be changed by the above iteration.

Return: $\mathcal{L} = \{(B, E) \in \mathcal{J} \mid \exists (s, a) \in sa(B, E) \,.\, time(s, a) > 0\}$. ∎

The following two theorems provide a characterization of admissible end components, and are the counterparts of Theorems 3.1 and 3.2 of Chapter 3.

Theorem 8.2 (stability of AECs) *Let (C, D) be an admissible end component. Then, for every admissible policy η there is an admissible policy η' such that:*

$$\Pr_s^{\eta}(reach(C)) \;=\; \Pr_s^{\eta'}(reach(C)) \;=\; \Pr_s^{\eta'}(inft(\omega) = sa(C, D)) \tag{8.2}$$

for all $s \in S$.

Proof. Policy η' can be constructed from η as follows. After the past $s_0 \cdots s_n$:

- if $s_n \notin C$, then $Q_{\eta'}(a \mid s_0 \cdots s_n) = Q_{\eta}(a \mid s_0 \cdots s_n)$ for all $a \in A(s)$;

- if $s_n \in C$, then $Q_{\eta'}(a \mid s_0 \cdots s_n)$ is equal to $|D(s)|^{-1}$ if $a \in D(s)$, and is equal to 0 otherwise.

It is easy to verify that η' is admissible, and that (8.2) holds for η and η'. ∎

The following theorem states that, for any initial state and admissible policy, a behavior will eventually follow with probability 1 an admissible end component.

Theorem 8.3 (fundamental theorem of admissible end components) *For any $s \in S$ and any admissible policy η, $\Pr_s^\eta(sub(inft(\omega))$ is an AEC$) = 1$.*

Proof. The proof of this theorem follows the idea of the proof of Theorem 3.2. First, note that with probability 1 the sub-MDP $sub(inft(\omega))$ is an end component. Since there are only countably many end components, it remaints to be proved that if $\Pr_s^\eta(sub(inft(\omega)) = (C, D)) > 0$ for an end component (C, D), then (C, D) is admissible. Assume thus that η is admissible, and that $\Pr_s^\eta(sub(inft(\omega)) = (C, D)) > 0$.

Since η is admissible, there is a lower bound ϵ to the probability of choosing an action in $\mathcal{F}(s)$ at every state s. Thus, by reasoning in analogy with the proof of Theorem 3.2 it can be proved that $\mathcal{F}(s) \subseteq D(s)$ for all $s \in C$, which is our first requirement for (C, D) to be admissible. It is then immediate to see that there must be at least one state-action pair $(s, a) \in sa(C, D)$ such that $time(s, a) > 0$, due to the time-divergence requirement. This is the second requirement for (C, D) to be admissible, and the proof is concluded. ∎

8.5.2 Existence of Admissible Policies

The first task of the model-checking algorithms is to verify that Assumption 8.1 holds. We present two algorithms. The first algorithm checks whether there is at least one admissible policy for every state. The second algorithm enables to obtain, given a TPS Π, the maximal sub-TPS Π' of Π that satisfies Assumption 8.1. The second algorithm is not necessary for model checking, but it can be used in a pre-processing step to obtain a TPS that can be model-checked.

Algorithm 8.2 (checking Assumption 8.1)

Input: A fair TPS $\Pi = (S, A, p, \mathcal{F}, S_{in}, time, \mathcal{I})$.

Output: Yes/no answer indicating whether Assumption 8.1 holds for Π.

Method: Let $G = \{s \in S \mid \exists a \in A(s) \,.\, time(s, a) > 0\}$ be the set of states from which time can advance. Answer "yes" if every state of S can reach some state of G in the graph (S, ρ_S), and "no" otherwise. ∎

Algorithm 8.3 (computing the sub-TPS that satisfy Assumption 8.1)

Input: A fair TPS $\Pi = (S, A, p, \mathcal{F}, S_{in}, \mathit{time}, \mathcal{I})$.

Output: The (possibly empty) maximal sub-TPS Π' of Π that satisfies Assumption 8.1.

Method:

- Let $S' = S$, $A' = A$.
- Repeat:
  - Let $G = \{s \in S' \mid \exists a \in A'(s) \,.\, \mathit{time}(s, a) > 0\}$.
  - Let B' be the subset of states of S' that cannot reach G in $(S', \rho_{(S',A')})$.
  - Remove from S' all states in B'.
  - Repeat:
    - Remove from S' all states $s \notin S_\phi$ that have a fair action leading to at least one removed state.
    - Remove from A' all the actions leading to at least one removed state.
    - Remove from S' all states $s \notin S_\phi$ that have no actions left in A'.
  - Until no more states or actions can be removed from S', A'.
- Until no more states or actions can be removed from S', A'.
- Let $\mathcal{F}'$, time', $\mathcal{I}'$ be the restrictions of $\mathcal{F}$, time, $\mathcal{I}$ to S', A', and output the fair TPS $\Pi' = (S', A', p', \mathcal{F}', S' \cap S_{in}, \mathit{time}', \mathcal{I}')$. ∎

Note that the fair TPS Π' needs not be connected, nor it needs to contain any initial state. These conditions can be tested after the application of Algorithm 8.3.

8.5.3 Path Quantifiers

Given a fair TPS $\Pi = (S, A, p, \mathcal{F}, S_{in}, \mathit{time}, \mathcal{I})$ and a sequence formula ϕ, we now present algorithms to decide whether $s \models \mathrm{A}\phi$ and $s \models \mathrm{E}\phi$. As in Section 4.4, we assume that the value of the maximal state subformulas $\alpha_1, \ldots, \alpha_n$ of ϕ has already been evaluated at all states of the TPS. Again, we let $\phi' = [r_1/\alpha_1] \ldots [r_n/\alpha_n]$ to be the result of replacing each α_i with a new propositional symbol r_i, for every $t \in S$ we define its label $l(t)$ by $l(t) = \{r_i \mid 1 \le i \le n \wedge t \models \alpha_i\}$.

Let $\Pi_{\neg\phi'} = \Pi \otimes DR_{\neg\phi'}$ be the MDP obtained by taking the product between the l-labeled TPS Π and the deterministic Rabin automaton $DR_{\neg\phi'}$ for $\neg\phi'$. This product is computed as in Algorithm 4.2. For each state $s = \langle t, v \rangle$ of Π' corresponding to states t of Π and v of $DR_{\neg\phi'}$, we let $\mathcal{F}'(s) := \mathcal{F}(t)$, and for $a \in A(t)$ we also let $\mathit{time}(s, a) := \mathit{time}(t, a)$. Let $U' = \{(P'_1, R'_1), \ldots, (P'_m, R'_m)\}$ be the acceptance list for Π'. The following theorem enables to decide whether $s \models \mathrm{A}\phi$.

Theorem 8.4 (model checking of A in FPTL, FPTL*) *$s \models \mathrm{A}\phi$ iff there is no admissible end component (C, D) reachable from $\langle s, q_{in}\rangle$ in $\Pi_{\neg\phi'}$ such that $C \subseteq P'_i$ and $C \cap R'_i \neq \emptyset$ for some $1 \leq i \leq m$.*

This theorem leads immediately to the following model-checking algorithm.

Algorithm 8.4 (model checking of A in FPTL, FPTL*)

Input: TPS Π and sequence formula ϕ.

Output: The set $S_{A\phi} = \{s \in S \mid s \models \mathrm{A}\phi\}$.

Method: First, compute the MDP $\Pi_{\phi'} = (S', A', p', \mathcal{F}', U')$ as described above, and let

$$S_{\neg\phi} = \bigcup \Big\{ B \;\Big|\; \exists (P, R) \in U' \,.\, \exists (C, D) \in maxAEC(P) \,.\, \big[B = C \wedge C \cap R \neq \emptyset \big] \Big\} .$$

Then,

$$S_{A\phi} = \Big\{ s \in S \;\Big|\; \langle s, q_{in}\rangle \text{ cannot reach } S_{\neg\phi} \text{ in } (S', \rho_{S'}) \Big\} . \qquad \blacksquare$$

Since operators A and E are the dual one of the other, as stated by Theorem 8.1, the above algorithm also provides a solution to the model-checking problem for the path quantifier E.

8.5.4 Operator P

We provide an algorithm to check whether $s \models \mathrm{P}_{\bowtie a}\phi$ holds, for $0 \leq a \leq 1$ and $\bowtie \in \{\leq, <\}$. The other cases can be solved by exploiting the relations

$$s \models \mathrm{P}_{>a}\phi \quad \textit{iff} \quad s \models \mathrm{P}_{<1-a}\neg\phi$$

$$s \models \mathrm{P}_{\geq a}\phi \quad \textit{iff} \quad s \models \mathrm{P}_{\leq 1-a}\neg\phi$$

which hold for FPTL and FPTL*, as it can be easily shown. While it is possible to use the algorithms of this subsection (taking $a = 0$ or 1) to obtain model-checking algorithms for A and E, the algorithms presented in the preceding subsection are more efficient, since they avoid the reduction to linear programming that is necessary for the case $0 < a < 1$.

Given a fair TPS $\Pi = (S, A, p, \mathcal{F}, S_{in}, time, \mathcal{I})$ and a sequence formula ϕ of FPTL or FPTL*, let ϕ' be the result of replacing the maximal state subformulas in ϕ with propositional symbols, as in the previous subsection, and let

$$\Pi_\phi = (S', A', p', \mathcal{F}, time, U') = \Pi \otimes DR_{\phi'}$$

be the MDP obtained by taking the product between the l-labeled TPS Π and the deterministic Rabin automaton $DR_{\phi'}$, and adding the $\mathcal{F}$ and *time* labels as described in the algorithm for path quantifiers. The following theorem is the analogous of Theorem 4.2, and provides an algorithm for the model checking of operator P in FPTL and FPTL*. This algorithm is very similar to the one presented in [KB96, Section 10] (which in turn is based in part on [BdA95, dA97]). The similarity of the algorithms is in spite of the fact that the definition of fairness, and other details of the system models, are different.

Theorem 8.5 (model-checking of P **in FPTL and FPTL*)** *Let*

$$S_\phi = \bigcup \Big\{ B \;\Big|\; \exists (P,R) \in U' \,.\, \exists (C,D) \in \mathit{maxAEC}(P) \,.\, \Big[B = C \wedge C \cap R \neq \emptyset \Big] \Big\} \,.$$

Then,

$$\sup_{\eta \in \mathit{Adm}(s)} \mathrm{Pr}_s^\eta(\phi) \;=\; \max_\eta \mathrm{Pr}^\eta_{\langle s, q_{in} \rangle}(\mathit{reach}(S_\phi)) \,,$$

where the rightmost reachability probability is computed in Π_ϕ *over the set of all possible policies.*

8.5.5 Model Checking $\mathrm{D}_{\bowtie a}$ for $\bowtie \in \{\leq, <\}$

The model checking of operator D is based on a reduction to the stochastic shortest path problem, as was the case for GPTL and GPTL*. The reduction used for FPTL and FPTL* is more complex, due to the presence of fairness and to the approach adopted to the problem of time divergence. We present first the case in which the inequality operator used as subscript of D is $\leq$ or $<$.

Consider a fair TPS $\Pi = (S, A, p, \mathcal{F}, S_{in}, \mathit{time}, \mathcal{I})$, and let $\mathrm{D}_{\bowtie a}\phi$ be the formula to be model checked, where ϕ is a state formula of FPTL or FPTL*, $\bowtie \in \{\leq, <\}$ and $a \geq 0$. As usual, we assume that the truth value of ϕ has already been evaluated at all states, and we let $S_\phi = \{s \in S \mid s \models \phi\}$. The following lemma takes care of the states in S_ϕ.

Lemma 8.1 *If* $s \in S_\phi$*, then* $s \models \mathrm{D}_{\bowtie a}\phi$*, for* $\bowtie \in \{\leq, <\}$ *and any* $a \geq 0$.

Next, we consider the states in $S - S_\phi$. To determine the subset of $S - S_\phi$ from which the expected time to S_ϕ diverges, we reason as follows. Assume that there is an end component (B, C) in $S - S_\phi$ containing at least one state-action pair $(s, a) \in \mathit{sa}(B, C)$ with $\mathit{time}(s, a) > 0$. Assume also that (B, C) is reachable from s without leaving $S - S_\phi$. Then, there is an admissible policy that reaches (B, C) from s with positive probability. If (B, C) is an admissible end component, this admissible policy can force any behavior that reaches (B, C) to remain in (B, C) forever, thus causing the expected time to S_ϕ from s to diverge. Even if (B, C) is not admissible, however, we can construct a sequence $\eta_0, \eta_1, \ldots$ of admissible policies that, once in (B, C), leave (B, C) with probability at most $\epsilon_0, \epsilon_1, \ldots$, with $\lim_{n \to \infty} \epsilon_n = 0$. These policies spend an expected amount of time in (B, C) that diverges as $n \to \infty$; the expected time from s to S_ϕ will also diverge. These considerations lead to the following result.

Theorem 8.6 *Let*

$$B = \bigcup \Big\{ B' \;\Big|\; \exists (C,D) \in \mathit{maxEC}(S - S_\phi) \,.\, \exists (s,a) \in \mathit{sa}(C,D) \,.\, \Big[B' = C \wedge \mathit{time}(s,a) > 0 \Big] \Big\} \;,$$

and let $\widehat{B}$ be the subset of $S - S_\phi$ consisting of the states that can reach B in the graph $(S - S_\phi, \rho_{S-S_\phi})$. For $s \in \widehat{B}$, $\bowtie \in \{\leq, <\}$ and any $a \geq 0$, it is $s \not\models \mathrm{D}_{\bowtie a}\phi$.

We must still compute the truth value of $\mathrm{D}_{\bowtie a}\phi$ at the states in $C = S - (S_\phi \cup \widehat{B})$. To do so, we construct an instance of SSP problem $(C \cup S_\phi, A, p, S_\phi, c, g)$ where the cost c is defined by $c(s,a) = -\mathit{time}(s,a)$ for all $s \in C$ and $a \in A(s)$, and g is identically 0. Note that we have slightly abused the notation, since in this instance there can be actions leading from S_ϕ outside of $C \cup S_\phi$. However, since the transition structure in S_ϕ plays no role in the SSP problem, this abuse of notation has no ill consequences.

Let $v^*_s = \inf_{\eta \in \boldsymbol{\eta}_P} v^\eta_s$, as usual. The following theorem establishes the connection between the solution of this instance of SSP problem and the model checking of $\mathrm{D}_{\bowtie a}$.

Theorem 8.7 *For $s \in C$, let v^*_s be the solution of the SSP problem mentioned above. Then,*

$$s \models \mathrm{D}_{\bowtie a}\phi \quad \mathit{iff} \quad -v^*_s \bowtie a \;,$$

for $s \in C$, $a \geq 0$ and $\bowtie \in \{\leq, <\}$.

Differently from Section 4.5.1, we cannot use the results of Bertsekas and Tsitsiklis [BT91] to solve this instance of SSP problem, since SSP Assumption 2 does not necessarily hold. In fact, we cannot rule out the presence of an end component (B, D) in C such that $\mathit{time}(s,a) = 0$ for all $s \in B$, $a \in D(s)$. In Section 4.5.2, the existence of such end components was ruled out by the non-Zenoness requirement. This instance of SSP problem can nonetheless be solved with the results of Section 7.2.

8.5.6 Model Checking $\mathrm{D}_{\bowtie a}$ for $\bowtie \in \{\geq, >\}$

Similarly to the case of GPTL and GPTL*, the first step in the model checking of $\mathrm{D}_{\bowtie a}\phi$ consists in computing the sub-MDP consisting of the states that can reach S_ϕ with probability 1 under an admissible policy. The following algorithm is the analogous of Algorithms 3.2 and 8.3.

Algorithm 8.5

Input: TPS Π and set S_ϕ.

Output: The sub-MDP (S', A'), where $S' = \{s \in S \mid \max_{\eta \in \mathit{Adm}(s)} \Pr^\eta_s(\mathit{reach}(S_\phi)) = 1\}$, and A' is the restriction of A to S'.

Method:

- Let $S' = S$, $A' = A$.
- Repeat:
  - Let B' be the subset of states of S' that cannot reach S_ϕ in the graph $(S', \rho_{(S',A')})$.
  - Remove from S' all states in B'.
  - Repeat:
    - Remove from S' all states $s \notin S_\phi$ that have a fair action leading to at least one removed state.
    - Remove from A' all the actions leading to at least one removed state.
    - Remove from S' all states $s \notin S_\phi$ that have no actions left in A'.
  - Until no more states or actions can be removed from S', A'.
- Until no more states or actions can be removed from S', A'.
- Output (S', A'). ∎

The following lemmas can be used to decide the value of $\mathrm{D}_{\bowtie a}\phi$ at all states of $(S - S') \cup S_\phi$.

Lemma 8.2 *Given a state formula ϕ, let S' be as computed by Algorithm 8.5. If $s \in S - S'$, it is*

$$\mathrm{E}_s^\eta\left\{\sum_{k=0}^{T_\phi - 1} \mathit{time}(X_k, Y_k)\right\} = \infty$$

for any policy η. Thus, for $\bowtie \in \{\geq, >\}$, $a \geq 0$ and $s \in S - S'$ it is $s \models \mathrm{D}_{\bowtie a}\phi$.

Lemma 8.3 *If $s \in S_\phi$, then $s \models \mathrm{D}_{\bowtie a}\phi$ iff both $a = 0$ and $\bowtie$ is $\geq$.*

We must still determine the truth value of $\mathrm{D}_{\bowtie a}\phi$ at all states in $C = S' - S_\phi$. To do so, we construct an instance of SSP problem $(S', A', p, S_\phi, c, g)$, where the cost c is defined by $c(s,a) = \mathit{time}(s,a)$ for all $s \in S' - S_\phi$ and $a \in A'(s)$, and g is identically 0. Note that this definition involves again a slight abuse of notation, similarly to the construction of the SSP instance for Theorem 8.7.

Let $v_s^* = \inf_{\eta \in \boldsymbol{\eta}_P} v_s^\eta$, as usual. The following theorem establishes the connection between the solution of this instance of SSP problem and the model checking of $\mathrm{D}_{\bowtie a}$.

Theorem 8.8 *For $s \in C$, let v_s^* be the solution of the SSP problem mentioned above. Then,*

$$s \models \mathrm{D}_{\bowtie a}\phi \quad \mathit{iff} \quad v_s^* \bowtie a\ ,$$

for $s \in C$, $a \geq 0$ and $\bowtie \in \{\geq, >\}$.

As in the preceding subsection, we cannot use the results of Bertsekas and Tsitsiklis [BT91] to solve this instance of SSP problem, since SSP Assumption 2 does not necessarily hold. Again, the reason is that we cannot rule out the presence of end components (B, D) in C such that $time(s, a) = 0$ for all $s \in B$, $a \in D(s)$. This instance of SSP problem can nonetheless be solved with the results of Section 7.1.

8.5.7 Operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$

The model checking of operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ in FPTL and FPTL* relies on an algorithm almost identical to the one given for GPTL and GPTL* in Section 6.1.2. Let $\Pi' = (S, A, p, \mathcal{F}, time, W, R)$ be the result of computing the product between a TPS Π_0 and an experiment Ψ, as in Definition 8.6, and of substituting the r, w labels with the R, W ones.

The only difference between the algorithm for FPTL and FPTL* and the one presented in Section 6.1.2 is that in the algorithm for FPTL and FPTL*, Step 1 is replaced by the following step:

1'. Compute the list $\mathcal{L} = \{(S_1, A_1), \dots, (S_n, A_n)\}$ of maximal admissible end components of Π'. For each end component (S_i, A_i), $1 \le i \le n$, construct an MDP $\Pi_i = (S_i, A_i, p^i, R_i, W_i)$, where p^i, R_i, W_i are the restrictions of p, R, W to (S_i, A_i).

The algorithm then proceeds as in Section 6.1.2.

8.6 Complexity of FPTL and FPTL* Model Checking

From an exam of the model-checking algorithms given in the previous sections, we can see that the model-checking of FPTL and FPTL* specifications shares the same global complexity bounds of the model-checking for GPTL and GPTL*.

Theorem 8.9 (complexity of FPTL, FPTL* model checking) *Given a TPS Π, the following assertions hold:*

- *Checking whether Π satisfies Assumption 8.1 has polynomial time-complexity in $||\Pi||$.*
- *Model checking an FPTL formula ϕ has time-complexity linear in $|\phi|$ and polynomial in $|\Pi|$.*
- *Model checking an FPTL* formula ϕ has time-complexity doubly exponential in $|\phi|$ and polynomial in $|\Pi|$.*

A more detailed result is as follows.

Theorem 8.10 (complexity of FPTL, FPTL* model checking II) *Assume that the truth of all state subformulas of ϕ has already been evaluated at all states. The following assertions hold:*

- *The model checking of $\mathrm{A}\phi$, $\mathrm{E}\phi$ has the following time-complexity:*

  FPTL: *Polynomial in $||\Pi||$ and independent of ϕ.*

Operator	Logic			
	GPTL	GPTL*	FPTL	FPTL*
A, E	$poly(\|\|\Pi\|\|)$ $indep(\phi)$	$poly(\|\|\Pi\|\|)$ $\mathbf{exp}(\|\phi\|)$	$poly(\|\|\Pi\|\|)$ $indep(\phi)$	$poly(\|\|\Pi\|\|)$ $\mathbf{2\text{-}exp}(\|\phi\|)$
P	$poly(\|\Pi\|)$ $indep(\phi)$	$poly(\|\Pi\|)$ $2\text{-}exp(\|\phi\|)$	$poly(\|\Pi\|)$ $indep(\phi)$	$poly(\|\Pi\|)$ $2\text{-}exp(\|\phi\|)$
D	$poly(\|\Pi\|)$ $indep(\phi)$	$poly(\|\Pi\|)$ $indep(\phi)$	$poly(\|\Pi\|)$ $indep(\phi)$	$poly(\|\Pi\|)$ $indep(\phi)$
$\bar{\mathrm{P}}$, $\bar{\mathrm{D}}$	$poly(\|\Pi\|)$ $poly(\|\Psi\|)$	$poly(\|\Pi\|)$ $poly(\|\Psi\|)$	$poly(\|\Pi\|)$ $poly(\|\Psi\|)$	$poly(\|\Pi\|)$ $poly(\|\Psi\|)$

Table 8.2: Complexity of the model-checking algorithms for the probabilistic logics considered in this dissertation. In this table, we assume that the truth of all state subformulas of ϕ has already been evaluated at all states. Symbol Ψ indicates an experiment. The symbol *indep*indicates that the complexity does not depend on that parameter; the symbols *poly*, *exp* and *2-exp* indicate polynomial, exponential and doubly exponential complexities respectively. The case where the logics with and without fairness differ has been indicated in boldface.

FPTL*: *Polynomial in* $||\Pi||$ *and doubly exponential in* $|\phi|$.

- *The model checking of* $\mathrm{P}_{\bowtie a}\phi$ *has the following time-complexity:*

  FPTL: *polynomial in* $|\Pi|$ *and independent of* $|\phi|$.

  FPTL*: *polynomial in* $|\Pi|$ *and doubly exponential in* $|\phi|$.

- *The model checking of* $\mathrm{D}_{\bowtie a}\phi$ *has time-complexity polynomial in* $|\Pi|$ *and independent of* ϕ.

- *The model checking of* $\bar{\mathrm{P}}_{\bowtie a}\Phi$, $\bar{\mathrm{D}}_{\bowtie a}\Phi$ *for an experiment* Ψ *has time-complexity polynomial in both* $|\Pi|$ *and* $|\Phi|$.

Proof. The lower bounds are a consequence of the results of Courcoubetis and Yannakakis [CY88]; the upper bounds are derived from an analysis of the proposed model-checking algorithms. ∎

Table 8.2 summarizes the results on the complexity of the model-checking algorithms for the probabilistic logics presented in this dissertation.

8.7 Correctness Proof of Model-Checking Algorithms (‡)

8.7.1 Policy Approximation

The computational part of the model-checking algorithms for FPTL and FPTL* relies on the solution of optimization problems on Markov decision processes. These problems are solved in an *unconstrained* fashion, without placing any restriction on the policies, and the optimal policy or policies may not be admissible. On the other hand, the semantics of the FPTL and FPTL* operators is defined only in terms of admissible policies.

To justify the use of unconstrained solutions in FPTL or FPTL* model checking, we have to show that the optimal values of the quantities of interest, such as probabilities or expected times, can be realized or at least approximated by admissible policies. To this end, we use the notion of *approximation* of non-admissible policies by admissible ones. This notion has been discussed in Pnueli [Pnu83] and Kwiatkowska and Baier [KB96].

Informally, the idea is that it is sometimes possible to approximate a non-admissible policy with a series of admissible ones, such that the quantity of interest (probability, expected time) for the admissible policies can be made as close as desired to that of the non-admissible policy. We have already seen this notion at work in the proof of Lemma 7.2. In this subsection, we present results that will help to construct other approximability proofs.

Recall that a *substochastic matrix* is a matrix $P = [p_{ij}]_{1 \le i,j \le N}$ such that $0 \le p_{ij} \le 1$ for all $1 \le i,j \le N$ and $\sum_{j=1}^{N} p_{ij} \le 1$ for all $1 \le i \le N$ [KSK66].

Theorem 8.11 (continuity of steady-state distributions) *For a fixed N, consider a family $P(x) = [p_{ij}(x)]_{1 \le i,j \le N}$ of substochastic matrices parameterized by a parameter $x \in I$, where $I \subseteq \mathbb{R}$ is an interval of real numbers. Assume that the Markov chain having P as transition matrix has the same set of closed recurrent classes for all $x \in I$. Then, if the coefficients of $P(x)$ depend continuously on x for $x \in I$, also the coefficients of the steady-state matrix $P^*(x)$ depend continuously on x for $x \in I$.*

Proof. The proof proceeds in stages, paralleling the classification of states in a finite Markov chain.

First, we consider each closed recurrent class separately. Assume that the matrix P corresponds to a chain consisting in a single closed recurrent class. Then, the matrix $P^*(x)$ can be written as $P^*(x) = \mathbf{1}\boldsymbol{\pi}(x)$, where $\mathbf{1}$ is a column vector consisting of N 1s, and $\boldsymbol{\pi}(x)$ is the row vector corresponding to the steady-state distribution of the chain. The row vector $\boldsymbol{\pi}(x)$ is the solution of the equation

$$\boldsymbol{\pi}(x)[P(x)\,\mathbf{1}] = [\boldsymbol{\pi}(x)\,1] , \tag{8.3}$$

where $[P(x)\,\mathbf{1}]$ indicates the $N \times (N+1)$ matrix obtained by adjoining the column vector $\mathbf{1}$ to $P(x)$, and similarly $[\boldsymbol{\pi}(x)\,1]$ is the row vector with $N+1$ elements obtained by adjoining a single 1 to $\boldsymbol{\pi}(x)$.

Equation (8.3) is over-constrained: since the chain is composed of a single closed recurrent class, any column of $P(x)$ can be written as the linear combination of the others. Thus, in (8.3) we can delete the columns corresponding to the first element, and determine $\boldsymbol{\pi}(x)$ by

$$\boldsymbol{\pi}(x)[P_{\sharp 1}(x)\,\mathbf{1}] = [\boldsymbol{\pi}_{\sharp 1}(x)\,1]\ , \tag{8.4}$$

where $P_{\sharp 1}(x)$ is the $(N-1) \times N$ matrix obtained from $P(x)$ by deleting the first column, and $\boldsymbol{\pi}_{\sharp 1}$ is the row vector of $N-1$ elements obtained by deleting the first element of $\boldsymbol{\pi}(x)$. Equation (8.4) can in turn be rewritten as

$$\boldsymbol{\pi}(x)Q(x) = \boldsymbol{\pi}(x)R + \boldsymbol{d}\ , \tag{8.5}$$

where

$$Q(x) = [P_{\sharp 1}(x)\,\mathbf{1}] \qquad R = [\delta_{i-1,j}]_{1 \le i,j \le N} \qquad \boldsymbol{d} = [\underbrace{00\cdots0}_{N-1}1]$$

and where $\delta_{i,j} = 1$ if $i = j$ and 0 otherwise. From (8.5), solving for $\boldsymbol{\pi}(x)$ we obtain

$$\boldsymbol{\pi}(x) = \boldsymbol{d}(Q(x) - R)^{-1}\ .$$

Since for $x \in I$ all states are in the same single closed recurrent class, this equation has a single solution for all $x \in I$, so that $\det(Q(x) - R) \neq 0$ for $x \in I$. If $\det(Q(x) - R) \neq 0$, the coefficients of $(Q(x) - R)^{-1}$ are continuous functions (in fact, rational functions) of the coefficients of $P(x)$. Hence, the coefficients of $\boldsymbol{\pi}(x)$ and those of $P^*(x)$ are continuous in x, for $x \in I$.

Assume now that the Markov chain corresponding to $P(x)$ has closed recurrent classes $C_1, \ldots C_M \subseteq \{1, \ldots, N\}$ for $x \in I$. For $1 \le i \le N$, call $\boldsymbol{P}_i^*(x)$ the i-th row of matrix $P^*(x)$. Let $C = \bigcup_{j=1}^{M} C_j$ be the union of the closed recurrent classes, and $T = \{1, \ldots, N\} - C$ be the set of transient states. For $1 \le i \le N$, row $\boldsymbol{P}_i^*(x)$ can be written as

$$\boldsymbol{P}_i^*(x) = \sum_{j=1}^{M} q_i^j(x)\,\boldsymbol{\pi}_j(x)\ , \tag{8.6}$$

where $\boldsymbol{\pi}_j(x)$ is the row vector corresponding to the steady-state distribution of class C_j, and $q_i^j(x)$ is the probability of reaching C_j from i. If $i \in C_k$, then $q_i^i(x) = 1$ and $q_i^j(x) = 0$ for all $x \in I$ and all $j \neq k$.

For $i \in T$, these probabilities are related by

$$q_i^j(x) = \sum_{k \in T} p_{ik}(x)\,q_k^j(x) + \sum_{k \in C_j} p_{ik}(x)$$

for $1 \leq j \leq M$. In matrix form, this can be written as

$$\boldsymbol{Q}_j(x) = P_T(x)\, \boldsymbol{Q}_j(x) + \boldsymbol{R}_j(x) ,$$

where $\boldsymbol{Q}_j = [q_i^j(x)]_{i \in T}$ is a column vector, $P_T(x)$ is the restriction of $P(x)$ to the transient states, and $\boldsymbol{R}_j(x) = [r_i^j(x)]_{i \in T}$ is defined by

$$r_i^j = \sum_{k \in C_j} p_{ik}(x) .$$

Solving for $\boldsymbol{Q}_j(x)$ we obtain

$$\boldsymbol{Q}_j(x) = (I - P_T(x))^{-1} \boldsymbol{R}_j(x) . \tag{8.7}$$

Since $P_T(x)$ is the matrix of a transient chain, $\det(I - P_T(x)) \neq 0$, and the coefficients of $(I - P_T(x))^{-1}$ are continuous functions (in fact, rational functions) of x for $x \in I$. From (8.6) and (8.7), and from the previous part of the proof, we can thus conclude that the coefficients of $P^*(x)$ are continuous in x, for $x \in I$. ∎

A similar result holds for chains in which there is a single closed recurrent class, and there is a state that is always in that class, for all values of the parameter. To state the result, we say that a state is *surely recurrent* if the Markov chain has only one closed recurrent class, and the state belongs to that class.

Theorem 8.12 *For a fixed N, consider a family $P(x) = [p_{ij}(x)]_{1 \leq i,j \leq N}$ of substochastic matrices parameterized by a parameter $x \in I$, where $I \subseteq \mathbb{R}$ is an interval of real numbers. Assume that there is a state $1 \leq k_0 \leq N$ that is surely recurrent for all $x \in I$. Then, if the coefficients of $P(x)$ depend continuously on x for $x \in I$, also the coefficients of the steady-state distribution vector $\boldsymbol{\pi}^*(x)$ depend continuously on x for $x \in I$.*

Proof. This theorem can be proved similarly to the first part of the previous one. The row vector $\boldsymbol{\pi}(x)$ is the solution of the equation

$$\boldsymbol{\pi}(x)[P(x)\,\mathbf{1}] = [\boldsymbol{\pi}(x)\,1] , \tag{8.8}$$

where $[P(x)\,\mathbf{1}]$ indicates the $N \times (N+1)$ matrix obtained by adjoining the column vector $\mathbf{1}$ to $P(x)$, and similarly $[\boldsymbol{\pi}(x)\,1]$ is the row vector with $N+1$ elements obtained by adjoining a single 1 to $\boldsymbol{\pi}(x)$.

Equation (8.8) is over-constrained: since the chain has a single closed recurrent class, the columns of $P(x)$ that correspond to states in the class can be written as the linear combination of the other columns. Thus, in (8.3) we can delete the column corresponding to k_0, and determine $\boldsymbol{\pi}(x)$ by

$$\boldsymbol{\pi}(x)[P_{\sharp k_0}(x)\,\mathbf{1}] = [\boldsymbol{\pi}_{\sharp k_0}(x)\,1] , \tag{8.9}$$

where $P_{\sharp k_0}(x)$ is the $(N-1) \times N$ matrix obtained from $P(x)$ by deleting column k_0, and $\boldsymbol{\pi}_{\sharp k_0}$ is the row vector of $N-1$ elements obtained by deleting the k_0-th element of $\boldsymbol{\pi}(x)$. The proof is then concluded in the same way as the previous one. ∎

8.7.2 Model Checking of Path Quantifiers

The proof of Theorem 8.4 is related to the results of Courcoubetis and Yannakakis [CY88, CY95] relative to the problem of *probabilistic emptiness.* The differences lie in the presence of fairness, in the concept of admissible policies, and in the use of Rabin automata and end components in the analysis of the problem.

Proof of Theorem 8.4. Recall that a behavior ω of Π satisfies $\neg\phi$ iff there is $1 \leq i \leq n$ such that $\mathit{inftst}(\omega') \subseteq P_i$ and $\mathit{inftst}(\omega') \cap R_i \neq \emptyset$, where ω' is the behavior of $\Pi_{\neg\phi}$ corresponding to ω, and $\mathit{inftst}(\omega')$ denotes the set of states occurring infinitely often along ω'.

If there is no admissible end component (B, C) reachable from $\langle s, q_{in}\rangle$ such that $B \subseteq P_i$ and $B \cap R_i \neq \emptyset$ for some $1 \leq i \leq n$, then from Theorem 8.3 we can conclude $\Pr^{\eta}_{s}(\neg\phi) = 0$ for every policy η, as desired.

Conversely, assume that there is an end component (B, C) reachable from $\langle s, q_{in}\rangle$ such that $B \subseteq P_i$ and $B \cap R_i \neq \emptyset$ for some $1 \leq i \leq n$. Then, it is not difficult to see that there is an admissible policy η such that $\Pr^{\eta}_{\langle s, q_{in}\rangle}(\mathit{reach}(B)) > 0$ (the existence of such a policy will be discussed in more detail in the next subsection). From Theorem 8.2, we see that there is another admissible policy η' such that

$$\Pr^{\eta'}_{\langle s, q_{in}\rangle}\Big(\mathit{inftst}(\omega) \subseteq P_i \wedge \mathit{inftst}(\omega) \cap R_i \neq \emptyset\Big) = \Pr^{\widetilde{\eta}}_{s}(\neg\phi) > 0 \ ,$$

where $\widetilde{\eta}$ is the policy that corresponds to η' in Π. ∎

The correctness of Algorithm 8.4 follows immediately from Theorem 8.4.

8.7.3 Model Checking of Operator P

In this and other proofs, it will be convenient to refer to a fixed Markovian admissible policy. One such policy can be defined as follows.

Definition 8.11 (η_f) Denote by η_f be the Markovian policy that chooses at every state s an action from $A(s)$ with uniform probability.

Lemma 8.4 *Policy η_f is admissible.*

Proof. Policy η_f is obviously fair. To see that it is admissible, note that from every state s there is a reachable state-action pair with $\mathit{time} > 0$, since Assumption 8.1 holds. Thus, from every state s

there are $q > 0$ and $k \in \mathbb{N}$ such that $\Pr_s^{\eta_f}(\exists i \,.\, i \leq k \wedge \mathit{time}(X_i, Y_i) > 0) \geq q$. This yields the result. ∎

The proof of Theorem 8.5 proceeds then as follows.

Proof of Theorem 8.5. Let $\Pi_\phi = (S', A', p', U') = \Pi \otimes DR_{\phi'}$ be the product between the l-labeled TPS Π and the Rabin automaton $DR_{\phi'}$ defined as in Algorithm 4.2. Add to Π_ϕ labels $\mathcal{F}'$, l' and time' defined by

$$\mathcal{F}'(\langle s, u \rangle) = \mathcal{F}(s) \qquad l'(\langle s, u \rangle) = l(s) \qquad \mathit{time}'((\langle s, u \rangle), a) = \mathit{time}(s, a)$$

for all $(s, u) \in S'$ and $a \in A(s)$. With these additional labelings, we can extend the notion of admissible policy to Π_ϕ, and we can define the truth value of ϕ' over a behavior of Π_ϕ.

The construction of Π_ϕ as the product of Π with a deterministic automaton creates a one-to-one correspondence between the policies for Π and those for Π_ϕ. Specifically, to the policy η for Π corresponds the policy $\mu(\eta)$ for Π_ϕ defined by

$$Q_\eta(a \mid s_0 s_1 \cdots s_n) = Q_{\mu(\eta)}\Big(a \;\Big|\; \langle s_0, q_{in} \rangle \langle s_1, q_1 \rangle \cdots \langle s_n, q_n \rangle\Big) ,$$

where $q_{in}, q_1, \ldots, q_n$ is the unique sequence of automata states induced by $s_0, s_1, \ldots, s_n$ in DR_ϕ. Note that $\mu(\eta)$ can be Markovian even when η is history dependent. However, from the previous considerations follows that η is admissible iff $\mu(\eta)$ is. By abuse of notation, we will denote $\mu(\eta)$ simply as η, relying on the context to distinguish the MDP for which the policy is defined.

From the correspondence between Π and Π_ϕ, we have

$$\sup_{\eta \in \mathit{Adm}(s)} \Pr_s^\eta(\phi) \;=\; \sup_{\eta \in \mathit{Adm}(\langle s, q_{in} \rangle)} \Pr_{\langle s, q_{in} \rangle}^\eta(\phi') \,.$$

Thus, it suffices to prove

$$\sup_{\eta \in \mathit{Adm}(s)} \Pr_s^\eta(\phi') \;=\; \max_\eta \Pr_s^\eta(\mathit{reach}(S_\phi)) \tag{8.10}$$

for all states $s \in S'$ of Π_ϕ. First, note that

$$\max_\eta \Pr_s^\eta(\mathit{reach}(S_\phi)) \;\geq\; \sup_{\eta \in \mathit{Adm}(s)} \Pr_s^\eta(\mathit{reach}(S_\phi)) \;\geq\; \sup_{\eta \in \mathit{Adm}(s)} \Pr_s^\eta(\phi') \tag{8.11}$$

for all $s \in S'$. The first inequality is immediate. The second inequality follows from the fact that a behavior from s follows with probability 1 an admissible end component. Thus, the probability that a behavior satisfies ϕ' without entering S_ϕ is 0 (see the proof of Theorem 4.2 for a more detailed argument).

In the reverse direction, let η_m be a deterministic policy for Π_ϕ such that

$$\Pr_s^{\eta_m}(\mathit{reach}(S_\phi)) \;=\; \max_\eta \Pr_s^\eta(\mathit{reach}(S_\phi)) \; . \tag{8.12}$$

The existence of η_m is guaranteed by Corollary 3.2. Let $B \subseteq S'$ be the set of states in S' that cannot reach S_ϕ. By (8.11), for $s \in B$ it is $\Pr_s^\eta(\phi') = 0$ under any policy η. From η_m we can construct a Markovian policy η_M that differs from η_m only on B and on S_ϕ. On S_ϕ, η_M is constructed as in the proof of Lemma 4.1; on B it coincides with η_f. Since η_m and η_M coincide on the set $C = S' - (B \cup S_\phi)$, it is

$$\Pr_s^{\eta_m}(\mathit{reach}(S_\phi)) \;=\; \Pr_s^{\eta_M}(\phi) \tag{8.13}$$

for all $s \in C$. Note also that η_M is admissible. Given $0 \leq x \leq 1$, denote by $\eta(x)$ the Markovian policy defined by

$$Q_{\eta(x)}(a \mid s) = \begin{cases} (1-x)\, Q_{\eta_M}(a \mid s) + x\, Q_{\eta_f}(a \mid s) & \text{for } s \in C; \\ Q_{\eta_M}(a \mid s) & \text{for } s \in S_\phi \cup B \end{cases}$$

for all $s \in S'$ and $a \in A'(s)$, where η_f is the admissible policy previously described.

For each $0 < x < 1$, policy $\eta(x)$ is admissible. In fact, outside of S_ϕ policy $\eta(x)$ is fair, being obtained by linear combination with η_f, and time-divergent, for the same reason; inside S_ϕ, policy $\eta(x)$ coincides with η_M.

Denote by $P(x) = [p_{st}(x)]_{s,t \in C}$ the transition matrix corresponding to policy $\eta(x)$ restricted to the set C of states. For $0 \leq x < 1$, $P(x)$ is a substochastic matrix. Note that $P(0)$ is the transition matrix for policy η_M.

The closed recurrent classes of the Markov chain corresponding to $P(x)$ are constant for $0 \leq x < 1$. In fact, the closed recurrent classes of η_M and $\eta(x)$ are all contained in $S_\phi \cup B$, and η_M and $\eta(x)$ coincide on $S_\phi \cup B$. Denoting by $P^*(x) = [p^*_{st}(x)]_{s,t \in C}$ the steady-state matrix corresponding to $P(x)$, we can write the reachability probabilities as:

$$\Pr_s^{\eta(x)}(\mathit{reach}(S_\phi)) = \sum_{t \in S_\phi} p^*_{st}(x) \; . \tag{8.14}$$

From $\lim_{x \to 0} P(x) = P_{\eta_M}$, by Theorem 8.11 we have $\lim_{x \to 0} P^*(x) = P^*_{\eta_M}$, and by (8.14) this yields

$$\lim_{x \to 0} \Pr_s^{\eta(x)}(\mathit{reach}(S_\phi)) = \Pr_s^{\eta_m}(\mathit{reach}(S_\phi)) \; .$$

By (8.12) and (8.13), this indicates that

$$\sup_{\eta \in \mathit{Adm}(\langle s, q_{in} \rangle)} \Pr^\eta_{\langle s, q_{in} \rangle}(\phi) \;\geq\; \max_\eta \Pr^\eta_{\langle s, q_{in} \rangle}(\mathit{reach}(S_\phi)) \; ,$$

which is the other direction of (8.11). This completes the proof. ∎

8.7.4 Model Checking $\mathrm{D}_{\bowtie a}$ for $\bowtie \in \{\leq, <\}$

To state the proofs, given a policy η we define the vector $\boldsymbol{w} = [w_s^\eta]_{s \in S}$ by

$$w_s^\eta = \mathrm{E}_s^\eta \Big\{ \sum_{k=0}^{T_\phi - 1} \mathit{time}(X_k, Y_k) \Big\}$$

for all $s \in S$. Thus, w_s^η represents the time needed to reach S_ϕ under $\eta(x)$. For simplicity, we consider in the following arguments a modified TPS, in which every state $s \in S_\phi$ has a single action a, with $\mathit{time}(s,a) = 1$ and $p_{ss}(a) = 1$. This modification does not change the expected time to S_ϕ, which is the quantity that interests us. This modification changes the set of admissible policies, but it will not cause problems. In fact, in the original TPS once reached S_ϕ it is possible to follow policy η_f, ensuring admissibility. In the modified TPS, once reached S_ϕ there is only one admissible policy possible. The two conditions are equivalent for the purpose of our analysis.

Proof of Theorem 8.6. The proof follows the lines of the informal argument that precedes the statement of the theorem, and is also related to the proof of Lemma 7.1.

Assume that there is an end component $(C, D) \in \mathit{maxEC}(S - S_\phi)$ containing at least one state-action pair $(s, a) \in \mathit{sa}(B, C)$ with $\mathit{time}(s, a) > 0$. Assume also that C is reachable from a state $s \in S - S_\phi$ without leaving $S - S_\phi$. Then, the admissible policy η_f of Definition 8.11 will reach C from s with positive probability. From η_f, we define another Markovian policy η_m which coincides with η_f outside of C, and that at $t \in C$ chooses an action from $D(s)$ uniformly at random.

If (C, D) is admissible, then η_m is admissible, and from $\Pr_s^{\eta_m}(T_\phi = \infty) > 0$ we have $\mathrm{E}_s^{\eta_m}\{\sum_{k=0}^{T_\phi - 1} \mathit{time}(X_k, Y_k)\} = \infty$, from which follows the result.

If (C, D) is not admissible, for $0 \leq x < 1$ we define the policy $\eta(x)$ by

$$Q_{\eta(x)}(a \mid t) = (1 - x)\, Q_{\eta_m}(a \mid t) + x\, Q_{\eta_f}(a \mid t)$$

for all $t \in S$ and $a \in A(t)$. Note that, for all $0 < x < 1$, policy $\eta(x)$ is admissible.

Let P_m be the transition matrix associated with η_m, and $P(x)$ be the matrix associated with $\eta(x)$, for $0 < x < 1$. Note that $\lim_{x \to 0} P(x) = P_m$; however, for $0 < x < 1$ the chain corresponding to $P(x)$ may have different closed recurrent classes than the chain corresponding to $P(0)$, so that we cannot use the results about the continuity of the steady-state matrix. Instead, define the vector $\boldsymbol{u} = [u_s]_{s \in S}$ by

$$u_s = \begin{cases} \dfrac{1}{2|D(s)|} \displaystyle\sum_{a \in D(s)} \mathit{time}(s, a) & \text{if } s \in C; \\ 0 & \text{otherwise} \end{cases}$$

for all $s \in S$. Note that at least one element of $\boldsymbol{u}$ is strictly positive.

Since $\boldsymbol{u}$ represents only part of the time spent along a behavior, for $0 < x < 1/2$ we have

$$w_s^{\eta(x)} \geq \sum_{k=0}^{\infty} P^k(x)\boldsymbol{u} \ .$$

In fact, when $0 < x < 1/2$, at state $s \in C$ a behavior under $\eta(x)$ will spend at least an expected time of u_s with the next move: the limit $1/2$ comes directly from the denominator in the definition of u_s. For any fixed k it is $\lim_{x\to 0} P^k(x) = P_m^k$, and since $\sum_{k=0}^{\infty} P_m^k \boldsymbol{u} = \infty$ it follows that

$$\lim_{x\to 0} \sum_{k=0}^{\infty} P(x)^k \boldsymbol{u} = \infty$$

from which we conclude $\lim_{x\to 0} w_s^{\eta(x)} = \infty$. The result then follows immediately. ∎

Proof of Theorem 8.7. First, consider any admissible policy η; we want to prove that $\Pr_s^\eta(\mathit{reach}(S_\phi)) = 1$ for all $s \in C$. To see this, assume towards the contradiction that $\Pr_s^\eta(\mathit{reach}(S_\phi)) < 1$. Then, there is an end component (B, D) with $B \subseteq C$ such that $\Pr_s^\eta(\mathit{inft}(\omega) = \mathit{sa}(B, D)) > 0$. By definition of C, it must be $\mathit{time}(t, a) = 0$ for all $(t, a) \in \mathit{sa}(B, D)$. However, this leads to

$$\Pr_s^\eta\left(\sum_{k=0}^{\infty} \mathit{time}(X_k, Y_k) < \infty\right) > 0 \ ,$$

contradicting the admissibility of η.

Since $\Pr_s^\eta(\mathit{reach}(S_\phi)) = 1$, policy η is SSP-proper, and from the definition of the SSP problem and the optimality of $\boldsymbol{v}^*$ follows $\boldsymbol{w}^\eta = -\boldsymbol{v}^\eta \leq -\boldsymbol{v}^*$. From the arbitrariness of η, we have

$$\sup_{\eta \in Adm(s)} \boldsymbol{w}^\eta \leq -\boldsymbol{v}^* \tag{8.15}$$

for all $s \in C$.

In the other direction, let η_m be a Markovian SSP-proper policy optimal for the SSP problem, i.e. such that $\boldsymbol{v}^{\eta_m} = \boldsymbol{v}^*$. The existence of at least one such policy follows from the construction in the proof of Theorem 7.1. There are two cases. If η_m is admissible, then

$$\boldsymbol{w}^{\eta_m} = -\boldsymbol{v}^* \ . \tag{8.16}$$

If η_m is not admissible, for $0 \leq x < 1$ define the Markovian policy $\eta(x)$ by

$$Q_{\eta(x)}(a \mid s) = (1 - x)Q_\eta(a \mid s) + xQ_{\eta_f}(a \mid s)$$

for all $s \in S$, where η_f is the admissible policy mentioned in Definition 8.11.

Denote by $P(x)$ the transition matrix corresponding to policy $\eta(x)$ on the set of states C, and by P_m the transition matrix for η_m also on the set C. As usual, $P(0) = P_m$, and the chain corresponding to $P(x)$ is transient for $0 \leq x < 1$. For $0 \leq x < 1$, define the vector $\boldsymbol{d}(x) = [d_s(x)]_{s \in C}$ by

$$d_s(x) = \sum_{a \in A(s)} Q_{\eta(x)}(a \mid s)\, \mathit{time}(s, a) \ ,$$

and abbreviate $\boldsymbol{w}(x) = \boldsymbol{w}^{\eta(x)}$. With this notation, we can write

$$\boldsymbol{w}(x) = \sum_{k=0}^{\infty} P^k(x)\boldsymbol{d}(x) = (I - P(x))^{-1}\boldsymbol{d}(x) \tag{8.17}$$

$$-\boldsymbol{v}^* = \sum_{k=0}^{\infty} P^k(0)\boldsymbol{d}(0) = (I - P(0))^{-1}\boldsymbol{d}(0) \ . \tag{8.18}$$

Since matrix $P(0)$ corresponds to a transient Markov chain for $0 \leq x < 1$, it is $\det(I - P(x)) \neq 0$ in this interval. Thus, for $0 \leq x < 1$ the coefficients of $(I - P(x))^{-1}$ are rational functions of x that have no poles in the interval $[0, 1)$. From the continuity of $P(x)$ in $[0, 1)$ we have

$$\lim_{x \to 0} (I - P(x))^{-1} = (I - P_\eta)^{-1} \qquad\qquad \lim_{x \to 0} \boldsymbol{d}(x) = \boldsymbol{d}^\eta \ .$$

Using these relations in (8.17) and (8.18) we obtain $\lim_{x \to 0} \boldsymbol{w}(x) = -\boldsymbol{v}^*$, or

$$\sup_{0 < x < 1} \boldsymbol{w}(x) \geq -\boldsymbol{v}^* \ . \tag{8.19}$$

Putting together (8.15) in one direction with (8.16) or (8.19) in the other, we conclude $\sup_{\eta \in Adm(s)} \boldsymbol{w}^\eta = -\boldsymbol{v}^*$, as was to be proved. ∎

8.7.5 Model Checking $\mathrm{D}_{\bowtie a}$ for $\bowtie \in \{\geq, >\}$

Proof of Lemma 8.2. By induction on the stage at which a state is removed from S', we can prove that under any admissible policy the probability of reaching S_ϕ from the state is less than 1. Under an admissible policy, time diverges with probability 1. Hence, if a state is removed from S', the time to S_ϕ diverges with positive probability, and this yields the result. ∎

Proof of Theorem 8.8. Let $\boldsymbol{w}^\eta = [w_s^\eta]_{s \in C}$. In one direction, consider any admissible policy η and any $s \in C$; we want to show that $w_s^\eta \geq v_s^*$. There are three cases.

- If $\Pr_t^\eta(\mathit{reach}(S_\phi)) = 1$ for all $t \in C$, then η is SSP-proper, and the result follows from the optimality of $\boldsymbol{v}^*$.

- If $\Pr_s^\eta(reach(S_\phi)) < 1$, since the time diverges with probability 1 under η we have $w_s^\eta = \infty$, which leads to the result.

- If $\Pr_s^\eta(reach(S_\phi)) = 1$ and $D = \{t \in C \mid \Pr_t^\eta(reach(S_\phi)) < 1\} \neq \emptyset$, it is easy to see that there is another policy η', differing from η only at the states in D, such that η' is SSP proper, and $w_t^{\eta'} = w_t^\eta$ for all $t \in C - D$. From $\boldsymbol{w}^{\eta'} \geq \boldsymbol{v}^*$ follows $w_s^\eta \geq v_s^*$ once more.

Since the result holds for any η and $s \in C$, we conclude

$$\inf_{\eta \in \boldsymbol{\eta}_A} \boldsymbol{w}^\eta \geq \boldsymbol{v}^* \ . \tag{8.20}$$

In the other direction, let η_m be a Markovian SSP-proper policy optimal for the SSP problem, i.e. such that $\boldsymbol{v}^{\eta_m} = \boldsymbol{v}^*$. If η_m is admissible, then $\boldsymbol{w}^{\eta_m} = \boldsymbol{v}^*$. Otherwise, we construct a family $\eta(x)$ of policies, with $0 \leq x < 1$, such that $\lim_{x \to 0} \boldsymbol{w}(x) = \boldsymbol{v}^{\eta_m} = \boldsymbol{v}^*$. To construct the family, let η_a be a policy that, at $s \in C$, chooses an action from $A'(s)$ uniformly at random. From Algorithm 3.2, it is easy to check that η_a is admissible, and that $\Pr_s^{\eta_a}(reach(S_\phi)) = 1$ for all $s \in C$. For $0 \leq x < 1$, define

$$Q_{\eta(x)}(a \mid s) = (1-x) Q_{\eta_m}(a \mid s) + x Q_{\eta_a}(a \mid s) \ .$$

Reasoning as in the proof of Theorem 8.7, we conclude

$$\inf_{\eta \in \boldsymbol{\eta}_A} \boldsymbol{w}^\eta \leq \boldsymbol{v}^* \ ,$$

which together with (8.20) yields the result. ∎

8.7.6 Model Checking of Operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$

We state and prove the correctness of the model checking algorithm for $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ on fair TPSs with the following theorem.

Theorem 8.13 (correctness model checking of $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ on fair TPSs) *The modifications presented in Section 8.5.7 to the algorithm of Section 6.1.2 yield a correct algorithm for the model checking of the operators $\bar{\mathrm{P}}$ and $\bar{\mathrm{D}}$ on fair TPSs.*

Proof. The first step in proving the result consists in showing the correctness of the decomposition into maximal *admissible* end components of Step 1', as opposed to the decomposition in maximal end components of Step 1. This is justified by Theorem 8.3, which states that under an admissible policy, behaviors eventually follow an admissible end component with probability 1. A formal proof can be done by repeating the argument of Theorem 6.4 in the context of fair MDPs.

Next, consider a sub-MDP $\Pi_i = (S_i, A_i, p^i, R_i, W_i) \in \mathcal{L}$, and assume that $\bowtie \in \{\geq, >\}$; the case for $\bowtie \in \{\leq, <\}$ is similar. If $i \in K^-$, let $J_0 = \lambda_i^-$; otherwise, let $J_0 = -\infty$. We have to show that, for all $s \in S_i$, it is $\bar{\mathrm{FT}}^-_s = J_0$. To prove this result, it suffices to show that the following two assertions hold:

1. For any admissible policy η for Π_i, and any $s \in S_i$, it is
$$\Pr_s^\eta\Bigl(\liminf_{n\to\infty} \mathcal{H}_n(\omega) < J_0\Bigr) = 0 \; . \tag{8.21}$$

2. For any $s \in S_i$, it is
$$J_0 \geq \inf_{\eta\in Adm(s)} \Bigl\{ J \Bigm| \Pr_s^\eta\Bigl(\liminf_{n\to\infty} \mathcal{H}_n(\omega) = J\Bigr) = 1 \Bigr\} \; . \tag{8.22}$$

Assertion 1 follows immediately from the results of Chapter 6, since the set of admissible policies is a subset of the set of all policies. To prove Assertion 2, we proceed as follows. By Corollary 6.7, there is an unichain policy η_m such that
$$\Pr_s^{\eta_m}\Bigl(\lim_{n\to\infty} \mathcal{H}_n(\omega) = J_0\Bigr) = 1 \; .$$

If η_m is admissible, we are done. Otherwise, denote by $C \subseteq S_i$ the single closed recurrent class of η_m. Let η_f be the admissible Markovian policy which at every state $s \in S_i$ chooses an action from $A_i(s)$ uniformly at random, and for $0 \leq x < 1$, define the Markovian policy $\eta(x)$ by
$$Q_{\eta(x)}(a \mid s) = (1-x)\, Q_{\eta_m}(a \mid s) + x\, Q_{\eta_f}(a \mid s) \; ,$$

for all $s \in S_i$ and $a \in A_i(s)$. From this definition, we see that policy $\eta(x)$ is admissible for all $0 < x < 1$. Let $P(x)$ be the matrix of the Markov chain corresponding to $\eta(x)$, and define the vectors $\boldsymbol{R}(x) = [R_s(x)]_{s\in S_i}$ and $\boldsymbol{W}(x) = [W_s(x)]_{s\in S_i}$ by
$$R_s(x) = \sum_{a\in A_i(s)} R_i(s,a)\, Q_{\eta(x)}(a \mid s) \qquad\qquad W_s(x) = \sum_{a\in A_i(s)} W_i(s,a)\, Q_{\eta(x)}(a \mid s) \; ,$$

for all $s \in S_i$.

Note that the policy $\eta(x)$ is unichain for all $0 \leq x < 1$: its closed recurrent class is equal to C for $x = 0$, and to S_i for $0 < x < 1$. Pick an arbitrary state $s_0 \in C$: state s_0 is thus surely recurrent for all $0 \leq x < 1$. Hence, we can apply Theorem 8.12, obtaining as a result the continuity of $\boldsymbol{\pi}(x)$ for all $0 \leq x < 1$. The result (8.22) then follows from
$$\lim_{x\to 0} \frac{\boldsymbol{\pi}(x)\, \boldsymbol{R}(x)}{\boldsymbol{\pi}(x)\, \boldsymbol{W}(x)} = J_0$$

and from

$$\Pr_s^{\eta(x)}\left(\lim_{n\to\infty}\mathcal{H}_n(\omega)=\frac{\boldsymbol{\pi}(x)\,\boldsymbol{R}(x)}{\boldsymbol{\pi}(x)\,\boldsymbol{W}(x)}\right)=1\ .$$

The proof is completed by combining Assertions 1 and 2. ∎

Chapter 9

Verification of Stochastic Transition Systems

The previous chapter concluded our presentation of temporal logics and model-checking algorithms for TPSs. Armed with these results, we return to the specification and verification of stochastic transition systems (STSs), which has been informally described in Chapter 2.

The first step in this direction consists in defining a formal semantics for STSs. We define this semantics by means of a translation from STSs to fair TPSs. This translation is also used for the verification of STSs, since our model-checking algorithms can be applied to fair TPSs, rather than STSs. This translation is faithful to the informal semantics presented in Section 2.1.2: this result is of practical interest, since usually the model of a real system is constructed with that informal semantics in mind, rather than with the formal semantics defined by translation.

An alternative approach, not followed here, consists in formalizing the informal semantics of Section 2.1.2, taking that definition as basic. If we followed this approach, we would have to prove the correspondence between the formal semantics of STSs and the semantics of their translations into fair TPSs. Since in this dissertation we place the emphasis on logics and model-checking algorithms, rather than semantical arguments, we prefer to avoid this additional step. We rely instead on a precise but informal argument to justify our translation.

The specification of probabilistic properties of STSs relies on the logics SPTL and SPTL*. These logics differ from FPTL and FPTL* only for a minor change in the semantics. The change has been introduced to account for the characteristics of the translation process, which introduces auxiliary states that have no counterpart in the original STS. The model-checking algorithms described in the previous chapter can be used with minor adaptations to decide whether an STS satisfies specifications written in these new logics. We conclude the chapter with some remarks about the role of nondeterminism in STSs.

9.1 Translation of STSs into TPSs

The translation from STSs to fair TPSs is based on the informal semantics for STSs presented in Section 2.1.2. Given an STS $\mathcal{S} = (\mathcal{V}, \Theta, \mathcal{T})$, its translation $\Pi_{\mathcal{S}} = (S \cup \widehat{S}, A, p, \mathcal{F}, \widehat{S}, S_{in}, \mathit{time}, \mathcal{I})$ is a *translation TPS,* formed by a fair TPS $(S \cup \widehat{S}, A, P, \mathcal{F}, S_{in}, \mathit{time}, \mathcal{I})$ together with a distinguished subset $\widehat{S}$ of *auxiliary states.* We define the translation TPS $\Pi_{\mathcal{S}}$ by stages.

Principal states. As in Definition 2.1, the set $\mathcal{V}$ of typed variables of the STS gives rise to a set S of states consisting of all type-consistent interpretations of the variables in $\mathcal{V}$. Since in this dissertation we are discussing only the verification of finite-state systems, we assume that all the variables in $\mathcal{V}$ have finite domain, so that S is a finite set. We call S the set of *principal states* of $\Pi_{\mathcal{S}}$. In addition to S, the translation will introduce also the set $\widehat{S}$ of auxiliary states. For each $s \in S$, we let $\mathcal{I}_s[\![x]\!] = s[\![x]\!]$.

The set S_{in} of initial states is defined simply as $S_{in} = \{s \in S \mid s \models \Theta\}$, and consists thus of principal states only.

9.1.1 Transition Structure of Principal States

Let $\mathcal{T}(s) = \{\tau \in \mathcal{T} \mid s \models \mathcal{E}_\tau\}$ be the set of STS transitions enabled at a principal state $s \in S$. In analogy with the definition of informal semantics of STS of Section 2.1.2, the transition structure associated with s depends on whether $\mathcal{T}(s)$ contains immediate transitions or not.

Some Immediate Action Enabled

Let $\mathcal{T}_i(s) = \mathcal{T}(s) \cap \mathcal{T}_i$ be the set of immediate transitions enabled at s, and assume that $\mathcal{T}_i(s) \neq \emptyset$. In this case, we let $A(s) = \{a_\tau \mid \tau \in \mathcal{T}_i(s)\}$, where action a_τ represents the choice of transition τ at s. For all $\tau \in \mathcal{T}_i(s)$, we let $\mathit{time}(s, a_\tau) = 0$; moreover, action a_τ is fair at s iff τ is fair: formally, $a_\tau \in \mathcal{F}(s)$ iff $\tau \in \mathcal{T}_f$, for all $\tau \in \mathcal{T}_i(s)$.

For each $\tau \in \mathcal{T}_i(s)$, we add to $\widehat{S}$ the auxiliary states $(s, \langle \tau, j \rangle)$, where $1 \leq j \leq m_\tau$. Each auxiliary state $(s, \langle \tau, j \rangle)$ represents the choice of mode j of transition τ from state s. Since auxiliary states represent choices done while at a principal state, we let $\mathcal{I}_{(s,\langle \tau, j \rangle)}[\![x]\!] = s[\![x]\!]$, for $\tau \in \mathcal{T}_i(s)$ and $1 \leq j \leq m_\tau$.

For $\tau \in \mathcal{T}_i(s)$ and $1 \leq j \leq m_\tau$, the transition probability from s to $(s, \langle \tau, j \rangle)$ is defined by

$$p_{s,(s,\langle \tau, j \rangle)}(a_\tau) = p_j^\tau \ ,$$

where p_j^τ is the probability of mode j of transition τ. All other transition probabilities are 0.

No Immediate Transitions Enabled

If $\mathcal{T}(s) \subseteq \mathcal{T}_d$, we let

$$\mathcal{T}_e(s) = \mathcal{T}(s) \cap \mathcal{T}_e \qquad\qquad \mathcal{T}_u(s) = \mathcal{T}(s) \cap \mathcal{T}_u \ ;$$

note that $\mathcal{T}_e(s) \neq \emptyset$, due to the presence of the idling transition. We let

$$A(s) = \{a_e\} \cup \{a_\tau \mid \tau \in \mathcal{T}_u(s)\} \ ; \tag{9.1}$$

action a_e represents the choice of a transition with exponential distribution, while action a_τ represents the choice of the transition τ, with unspecified delay distribution.

We let $\mathcal{F}(s) = A(s)$, and we define the expected times of the actions by

$$time(s, a_e) = \left(\sum_{\tau' \in \mathcal{T}_e(s)} \gamma_{\tau'} \right)^{-1} \qquad\qquad time(s, a_\tau) = 0 \tag{9.2}$$

for all $\tau \in \mathcal{T}_e(s)$. These decisions will be justified once the presentation of the translation is completed.

For each $\tau \in \mathcal{T}(s)$, we add to $\widehat{S}$ the auxiliary states $(s, \langle \tau, j \rangle)$, where $1 \leq j \leq m_\tau$. Each auxiliary state $(s, \langle \tau, j \rangle)$ represents the choice of mode j of transition τ from state s. Since auxiliary states represent choices done while at a principal state, we let $\mathcal{I}_{(s,\langle \tau, j \rangle)}[\![x]\!] = s[\![x]\!]$, for $\tau \in \mathcal{T}(s)$ and $1 \leq j \leq m_\tau$.

The transition probabilities to these auxiliary states are defined as follows. For $\tau \in \mathcal{T}(s)$ and $1 \leq j \leq m_\tau$, we let

$$\tau \in \mathcal{T}_u(s): \qquad p_{s,(s,\langle \tau, j \rangle)}(a_\tau) = p_j^\tau \tag{9.3}$$

$$\tau \in \mathcal{T}_e(s): \qquad p_{s,(s,\langle \tau, j \rangle)}(a_e) = p_j^\tau \gamma_\tau \Bigg/ \left(\sum_{\tau' \in \mathcal{T}_e(s)} \gamma_{\tau'} \right), \tag{9.4}$$

where p_j^τ is the probability of mode j of transition τ. All other transition probabilities are 0.

9.1.2 Transition Structure of Auxiliary States

Consider an arbitrary auxiliary state $(s, \langle \tau, j \rangle) \in \widehat{S}$. This state represents the choice of mode j of action τ at principal state s. Let $T_j^\tau(s) = \{s' \in S \mid (s, s') \models \rho_j^\tau\}$ be the set of possible successor states of s according to this transition mode. We let $A(s, \langle \tau, j \rangle) = T_j^\tau(s)$, and

$$p_{(s,\langle \tau, j \rangle),t}(t) = 1 \qquad\qquad time(s, t) = 0$$

for all $t \in T_j^\tau(s)$, with all other transition probabilities from the auxiliary state being 0. None of the actions from the auxiliary state is fair: we set $\mathcal{F}(s, \langle \tau, j \rangle) = \emptyset$.

The presentation of the translation from STSs to fair TPSs is thus concluded.

9.1.3 Complexity of Translation and Optimizations

The following theorem relates the size of an STS to the size of its translation TPS.

Theorem 9.1 (size of translation TPS) *Given an STS $\mathcal{S} = (\mathcal{V}, \Theta, \mathcal{T})$, let $\Pi_\mathcal{S}$ be the translation TPS for $\mathcal{S}$. Denote by S the set of states arising from the set of variables $\mathcal{V}$, and let $M = \max_{\tau \in \mathcal{T}} m_\tau$. The size $|\Pi_\mathcal{S}|$ is then polynomial in $M \cdot |S| \cdot |\mathcal{T}|$.*

Proof. The result follows immediately from an exam of the translation process. ∎

To reduce the size of the translation TPS, it is possible to eliminate the idling transition from the set $\mathcal{T}(s)$ of transitions enabled at a state s, provided the set $\mathcal{T}(s)$ does not become empty. We have retained the idling transition in our definition of translation purely to simplify the definition.

Another optimization consists in avoiding the generation of auxiliary states that have only one successor state. More precisely, assume that for state $s \in S$, transition $\tau \in \mathcal{T}(s)$ and mode $1 \leq j \leq m_\tau$ there is only one state $t \in S$ such that $(s,t) \models \rho_j^\tau$. In this case, instead of creating the auxiliary state $(s, \langle \tau, j \rangle)$ having t as single successor, we can direct all transitions to this auxiliary state to the principal state t instead. This optimization succeeds in removing all auxiliary states corresponding to deterministic transitions.

9.1.4 An Example of Translation

Consider the TPS $\mathcal{S} = (\mathcal{V}, \Theta, \mathcal{T})$ with set of variables $\mathcal{V} = \{x\}$, initial condition $\Theta : x = 0$ and set of transitions $\mathcal{T} = \{\tau_1, \tau_2, \tau_3, \tau_4\}$. The domain of x is $\{0, 1, 2, 3\}$, and the transitions are as follows:

- Transition $\tau_1 \in \mathcal{T}_e$ has rate $\gamma_{\tau_1} = 2$ and $m_{\tau_1} = 1$. Its enabling transition and transition formula are, respectively, $\mathcal{E}_{\tau_1} : x < 3$ and $\rho_{\tau_1} : x' = x + 1$.

- Transition $\tau_2 \in \mathcal{T}_e$ has rate $\gamma_{\tau_2} = 1$ and $m_{\tau_2} = 1$. Its enabling transition and transition formula are, respectively, $\mathcal{E}_{\tau_2} : x > 0$ and $\rho_{\tau_2} : x' = x - 1$.

- Transition $\tau_3 \in \mathcal{T}_i$ is immediate, and it is $m_{\tau_3} = 1$. Its enabling transition and transition formula are, respectively, $\mathcal{E}_{\tau_3} : x = 0$ and $\rho_{\tau_3} : x' = 1 \vee x' = 2$.

- Transition $\tau_4 \in \mathcal{T}_u$ has unspecified rate, and it is $m_{\tau_4} = 1$. Its enabling transition and transition formula are, respectively, $\mathcal{E}_{\tau_4} : x = 3$ and $\rho_{\tau_4} : x' = 0$.

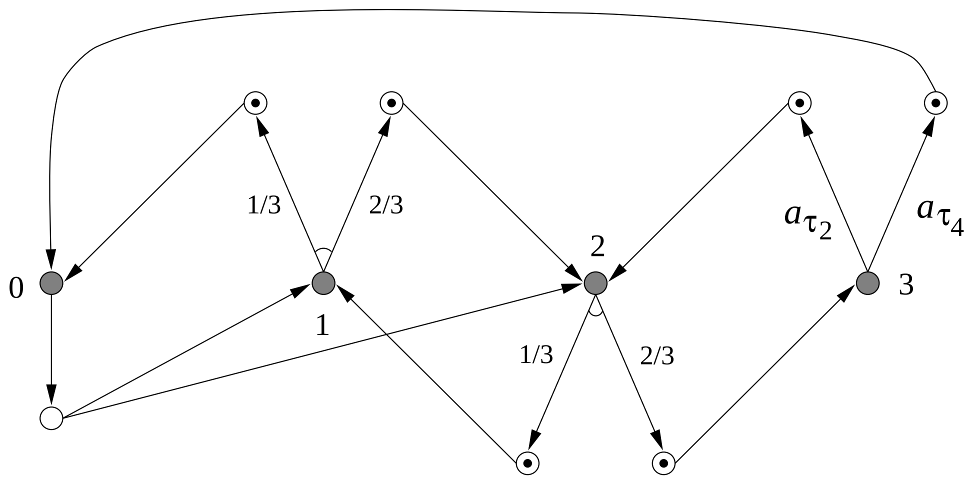

Figure 9.1: Translation TPS $\Pi_{\mathcal{S}}$. The principal states are depicted as filled circles, and are labeled with the corresponding value of x. The names of the auxiliary states have been omitted, and only the names of some actions have been indicated. The auxiliary states denoted by a dot have only one successor state, and can be eliminated by a simple optimization.

This STS represents a continuous-time random walk between the integers 0, 1, 2, and 3. If $x < 3$, there is an exponential transition with rate 2 that increases x; if $x > 0$ there is an exponential transition with rate 1 that decreases x. The random walk has boundaries 0 and 3. When boundary 0 is reached, there is an instantaneous transition that sets x to either 1 or 2. When boundary 3 is reached, alongside the transition with exponential distribution that decreases x is activated a transition with unspecified rate that sets x to 0.

The translation TPS Π_{Ψ} is depicted in Figure 9.1. To reduce clutter, we have used the optimization of omitting the idling transition, since there is at least one transition enabled on every state. Denote by s_i the principal state where $x = i$, for $0 \leq i \leq 3$. The expected times of the actions from s_3 are given by $\mathit{time}(s_3, a_{\tau_2}) = 1$ and $\mathit{time}(s_3, a_{\tau_4}) = 0$. The only action from state s_0 has expected time 0. The only actions from states s_1 and s_2 have expected times $1/3$. The expected times of actions from auxiliary states are always 0, by our definition of translation. In Figure 9.2 we depict the translation TPS obtained from the one of Figure 9.1 by applying the optimization of removing all auxiliary states having only one successor.

9.2 Translation and Informal Semantics

Even though the formal semantics of STSs is defined by translation into fair TPSs, there is a correspondence between the proposed translation and the informal semantics presented in Section 2.1.2.

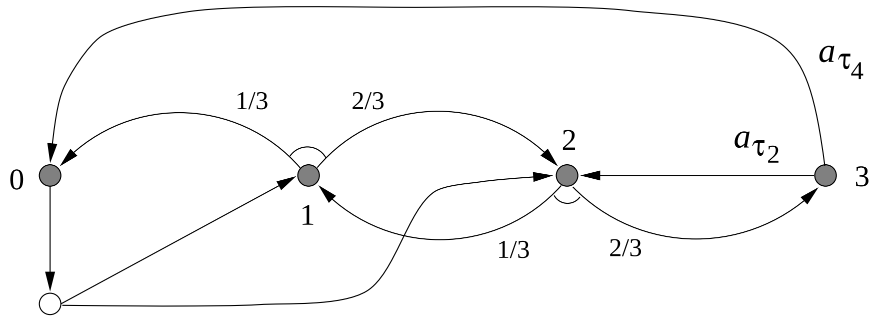

Figure 9.2: Result of removing all auxiliary states having only one successor from the TPS of Figure 9.1.

This correspondence is important from a pragmatic point of view, since the models of real probabilistic systems are usually constructed in terms of this intuitive semantics. We justify the translation in three steps, considering first the structure of the translation TPS, then the use of fairness, and lastly the interaction between translation and specification languages.

9.2.1 Structure of the Translation TPS

To understand the correspondence between the translation and the informal semantics, consider first auxiliary states. As remarked in the definition of translation, auxiliary states represent a choice of transition and transition mode from a principal state. The introduction of these states is necessary because of the different interaction between probability and nondeterminism in Markov decision processes and STSs. In a Markov decision process, the successor of a state is chosen in two steps, a nondeterministic one (the choice of action) and a probabilistic one (the choice of destination state). In an STS, there are up to three steps: the choice of transition (probabilistic or nondeterministic), the choice of transition mode (probabilistic), and finally the choice of destination state (nondeterministic). The role of auxiliary states is precisely to accommodate the three-step selection process of STSs into the two-step one of TPSs.

Consider now the system evolution from a principal state s. If there are immediate transitions enabled at s, the correspondence between the informal semantics of the STS and the translation TPS is immediate.

If the set $\mathcal{T}(s)$ of transitions enabled at s is a subset of the delayed transitions $\mathcal{T}_d$, let as before $\mathcal{T}_e(s) = \mathcal{T}(s) \cap \mathcal{T}_e$ and $\mathcal{T}_u(s) = \mathcal{T}(s) \cap \mathcal{T}_u$. When a behavior of the TPS reaches s, the set of available actions is $\{a_e\} \cup \{a_\tau \mid \tau \in \mathcal{T}_u(s)\}$. Let q_e and q_τ, for $\tau \in \mathcal{T}_u(s)$, be the probabilities with which these actions are chosen by a policy. Note that these probabilities can depend on the past history of the TPS.

There is a relation between the probabilities q_e and q_τ, $\tau \in \mathcal{T}_u$, selected by the policy, and the rates of the transitions in $\mathcal{T}_u(s)$, selected nondeterministically in the informal semantics. To derive the relation, consider the probability of choosing $\tau \in \mathcal{T}(s)$ in the TPS and in the informal semantics. From (9.3) and (9.4), the probability of choosing $\tau \in \mathcal{T}(s)$ in the TPS is given by

$$\tau \in \mathcal{T}_e(s): \qquad q_e \gamma_\tau \Big/ \Big(\sum_{\tau' \in \mathcal{T}_e(s)} \gamma_{\tau'} \Big) \tag{9.5}$$

$$\tau \in \mathcal{T}_u(s): \qquad q_\tau \ . \tag{9.6}$$

In the informal semantics, the probability is given by

$$\gamma_\tau \Big/ \Big(\sum_{\tau' \in \mathcal{T}(s)} \gamma_{\tau'} \Big) \tag{9.7}$$

for all $\tau \in \mathcal{T}(s)$, which is just a restatement of (2.2). Equating (9.7) with (9.5) and (9.6), we obtain

$$q_e = \Big(\sum_{\tau' \in \mathcal{T}_e(s)} \gamma_{\tau'} \Big) \Big/ \Big(\sum_{\tau' \in \mathcal{T}(s)} \gamma_{\tau'} \Big) \qquad\qquad q_\tau = \gamma_\tau \Big/ \Big(\sum_{\tau' \in \mathcal{T}(s)} \gamma_{\tau'} \Big) \tag{9.8}$$

for all $\tau \in \mathcal{T}_u(s)$.

This relation between q_e, $\{q_\tau\}_{\tau \in \mathcal{T}_u(s)}$ and $\{\gamma_\tau\}_{\tau \in \mathcal{T}_u(s)}$ preserves not only the probabilities of choosing the transitions at s, but also the expected time spent at s. In fact, from (9.2) the expected time spent by the TPS at s is given by

$$q_e \Big/ \Big(\sum_{\tau' \in \mathcal{T}_e(s)} \gamma_{\tau'} \Big) \ . \tag{9.9}$$

If we substitute into this equation the value of q_e given by (9.8), we obtain

$$\frac{\displaystyle\sum_{\tau' \in \mathcal{T}_e(s)} \gamma_{\tau'}}{\Big(\displaystyle\sum_{\tau' \in \mathcal{T}(s)} \gamma_{\tau'} \Big) \Big(\displaystyle\sum_{\tau' \in \mathcal{T}_e(s)} \gamma_{\tau'} \Big)} = \Big(\sum_{\tau' \in \mathcal{T}(s)} \gamma_{\tau'} \Big)^{-1} ,$$

which is exactly the expected time spent at s under the informal STS semantics, as defined by (2.1).

Thus, equations (9.8) together with the constraint $q_e + \sum_{\tau \in \mathcal{T}_u(s)} q_\tau = 1$ define a one-to-one mapping between the unspecified transition rates in the informal STS semantics and the probabilities of choosing the actions in the translation TPS. The mapping preserves both the expected time spent at s, and the probabilities of selecting transitions from s. Given a nondeterministic choice for the transition rates $\{\gamma_\tau\}_{\tau \in \mathcal{T}_u(s)}$, we can determine a policy which simulates this choice; conversely, each policy can be interpreted as a choice for these rates. This correspondence indicates that the proposed translation of STS into fair TPS preserves the informal semantics presented in Section 2.1.2.

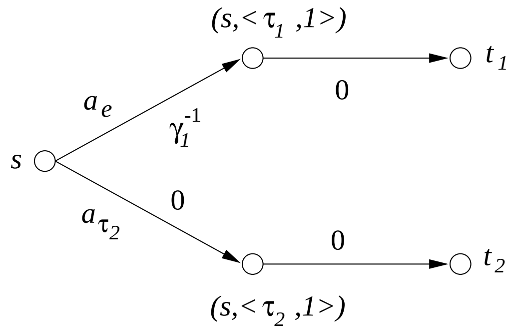

Figure 9.3: Portion of translation TPS corresponding to states s, t_1, t_2 and transitions τ_1, τ_2. Each edge is labeled by the expected time corresponding to the source state and the action.

9.2.2 Translation and Fairness

The above considerations also justify our use of fairness in the translation, and more generally our definition of probabilistic fairness.

In fact, for $\tau \in \mathcal{T}_u(s)$ the fairness of a_τ requires that $q_\tau > 0$, which by (9.8) corresponds to the requirement $\gamma_\tau > 0$. Similarly, the fairness of a_e requires that $q_e > 0$, which corresponds to the requirement $\gamma_\tau < \infty$ for all $\tau \in \mathcal{T}_u(s)$. Thus, the fairness requirements associated with principal states in the translation TPS are the exact counterpart of the requirements $0 < \gamma_\tau < \infty$ for the rates of transitions $\tau \in \mathcal{T}_u$.

From this we see that fairness gives us precisely the tool needed to represent transitions with unspecified delay distributions in our translation TPS. This consideration has been one of the main motivations for our new definition of probabilistic fairness.

9.2.3 Translation and Specification Language

We conclude our justification of the translation with a remark about the relationship between the translation and the specification languages presented in this dissertation.

In (9.2), we assign an expected time 0 to the actions that correspond to transitions with unspecified rates. This assignment is justified by our previous considerations on the equivalence of choosing the transition rates in the STS, and choosing the policy in the TPS. However, the justification is incomplete without the observation that the specification languages we presented can only refer to the expected time spent at a state, *not conditional on the successor state.*

To clarify this point, consider a state s on which two transitions are enabled: a transition τ_1, with rate γ_1, and a transition τ_2, with unspecified rate. The two transitions have only one mode, and they lead to states t_1 and t_2, respectively. The portion of translation TPS for s, τ_1, τ_2 is depicted in Figure 9.3. The translation would be inappropriate if our specification languages could express properties such as:

the time spent at s *when* $(s, \langle \tau_2, 1 \rangle)$ *is the immediate successor is on average* $> b$.

In fact, for the purposes of this property the choice of assigning $time(s, a_{\tau_2}) = 0$ would not correspond to the idea of assigning nondeterministically a transition rate to τ_2.

However, an exam of the semantical definitions of our specification languages reveals that the languages cannot specify properties in which the expected time is measured conditionally on the successor state. The key definitions are definition (4.15) and (8.1) of semantics of D, and Definitions 5.5 and 8.6 of product between TPS and D-experiment.

- In (4.15) and (8.1), for any $k \geq 0$, the fact that the time $time(X_k, Y_k)$ belongs to the summation depends only on the sequence of states $X_0 \cdots X_k$, and not on the successor state X_{k+1}.
- In Definitions 5.5 and 8.6, consider a state $\langle s, u \rangle$ of the product MDP. If u is a timed vertex, all actions from $\langle s, u \rangle$ contribute to r; if u is untimed, none of the actions does. Hence, the contribute of the r label does not depend on the successor state of $\langle s, u \rangle$.

These observations conclude our justification of the translation process.

9.3 Specification and Verification of Stochastic Transition Systems

For the specification of probabilistic properties of STSs, we introduce the new logics SPTL and SPTL*. The difference between these logics and the fair probabilistic logics FPTL and FPTL* is limited to the semantics of state formulas, and is motivated by the presence of auxiliary states in the translation TPS. The definition is as follows.

Definition 9.1 (semantics of SPTL, SPTL* formulas) Let $\Pi = (S \cup \widehat{S},\ A,\ p,\ \mathcal{F},\ S_{in},\ time,\ \mathcal{I},\ \widehat{S})$ be a translation TPS, where $\widehat{S}$ is the set of auxiliary states, and consider a state formula ϕ of SPTL or SPTL*.

The truth value of ϕ at a principal state $s \in S - \widehat{S}$ is defined using the same clauses of FPTL and FPTL*. The truth value of ϕ at an auxiliary state $(s, \langle \tau, j \rangle) \in \widehat{S}$ is defined by

$$(s, \langle \tau, j \rangle) \models \phi \quad \textit{iff} \quad s \models \phi\,. \tag{9.10}$$

Thus, the truth value of a state formula at an auxiliary state is equal to the truth value of the same state formula at the corresponding principal state. ∎

We say that an STS satisfies a specification iff its translation TPS does.

Definition 9.2 (satisfaction of specification over STS) Consider an STS $\mathcal{S} = (\mathcal{V}, \Theta, \mathcal{T})$ and its translation TPS $\Pi_{\mathcal{S}} = (S \cup \widehat{S}, A, P, \mathcal{F}, S_{in}, time, \mathcal{I}, \widehat{S})$. Given a specification $\phi \in \mathit{Stat}$ of SPTL or SPTL*, we say that $\Pi_{\mathcal{S}} \models \phi$ iff $s \models \phi$ for every $s \in S_{in}$, and we say that $\mathcal{S} \models \phi$ iff $\Pi_{\mathcal{S}} \models \phi$. ∎

To understand the rationale for Definition 9.1, note that under the semantics for FPTL and FPTL* it is possible for a state formula to assume different truth values on a principal state and on the related auxiliary states. For example, the expected time to reach a given set of states can be different when measured from a principal state, or from an auxiliary state where the next transition has already been selected. If our definition of semantics took into account the truth values of state formulas at auxiliary states, the logic would present often counterintuitive and undesirable properties. The following example illustrates this situation.

Example 9.1 Consider an STS $\mathcal{S} = (\mathcal{V}, \Theta, \mathcal{T})$, where:

- $\mathcal{V} = \{x\}$, and the domain of x is $\{0, 1\}$;
- $\Theta : x = 0$;
- $\mathcal{T} = \mathcal{T}_e = \{\tau_0\}$, where the exponential-delay transition τ_0 is defined by enabling condition $\mathcal{E}_{\tau_0} : \mathit{true}$, transition rate $\gamma_{\tau_0} = 1$, number of modes $m_{\tau_0} = 1$, and transition relation $\gamma_{\tau_0} : x' = 1 - x$.

Informally, the STS $\mathcal{S}$ describes a system that oscillates between states 0 and 1, and whose switching time from one state to the other obeys an exponential distribution with unit rate. Since the exponential distribution is memoryless, if the system is in state 0, the expected amount of time needed to reach state 1 is 1. Thus, we expect $\mathcal{S}$ to satisfy the FPTL specification $\phi_0 : \mathrm{A}\Box[x = 0 \rightarrow \mathrm{D}_{\geq 0.7}(x = 1)]$.

However, consider the translation TPS $\Pi_{\mathcal{S}}$ depicted in Figure 9.4. To understand its structure, recall that to any STS is implicitly added the idling transition τ_{idle}. From state s_0, the expected time to a state where $x = 1$ is 1, so that $\mathrm{D}_{\geq 0.7}(x = 1)$ holds at s_0. However, the expected time to $x = 1$ from state $(s_0, \langle \tau_0, 1 \rangle)$ is 0, since the only action from this auxiliary state leads to s_1 and has expected time 0. This implies that $\mathrm{D}_{\geq 0.5}(x = 1)$ does not hold at $(s_0, \langle \tau_0, 1 \rangle)$ under the normal definition of semantics, leading to $\mathcal{S} \not\models \phi_0$.

This example demonstrates that the truth value of state formulas at auxiliary states is not necessarily related to any property of the physical system being modeled. In this very simple example, the problem is eliminated if we apply the optimization of removing the auxiliary states having only one successor, but this of course is no solution in general. If our definition of semantics took into account these truth values, our logics would have scarce use as specification languages. Under the semantics provided by Definition 9.1, the truth values of state formulas at auxiliary states are disregarded, and we have $\mathcal{S} \models \phi_0$ as expected. ∎

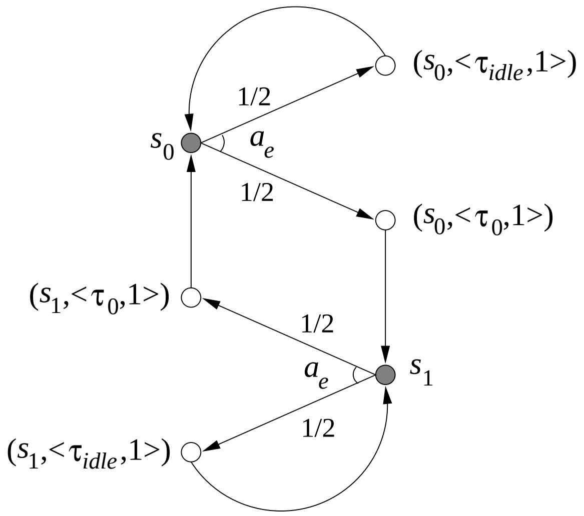

Figure 9.4: Translation TPS $\Pi_{\mathcal{S}}$ obtained from the STS described in Example 9.1. The principal states are s_0, s_1, and are depicted as filled circles; the other states are auxiliary states. Variable x assumes value 0 at s_0 and at the two auxiliary corresponding to s_0, and value 1 at the other states; the initial state is s_0. It is $time(s_0, a_e) = time(s_1, a_e) = 1/2$; the expected times of the other state-action pairs are 0.

Verification of Stochastic Transition Systems

Due to the similarity between the semantics of SPTL, SPTL* and FPTL, FPTL*, the algorithms that have been presented in the previous chapter for the model checking of FPTL, FPTL* specifications can be immediately adapted to SPTL and SPTL*.

To decide the truth value of a specification $\phi \in \mathit{Stat}$ at all initial states of a translation TPS $\Pi_{\mathcal{S}}$, we recursively evaluate the truth values of the state subformulas of ϕ at all states, until the truth value of ϕ itself can be computed at all initial states. With respect to the algorithms for FPTL and FPTL*, the only change consists in the fact that the truth value of state formulas at auxiliary states is defined by (9.10), rather than being computed directly. This modification does not change the complexity of the model-checking algorithms, which share the same complexity bounds of those for FPTL and FPTL*.

9.4 A Highway Commuter Example

Having concluded the presentation of our methodology for formal specification and verification of systems, we present a final example. Our system, HIGHWAY-REPAIR, is shown in Figure 9.5, and models a commuter that travels perennially back and forth between two cities A and B, passing through an intermediate city C.

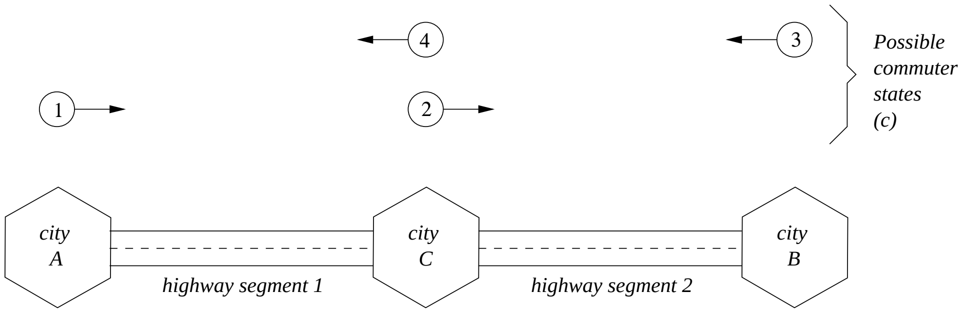

Figure 9.5: System HIGHWAY-REPAIR. Depicted are also the 4 possible states of the commuter, represented by the values 1..4 of variable c.

Each of the two highway segment from A to C and from C to B can be in one of three states: *good*, *broken* and *repair*, modeling a good stretch of highway, one with pot-holes in it, and one under repair. For a highway segment, the transition from *good* to *broken* has rate γ_{gb}, and the transition from *repair* to *good* has rate γ_{rg}. The transition from state *broken* to state *repair,* instead, has unspecified delay distribution: the criteria with which the organizations in charge of road maintenance decide to start repair works are not known to the layperson.

As depicted in Figure 9.5, the commuter can be at one of 4 states, depending on which segment must be traversed next, and in which direction. Depending on the state in which the next link is, the commuter traverses the link with rate γ_g, γ_b or γ_r.

The STS $\Pi = (\mathcal{V}, \Theta, \mathcal{T})$ for this example has set of variables $\mathcal{V} = \{l_1, l_2, c\}$, where l_1 and l_2 have domain $\{good, broken, repair\}$ and represents the link status, and c has domain $\{1, 2, 3, 4\}$ and represents the position of the commuter. The initial condition is Θ : *true*. Note that the initial condition is not important, since we are interested in the steady-state behavior of the system, which does not depend on the initial condition, as all states are reachable from all other states.

The transitions are divided in two groups: the transitions that describe the status of the highway links, and the transitions that describe the commuter. All of these transitions have only one mode, so that for every $\tau \in \mathcal{T}$ it is $m_\tau = 1$ and $p_1^\tau = 1$. To avoid clutter, when describing transition τ we write $\mathcal{E}$ instead of $\mathcal{E}_1^\tau$, and similarly for the other components.

Highway link transitions. The transitions for the highway links are as follows, for $i = 1, 2$.

- Transition $\tau_{gb}^i \in \mathcal{T}_d$ has rate γ_{gb}, and it has $\mathcal{E} : l_i = good$ and $\rho : l_i' = broken$.

- Transition $\tau_{br}^i \in \mathcal{T}_u$ has $\mathcal{E} : l_i = broken$ and $\rho : l_i' = repair$.

- Transition $\tau_{rg}^i \in \mathcal{T}_d$ has rate γ_{rg}, and it has $\mathcal{E} : l_i = repair$ and $\rho : l_i' = good$.

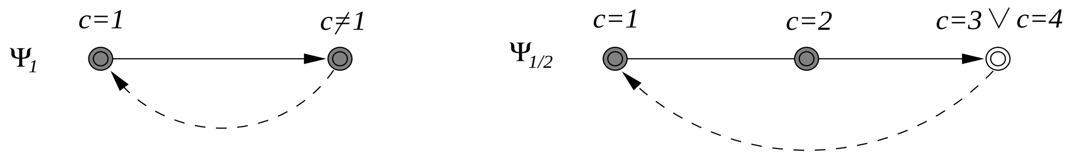

Figure 9.6: Experiments for the specification of one-way trip and round-trip commuter times in system HIGHWAY-REPAIR. Experiment Ψ_1 measures the round-trip time, experiment $\Psi_{1/2}$ the one-way trip time from A to B. All edges of the two experiments are timed, and thus shown in boldface.

Commuter transitions. There are three transitions for the commuter, depending on the condition of the link to be traversed.

- Transition $\tau_g^c \in \mathcal{T}_d$ has rate γ_g, and it has

$$\mathcal{E} : [(c = 1 \vee c = 4) \wedge l_1 = \mathit{good}] \vee [(c = 2 \vee c = 3) \wedge l_2 = \mathit{good}] \qquad c' = (c \bmod 4) + 1 \; .$$

- Transition $\tau_r^c \in \mathcal{T}_d$ has rate γ_r, and it has

$$\mathcal{E} : [(c = 1 \vee c = 4) \wedge l_1 = \mathit{repair}] \vee [(c = 2 \vee c = 3) \wedge l_2 = \mathit{repair}] \qquad c' = (c \bmod 4) + 1 \; .$$

- Transition $\tau_b^c \in \mathcal{T}_d$ has rate γ_b, and it has

$$\mathcal{E} : [(c = 1 \vee c = 4) \wedge l_1 = \mathit{broken}] \vee [(c = 2 \vee c = 3) \wedge l_2 = \mathit{broken}] \qquad c' = (c \bmod 4) + 1 \; .$$

The set of transitions of the STS is thus

$$\mathcal{T} = \left\{ \tau_{gb}^1, \tau_{br}^1, \tau_{rg}^1, \tau_{gb}^2, \tau_{br}^2, \tau_{rg}^2, \tau_g^c, \tau_b^c, \tau_r^c \right\} \; .$$

The TPS resulting from this STS, after the simple optimizations, consists of $3^2 \cdot 4 = 36$ states. The specifications we write for this TPS refer to the average time required for the commuter to go from A to B, and for the commuter to perform a round-trip. These specifications are written with the help of the experiments depicted in Figure 9.6. The two specifications we write are:

$$\bar{\mathrm{D}}_{\leq 14}(\Psi_1) \qquad \bar{\mathrm{D}}_{\leq 7}(\Psi_{1/2}) \; .$$

These specifications require that the time required for a round-trip (resp. halftrip) is on average less than 14 (resp. 7).

Consider the case in which the transition rates are given by:

$$\gamma_{gb} = 0.05 \qquad \gamma_{rg} = 0.1 \qquad \gamma_g = 0.5 \qquad \gamma_r = 0.3 \qquad \gamma_b = 0.1 \; .$$

Using the model-checking algorithms presented in the dissertation, it can be decided that all states s of Π are such that

$$s \models \bar{\mathrm{D}}_{\leq 14}(\Psi_1) \qquad s \not\models \bar{\mathrm{D}}_{\leq 7}(\Psi_{1/2}) \; .$$

Thus, the long-run average duration of a round-trip is guaranteed to be at most 14 time units, but the long-run average duration of a one-way trip is *not* guaranteed to be at most 7 time units! The model-checking algorithm enables to establish that, approximately, the threshold outcomes are

$$\bar{\mathrm{FT}}^{-}_{s}(\Psi_1) \simeq 11.7774 \qquad \bar{\mathrm{FT}}^{-}_{s}(\Psi_{1/2}) \simeq 5.5458$$
$$\bar{\mathrm{FT}}^{+}_{s}(\Psi_1) \simeq 13.5986 \qquad \bar{\mathrm{FT}}^{+}_{s}(\Psi_{1/2}) \simeq 7.5526$$

for all states s of Π. Intuitively, this can be explained as follows. The policy η_1 that maximizes the round-trip time of the commuter tries to slow down the commuter along the entire round-trip, without discriminating the first from the second part of the round-trip. On the other hand, the policy $\eta_{1/2}$ that maximizes the one-way time concentrates the slowdown on the first part, from city A to B, without making slowdown efforts during the return part of the trip. Since the slowdown in this latter case is concentrated on the part of trip that is being monitored by the experiment, the time required to complete it will be on average more than half of the time required to complete a round-trip.

9.5 Concluding Remarks

We would like to conclude this presentation with some remarks on the use of nondeterminism in the modeling and specification of probabilistic systems. We take as starting points the HIGHWAY-REPAIR system, and the example of the token ring system presented in Chapter 2.

Nondeterminism can have different functions in a system model. For example, it can be used to model unknown schedulers, and unspecified "black-box" algorithms that can be implemented. This was the case in the token-ring example, where we used nondeterminism to model the leader-election algorithm and the recoveries from failure. Another use of nondeterminism, as we have seen in this chapter, is to represent transitions having unknown distributions of waiting times.

Regarding this last use of nondeterminism, it is important to realize that nondeterminism not only implies that the waiting-time distribution of the system is unspecified, but also that the distribution might in fact depend on the past history of the system, or on the current state of the system — including portions of the state that might not seem at first related to the transition. For example, in the last system we modeled the highway-segment transition from *broken* to *repair* by a transition with unspecified distribution. This implies that the organization in charge of highway maintenance can decide how to schedule the repair works as a function of the system state (our results from Chapter 6, together with the forms of experiments Ψ_1 and $\Psi_{1/2}$, imply that the pessimal strategies need not depend on the past).

It seems quite reasonable to allow the repair policy to depend on the state of the links: this makes it possible to model facts such as a preference, on the organization's part, for repairing at most one segment at a time. On the other hand, it seems at first rather unreasonable to enable the policy to depend on where the commuter is. It is as if the repair organization, to decide whether to begin repair works, looks at where a specific unlucky commuter is! Of course, many people feel that this is exactly what must be happening to them when they travel. Beyond this pessimistic explanation, however, there is a more complicated picture.

It might indeed be unreasonable to expect that the repair policy is based on the position of a single unlucky commuter. However, it is possible that there are *external* events that influence both the commuter and the repair policy. For example, the commuter might start the journey from city A to B always in the mornings, and the organization might also prefer to start repair work in the mornings.

As these considerations demonstrate, the use of nondeterminism in the construction of a system model is a complex issue. In this dissertation, we have taken the approach of granting the highest possible degree of freedom to nondeterminism: nondeterministic choices can be randomized, and can depend on the full history of the system. This approach guarantees that system models are a conservative approximation of real systems: if a system model can be shown to satisfy a specification, the real system will also satisfy it. It would be very interesting, nonetheless, to examine intermediate concepts of nondeterminism, in which the policy is not known in advance, but not completely unconstrained either.

Bibliography

[ABC84] M. Ajmone Marsan, G. Balbo, and G. Conte. A class of generalized stochastic Petri nets for the performance analysis of multiprocessor systems. *ACM Trans. Comp. Sys.*, 2(2):93–122, May 1984.

[ABC+94] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Genralized Stochastic Petri Nets.* John Wiley & Sons, 1994.

[ACD90] R. Alur, C. Courcoubetis, and D. Dill. Model-checking for real-time systems. In *Proc. 5th IEEE Symp. Logic in Comp. Sci.*, 1990.

[ACD91] R. Alur, C. Courcoubetis, and D.L. Dill. Model-checking for probabilistic real-time systems. In *Proc. 18th Int. Colloq. Aut. Lang. Prog.*, volume 510 of *Lect. Notes in Comp. Sci.*, pages 115–126. Springer-Verlag, 1991.

[ACD92] R. Alur, C. Courcoubetis, and D. Dill. Verifying automata specifications of probabilistic real-time systems. In *Real Time: Theory in Practice*, volume 600 of *Lect. Notes in Comp. Sci.*, pages 28–44. Springer-Verlag, 1992.

[AD90] R. Alur and D. Dill. Automata for modeling real-time systems. In *Proc. 17th Int. Colloq. Aut. Lang. Prog.*, volume 443 of *Lect. Notes in Comp. Sci.*, pages 322–335. Springer-Verlag, 1990.

[AFK88] K.R. Apt, N. Francez, and S. Katz. Appraising fairness in languages for distributed programming. *Distributed Computing*, 2:226–241, 1988.

[ASB+95] A. Aziz, V. Singhal, F. Balarin, R.K. Brayton, and A.L. Sangiovanni-Vincentelli. It usually works: The temporal logic of stochastic systems. In *Computer Aided Verification*, volume 939 of *Lect. Notes in Comp. Sci.* Springer-Verlag, 1995.

[Bai96] C. Baier. Polynomial time algorithms for testing probabilistic bisimulation and simulation. In *Computer Aided Verification*, volume 1102 of *Lect. Notes in Comp. Sci.*, pages 38–49. Springer-Verlag, 1996.

[Bal95] G. Balbo. On the success of stochastic Petri nets. In *Proc. 6th Int. Workshop on Petri Nets and Performance Models*, pages 2–9, 1995.

[BAPM83] M. Ben-Ari, A. Pnueli, and Z. Manna. The temporal logic of branching time. *Acta Informatica*, 20:207–226, 1983.

[BBG95] M. Bernardo, N. Busi, and R. Gorrieri. A distributed semantics for EMPA based on stochastic contextual nets. *Computer J.*, 38(7):492–509, 1995.

[BBS92] J.C.M. Baeten, J.A. Bergstra, and S.A. Smolka. Axiomatizing probabilistic processes: ACP with generative probabilities. In *CONCUR'92: Concurrency Theory. 3rd Int. Conf.*, volume 630 of *Lect. Notes in Comp. Sci.* Springer-Verlag, 1992.

[BdA95] A. Bianco and L. de Alfaro. Model checking of probabilistic and nondeterministic systems. In *Found. of Software Tech. and Theor. Comp. Sci.*, volume 1026 of *Lect. Notes in Comp. Sci.*, pages 499–513. Springer-Verlag, 1995.

[BDG94a] M. Bernardo, L. Donatiello, and R. Gorrieri. MPA: a stochastic process algebra. Technical Report UBLCS-94-10, Laboratory for Computer Science, University of Bologna, May 1994.

[BDG94b] M. Bernardo, L. Donatiello, and R. Gorrieri. Operational GSPN semantics of MPA. Technical Report UBLCS-94-12, Laboratory for Computer Science, University of Bologna, May 1994.

[BDG95] M. Bernardo, L. Donatiello, and R. Gorrieri. Giving a net semantics to Markovian process algebra. In *Proc. 6th Int. Workshop on Petri Nets and Performance Models*, pages 169–178. IEEE Comput. Soc. Press, 1995.

[Bel57] R.E. Bellman. *Dynamic Programming.* Princeton University Press, 1957.

[Ber87] D.P. Bertsekas. *Dynamic Programming.* Prentice-Hall, 1987.

[Ber95] D.P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, 1995. Volumes I and II.

[BG96] M. Bernardo and R. Gorrieri. Extended Markovian process algebra. In *CONCUR'96: Concurrency Theory. 7th Int. Conf.*, volume 1119 of *Lect. Notes in Comp. Sci.*, pages 315–330. Springer-Verlag, 1996.

[BH97] C. Baier and H. Hermanns. Weak bisimulation for fully probabilistic processes. In *Computer Aided Verification*, volume 1254 of *Lect. Notes in Comp. Sci.*, pages 119–130, June 1997.

[BK84] J.A. Bergstra and J.W. Klop. Process algebra for synchronous communication. *Information and Computation*, 60:109–137, 1984.

[BKR+97] C. Baier, M. Kwiatkowska, M. Ryan, E. Clarke, and V.Hartonas Garmhausen. Symbolic model checking for probabilistic processes. In *Proc. 24th Int. Colloq. Aut. Lang. Prog.*, 1997.

[BLM97] N.S. Bjørner, U. Lerner, and Z. Manna. Deductive verification of parameterized fault-tolerant systems: A case study. In *Intl. Conf. on Temporal Logic*. Kluwer, 1997. To appear.

[BS96] D. Beauquier and A. Slissenko. Polytime model checking for timed probabilistic computation tree logic. Technical Report TR-96-08, Dept. of Informatics, Univ. Paris-12, April 1996.

[BT91] D.P. Bertsekas and J.N. Tsitsiklis. An analysis of stochastic shortest path problems. *Math. of Op. Res.*, 16(3):580–595, 1991.

[CC91] L. Christoff and I. Christoff. Efficient algorithms for verification of equivalences for probabilistic processes. In *Computer Aided Verification*, volume 575 of *Lect. Notes in Comp. Sci.*, pages 310–321. Springer-Verlag, 1991.

[CE81] E.M. Clarke and E.A. Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In *Proc. Workshop on Logic of Programs*, volume 131 of *Lect. Notes in Comp. Sci.*, pages 52–71. Springer-Verlag, 1981.

[CFZ96] E. Clarke, M. Fujita, and X. Zhao. *Multi-Terminal Binary Decision Diagrams and Hybrid Decision Diagrams, Representations of Discrete Functions*, pages 93–108. Kluwer Academic Publishers, 1996.

[CM96] K. Seidel C. Morgan, A. McIver. Probabilistic predicate transformers. *ACM Trans. Prog. Lang. Sys.*, 18(3):325–353, May 1996.

[CMP92] E. Chang, Z. Manna, and A. Pnueli. The safety-progress classification. In *Logic, Algebra, and Computation*, NATO ASI Series, Subseries F: Computer and System Sciences. Springer-Verlag, Berlin, 1992.

[CMT91] G. Ciardo, J.K. Muppala, and K.S. Trivedi. On the solution of GSPN reward models. *Performance Evaluation*, 12:237–253, 1991.

[CSZ92] R. Cleaveland, S.A. Smolka, and A. Zwarico. Testing preorders for probabilistic processes. In *Proc. 19th Int. Colloq. Aut. Lang. Prog.*, Lect. Notes in Comp. Sci., pages 708–719. Springer-Verlag, 1992.

[CY88] C. Courcoubetis and M. Yannakakis. Verifying temporal properties of finite-state probabilistic programs. In *Proc. 29th IEEE Symp. Found. of Comp. Sci.*, 1988.

[CY90] C. Courcoubetis and M. Yannakakis. Markov decision processes and regular events. In *Proc. 17th Int. Colloq. Aut. Lang. Prog.*, volume 443 of *Lect. Notes in Comp. Sci.*, pages 336–349. Springer-Verlag, 1990.

[CY95] C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *J. ACM*, 42(4):857–907, July 1995.

[dA97] L. de Alfaro. Temporal logics for the specification of performance and reliability. In *Proc. of Symp. on Theor. Asp. of Comp. Sci.*, volume 1200 of *Lect. Notes in Comp. Sci.*, pages 165–176. Springer-Verlag, February 1997.

[DC64] J.S. De Cani. A dynamic programming algorithm for embedded Markov chains when the planning horizon is at infinity. *Management Sci.*, 10:710, 1964.

[Den70] E.V. Denardo. Computing a bias-optimal policy in a discrete-time Markov decision problem. *Op. Res.*, 18:279–289, 1970.

[Der63] C. Derman. Stable sequential control rules and Markov chains. *J. of Math. Anal. and Appl.*, 6:257–265, 1963.

[Der64] C. Derman. On sequential control processes. *Ann. Math. Stat.*, 35:341–349, 1964.

[Der70] C. Derman. *Finite State Markovian Decision Processes.* Acedemic Press, 1970.

[DNH84] R. De Nicola and M. Hennessy. Testing equivalences for processes. *Theoretical Computer Science*, 34:83–133, 1984.

[Doo94] J.L. Doob. *Measure Theory.* Springer-Verlag, 1994.

[DRH95] S. Donatelli, M. Ribaudo, and J. Hillston. A comparison of performance evaluation process algebra and generalized Petri nets. In *Proc. 6th Int. Workshop on Petri Nets and Performance Models*, pages 158–168. IEEE Comput. Soc. Press, 1995.

[DS66] C. Derman and R. Strauch. A note on memoryless rules for controlling sequential control processes. *Ann. Math. Stat.*, 37:276–278, 1966.

[EH85] E.A. Emerson and J.Y. Halpern. “sometimes” and “not never” revisited: on branching versus linear time temporal logic. *J. ACM*, 33(1):151–178, 1985.

[EL85] E.A. Emerson and C.L. Lei. Modalities for model checking: Branching time strikes back. In *Proc. 12th ACM Symp. Princ. of Prog. Lang.*, pages 84–96, 1985.

[Eme90] E.A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, chapter 16, pages 995–1072. Elsevier Science Publishers (North-Holland), Amsterdam, 1990.

[Fel83] Y.A. Feldman. A decidable propositional probabilistic dynamic logic. In *Proc. 15th ACM Symp. Theory of Comp.*, pages 298–309, 1983.

[FH82] Y.A. Feldman and D. Harel. A probabilistic dynamic logic. In *Proc. 14th ACM Symp. Theory of Comp.*, pages 181–195, 1982.

[Fra86] N. Francez. *Fairness*. Springer-Verlag, 1986.

[GH94] S. Gilmore and J. Hillston. The PEPA workbench: A tool to support a process algebra-based approach to performance modelling. In *7th Int. Conf. on Modelling Techniques and Tools for Computer Performance Evaluation*, May 1994.

[GHR93] H.N. Götz, U. Herzog, and M. Rettelbach. Multiprocessor and distributed system design: the integration of functional specification and performance analysis using stochastic process algebras. In *PERFORMANCE'93*, volume 729 of *Lect. Notes in Comp. Sci.* Springer-Verlag, 1993.

[Han94] H. Hansson. *Time and Probabilities in Formal Design of Distributed Systems*. Real-Time Safety Critical Systems Series. Elsevier, 1994.

[Hil95] J. Hillston. Compositional Markovian modelling using a process algebra. In *Second Int. Work. on Numerical Solution of Markov Chains*. Kluwer Academic Press, January 1995. Book title: Computations with Markov Chains.

[Hil96] J. Hillston. *A Compositional Approach to Performance Modelling*. Distinguished Dissertations Series. Cambridge University Press, 1996.

[HJ89] H. Hansson and B. Jonsson. A framework for reasoning about time and reliability. In *Proc. of Real Time Systems Symposium*, pages 102–111. IEEE, 1989.

[HJ94] H. Hansson and B. Jonsson. A logic for reasoning about time and probability. *Formal Aspects of Computing*, 6(5):512–535, 1994.

[HK97] M. Huth and M. Kwiatkowska. Quantitative analysis and model checking. In *Proc. 12th IEEE Symp. Logic in Comp. Sci.*, pages 111–122, 1997.

[HMS96] J. He, A.K. McIver, and K. Seidel. Probabilistic models for the guarded command language. *Sci. Comput. Program.*, 1996. In FMTA'95 special issue.

[How60] H. Howard. *Dynamic Programming and Markov Processes*. MIT Press, 1960.

[How63] R.A. Howard. Semi-Markovian decision problems. In *Intern. Stat. Inst.*, Ottawa, Canada, 1963.

[How71] R. Howard. *Dynamic Probabilistic Systems: Semi-Markov and Decision Systems*, volume II. Wiley, 1971.

[HS82] D.P. Heyman and M.J. Sobel. *Stochastic Models in Operations Research*, volume I. McGraw-Hill, 1982.

[HS84a] S. Hart and M. Sharir. Probabilistic temporal logic for finite and bounded models. In *Proc. 16th ACM Symp. Theory of Comp.*, pages 1–13, 1984.

[HS84b] D.P. Heyman and M.J. Sobel. *Stochastic Models in Operations Research*, volume II. McGraw-Hill, 1984.

[HS86] S. Hart and M. Sharir. Probabilistic propositional temporal logics. *Information and Computation*, 70(2–3):97–155, 1986.

[HSP82] S. Hart, M. Sharir, and A. Pnueli. Termination of probabilistic concurrent programs. In *Proc. 9th ACM Symp. Princ. of Prog. Lang.*, pages 1–6, 1982.

[HSP83] S. Hart, M. Sharir, and A. Pnueli. Termination of probabilistic concurrent programs. *ACM Trans. Prog. Lang. Sys.*, 5(3):356–380, July 1983.

[HSP84] S. Hart, M. Sharir, and A. Pnueli. Verification of probabilistic programs. *SIAM J. Computing*, 13(2):292–314, May 1984.

[Jen53] A. Jensen. Markov chains as an aid to the study of Markov processes. *Skand. Aktuariedtishr.*, 36:87–91, 1953.

[JHW94] B. Jonsson, C. Ho-Stuart, and Y. Wang. Testing and refinement for nondeterministic and probabilistic processes. In *Proc. of Symp. on Formal Techniques in Real-Time and Fault-Tolerant Systems*, volume 863 of *Lect. Notes in Comp. Sci.*, pages 418–430. Springer-Verlag, 1994.

[JL91] B. Jonsson and K.G. Larsen. Specification and refinement of probabilistic processes. In *Proc. 6th IEEE Symp. Logic in Comp. Sci.*, pages 266–277, 1991.

[JS90] C.-C. Jou and S.A. Smolka. Equivalences, congruences and complete axiomatizations for probabilistic processes. In *CONCUR'90: Concurrency Theory. 1st Int. Conf.*, volume 458 of *Lect. Notes in Comp. Sci.*, pages 367–383. Springer-Verlag, 1990.

[JW95] B. Jonsson and Y. Wang. Compositional testing preorders for probabilistic processes. In *Proc. 10th IEEE Symp. Logic in Comp. Sci.*, pages 431–441, June 1995.

[KB96] M. Kwiatkowska and C. Baier. Model checking for a probabilistic branching time logic with fairness. Technical Report CSR-96-12, University of Birmingham, June 1996.

[Koz79] D. Kozen. Semantic of probabilistic programs. In *Proc. 20th IEEE Symp. Found. of Comp. Sci.*, pages 101–114, 1979. Also appeared in the *Journal of Computer and System Sciences*, vol. 22, 328–350 (1981).

[Koz83] D. Kozen. A probabilistic PDL. In *Proc. 15th ACM Symp. Theory of Comp.*, pages 291–297, 1983.

[KP95] O. Kupferman and A. Pnueli. Once and for all. In *Proc. 10th IEEE Symp. Logic in Comp. Sci.*, pages 25–35, 1995.

[KSK66] J.G. Kemeny, J.L. Snell, and A.W. Knapp. *Denumerable Markov Chains.* D. Van Nostrand Company, 1966.

[KT75] S. Karlin and H.M. Taylor. *A First Course in Stochastic Processes.* Academic Press, 1975. Second edition.

[Lam77] L. Lamport. Proving the correctness of multiprocessor programs. *IEEE Trans. Software Engin.*, 3:125–143, 1977.

[Lam83] L. Lamport. What good is temporal logic? In R.E.A. Mason, editor, *Proc. IFIP 9th World Congress*, pages 657–668. Elsevier Science Publishers (North-Holland), 1983.

[LDPK95] M.L. Littman, T.L. Dean, and L. Pack Kaelbling. On the complexity of solving Markov decision problems. In *Uncertainty in Artificial Intelligence. Proceedings of the 11th Conference*, pages 394–402, 1995.

[LPS81] D. Lehmann, A. Pnueli, and J. Stavi. Impartiality, justice and fairness: the ethics of concurrent termination. In *Proc. 8th Int. Colloq. Aut. Lang. Prog.*, volume 115 of *Lect. Notes in Comp. Sci.*, pages 264–277. Springer-Verlag, 1981.

[LR81] D. Lehmann and M. Rabin. On the advantage of free choice: a symmetrical and fully distributed solution to the dining philosophers problem. In *Proc. 8th ACM Symp. Princ. of Prog. Lang.*, pages 133–138, 1981.

[LS82] D. Lehman and S. Shelah. Reasoning with time and chance. *Information and Control*, 53(3):165–198, 1982.

[LS89] K.G. Larsen and A. Skou. Bisimulation through probabilistic testing (preliminary report). In *Proc. 16th ACM Symp. Princ. of Prog. Lang.*, 1989.

[LS91] K.G. Larsen and A. Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94(1):1–28, 1991. Preliminary report in POPL'89.

[LS92] K.G. Larsen and A. Skou. Compositional verification of probabilistic processes. In W.R. Cleaveland, editor, *CONCUR'92: Concurrency Theory. 3rd Int. Conf.*, volume 630 of *Lect. Notes in Comp. Sci.* Springer-Verlag, 1992.

[LSS94] N. Lynch, I. Saias, and R. Segala. Proving time bounds for randomized distributed algorithms. In *Proc. of Symp. on Principles of Distributed Computing*, pages 314–323, 1994.

[LT87] N.A. Lynch and M. Tuttle. Hierarcical correctness proofs for distributed algorithms. In *Proc. 6th ACM Symp. Princ. of Dist. Comp.*, 1987.

[MC94] M. Melekopoglou and A. Condon. On the complexity of the policy improvement algorithm for Markov decision processes. *ORSA Journal on Computing*, 6(2):188–192, 1994.

[McN66] R. McNaughton. Testing and generating infinite sequences by a finite automaton. *Information and Computation*, 9:521–530, 1966.

[Mil83] R. Milner. Calculi for synchrony and asynchrony. *Theoretical Computer Science*, 25:267–310, 1983.

[Mil90] R. Milner. Operational and algebraic semantics of concurrent processes. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 1202–1242. Elsevier Science Publishers (North-Holland), Amsterdam, 1990.

[MMP92] O. Maler, Z. Manna, and A. Pnueli. From timed to hybrid systems. In *Proc. of the REX Workshop "Real-Time: Theory in Practice"*, volume 600 of *Lect. Notes in Comp. Sci.*, pages 447–484. Springer-Verlag, 1992.

[Mol81] M.K. Molloy. *On the Integration of Delay and Throughput Measure In Distributed Processing Models.* PhD thesis, UCLA, Los Angeles, 1981.

[Mol82] M.K. Molloy. Performance analysis using stochastic Petri nets. *IEEE Trans. on Computers*, C-31(9):913–917, September 1982.

[MP91] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification.* Springer-Verlag, New York, 1991.

[MP93] Z. Manna and A. Pnueli. Models for reactivity. *Acta Informatica*, 30:609–678, 1993.

[MP95] Z. Manna and A. Pnueli. *Temporal Verification of Reactive Systems: Safety.* Springer-Verlag, New York, 1995.

[Nat80] S. Natkin. *Les Reseaux de Petri Stochastiques et leur Application a l'Evaluation des Systemes Informatiques.* PhD thesis, CNAM, Paris, 1980.

[Pet] Proc. of the international workshop on Petri nets and performance models, 1987–.

[Pnu83] A. Pnueli. On the extremely fair treatment of probabilistic algorithms. In *Proc. 15th ACM Symp. Theory of Comp.*, pages 278–290, 1983.

[Pro] Proc. of international workshop on process algebra and performance modelling, 1993–.

[PS95] A. Pogosyants and R. Segala. Formal verification of timed properties of randomized distributed algorithms. In *Proc. of Symp. on Principles of Distributed Computing*, pages 173–183, 1995.

[PT87] C.H. Papadimitriou and J.N. Tsitsiklis. The complexity of Markov decision processes. *Math. of Op. Res.*, 12(3):441–450, August 1987.

[Put94] M.L. Puterman. *Markov Decision Processes.* John Wiley and Sons, 1994.

[PZ86] A. Pnueli and L. Zuck. Probabilistic verification by tableaux. In *Proc. First IEEE Symp. Logic in Comp. Sci.*, pages 322–331, 1986.

[PZ93] A. Pnueli and L.D. Zuck. Probabilistic verification. *Information and Computation*, 103:1–29, 1993.

[QS83] J.P. Queille and J. Sifakis. Fairness and related properties in transition systems — A temporal logic to deal with fairness. *Acta Informatica*, 19:195–220, 1983.

[Rab63] M.O. Rabin. Probabilistic automata. *Information and Computation*, 6:230–245, 1963.

[Ram80] L. Ramshaw. *Formalizing the Analysis of Algorithms.* PhD thesis, Stanford University, 1980.

[Rei80] J.H. Reif. Logic for probabilistic programming. In *Proc. 12th ACM Symp. Theory of Comp.*, pages 8–13, 1980.

[Rib95] M. Ribaudo. Stochastic Petri net semantics for stochastic process algebras. In *Proc. 6th Int. Workshop on Petri Nets and Performance Models*, pages 148–157. IEEE Comput. Soc. Press, 1995.

[Ros70a] S.M. Ross. *Applied Probability Models with Optimization Applications.* Holden-Day, San Francisco, California, 1970.

[Ros70b] S.M. Ross. Average cost semi-Markov decision processes. *J. Appl. Prob.*, 7:649–656, 1970.

[Saf88] S. Safra. On the complexity of ω-automata. In *Proc. 29th IEEE Symp. Found. of Comp. Sci.*, 1988.

[Saf92] S. Safra. Exponential determinization for ω-automata with strong-fairness acceptance condition. In *Proc. ACM Symp. Theory of Comp.*, pages 275–282, 1992.

[Sch71] P.J. Schweitzer. Iterative solution of the functional equations of undiscounted Markov renewal programming. *J. Math. Anal. Appl.*, 34:495–501, 1971.

[Sch87] A. Schrijver. *Theory of Linear and Integer Programming.* J. Wiley & Sons, 1987.

[Seg95a] R. Segala. A compositional trace-based semantics for probabilistic automata. In Springer-Verlag, editor, *CONCUR'95: Concurrency Theory. 6th Int. Conf.*, Lect. Notes in Comp. Sci., pages 234–248, 1995.

[Seg95b] R. Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems.* PhD thesis, MIT, June 1995. Technical Report MIT/LCS/TR-676.

[Seg96] R. Segala. Testing probabilistic automata. In *CONCUR'96: Concurrency Theory. 7th Int. Conf.*, Lect. Notes in Comp. Sci., pages 299–314. Springer-Verlag, 1996.

[SL94] R. Segala and N.A. Lynch. Probabilistic simulations for probabilistic processes. In *CONCUR'94: Concurrency Theory. 5th Int. Conf.*, volume 836, pages 481–496. Springer-Verlag, 1994.

[SL95] R. Segala and N.A. Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250–273, 1995.

[SS71] D. Scott and C. Strachey. Towards a mathematical semantics for computer languages. Technical Report PRC6, Oxford Univ., 1971.

[Str66] R.E. Strauch. Negative dynamic programming. *Annals of Math. Stat.*, 37(4), 1966.

[SV66] R. Strauch and A.F. Veinott. *A Property of Sequential Control Processes.* Rand McNally, Chicago, Illinois, USA, 1966.

[Sym80] F.J.W. Symons. Introduction to numerical Petri nets, a general graphical model for concurrent processing systems. *Australian Telecommunications Research*, 14(1):28–33, January 1980.

[Tho90] W. Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, chapter 4, pages 135–191. Elsevier Science Publishers (North-Holland), Amsterdam, 1990.

[Tij86] H.C. Tijms. *Stochastic Modelling and Analysis, a Computational Approach.* Wiley, 1986.

[Var85] M.Y. Vardi. Automatic verification of probabilistic concurrent finite-state systems. In *Proc. 26th IEEE Symp. Found. of Comp. Sci.*, pages 327–338, 1985.

[Vei69] A.F. Veinott. On discrete dynamic programming with sensitive discount optimality criteria. *Ann. Math. Stat.*, 40:1635–1660, 1969.

[vGSST90] R. van Glabbeek, S.A. Smolka, B. Steffen, and C.M.N. Tofts. Reactive, generative, and stratified models of probabilistic processes. In *Proc. 5th IEEE Symp. Logic in Comp. Sci.*, 1990.

[vGSST95] R. van Glabbeek, S.A. Smolka, B. Steffen, and C.M.N. Tofts. Reactive, generative, and stratified models of probabilistic processes. *Information and Computation*, 121(1):59–80, 1995. Preliminary report in LICS'90.

[VW86] M.Y. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification. In *Proc. First IEEE Symp. Logic in Comp. Sci.*, pages 332–344, 1986.

[Wil91] D. Williams. *Probability With Martingales.* Cambridge University Press, 1991.

[WL92] Y. Wang and K. Larsen. Testing probabilistic and nondeterministic processes. In *Protocol Specification, Testing, and Verification XII*, 1992.

[WSS94] S.-H. Wu, S.A. Smolka, and E.W. Stark. Composition and behaviors of probabilistic I/O automata. In Springer-Verlag, editor, *CONCUR'94: Concurrency Theory. 5th Int. Conf.*, Lect. Notes in Comp. Sci., pages 511–528, 1994.

[YCDS94] S. Yuen, R. Cleaveland, Z. Dayar, and S.A. Smolka. Fully abstract characterizations of testing preorders for probabilistic processes. In *CONCUR'94: Concurrency Theory. 5th Int. Conf.*, Lect. Notes in Comp. Sci., pages 497–512. Springer-Verlag, 1994.