UNIVERSIDADE DE LISBOA
INSTITUTO SUPERIOR TÉCNICO

Interplay between Probabilistic and Temporal Reasoning:

Applications to the Analysis and Synthesis of Complex Systems

Manuel Biscaia Martins

Supervisor: Doctor Paulo Alexandre Carreira Mateus

Thesis specifically prepared to obtain the PhD Degree in Information Security

Draft

July 2015

Resumo

Com o propósito de obter uma melhor compreensão sobre raciocínios formais que combinam asserções temporais com argumentos probabilísticos, consideram-se extensões temporais de uma lógica probabilística proposicional EPPL; esta lógica serve facilmente de alicerce para raciocínios probabilísticos, e é baseada numa lógica probabilística fundamental proposta por Fagin et al. O trabalho é fortemente motivado pela necessidade de verificar e sintetizar sistemas, protocolos ou programas críticos de forma automática. Em particular, no contexto de anÃ¡lise de protocolos, o raciocínio probabilístico permite a análise de protocolos com componentes probabilísticos e providencia forma de obter limites quantitativos sobre a correcção de um protocolo, modelando apropriadamente as capacidades de um atacante. Pretende-se pois usar as lógicas propostas para melhorar as capacidades de verificação/sintetização de sistemas formais já conhecidos, ou mostrar as limitações inerentes à sua análise, já que muitas técnicas clássicas comumente usadas não podem ser utilizadas devido a problemas de indecidibilidade.

Em Lógica Temporal, a introdução de quantificação proposicional reveste-se de uma importância relevante, uma vez que aumenta o poder expressivo da lógica temporal de base com consequeências teóricas e práticas; não só a quantificação proposicional na Lógica Temporal Linear (LTL) ou na Lógica Temporal Ramificada (CTL) enriquece estas lógicas para o nível da Lógica Monádica de Segunda Ordem, mas permite também expressar propriedades relativas a alternâncias muito comuns que de outra forma não seriam expressáveis.

Neste trabalho, mostra-se que a lógica EPPL admite eliminação de quantificadores proposicionais. Este resultado é de importância fulcral quando se considera extensões temporais de EPPL, já que espelha os resultados conhecidos sobre Lógica Proposicional, que é a lógica base tanto da CTL como da LTL. Relativamente às extensões temporais consideradas, propõe-se e

investiga-se uma extensão baseada no Cálculo μ, mostrando que a verificação de fórmulas de um seu fragmento (EPLTL) é equiredutível a um problema em aberto em Teoria de Números; contudo, considerando versões aproximadas destes problemas consegue-se obter um procedimento algorítmico que, para pontos de vista práticos, resolve adequadamente o problema da verificação. Mostra-se ainda que a lógica EPLTL com quantificação proposicional é decidível, usando para tal uma generalização das técnicas usadas classicamente para a lógica temporal linear quantificada.

Palavras-chave Lógica Probabilística, Lógica Temporal, Cadeias de Markov, Verificação de Modelos, Problemas de Satisfatibilidade.

Abstract

Hoping to obtain a better understanding of formal reasonings that combine assertions about time with probability arguments, we investigate and extend a probabilistic propositional logic EPPL that quite readily serves as a foundation for probabilistic reasoning, based on a seminal probabilistic logic proposed by Fagin et al. Our work is motivated by the need to verify and synthesize critical complex systems, protocols and programs in automatic fashion. Namely, in the context of protocol analysis, probabilistic reasoning enables the analysis of protocols that have a probabilistic component, providing also a method to obtain quantitative bounds on the correctness of a protocol, by adequately modeling the adversary capabilities. To that extent, our aim is to use our extended logics in order to either enhance the verification and synthesis capabilities of preexisting formal languages, or to show the limitations inherent in some verification/synthesis problems, as in the probabilistic setting many classical used techniques become unavailable due to undecidability issues.

Propositional quantification in temporal logic is a matter of paramount importance, as it increases the expressiveness of the base temporal logic with both theoretical and practical consequences; not only propositional quantification in Linear Temporal Logic (LTL) or Computation Tree Logic (CTL) brings both logics to the level of Monadic Second-order Logic, but also it allows one to express extremely natural properties regarding alternations that otherwise would be inexpressible.

In this work, we show that EPPL admits elimination of propositional quantifiers, which is an extremely important result when considering the temporal extensions of EPPL, as it mimics the well-known results about Propositional Logic, which is the base logic for both CTL and LTL. Regarding the temporal extensions of EPPL, we propose and investigate an extension based on the μ-Calculus, showing that the verification problem of one of its important fragments (EPLTL) over Markov chains is Turing equireducidble

to a long-standing and well-known open problem in Number Theory; nevertheless, by considering approximate versions of these problems we obtain an algorithmic procedure that adequately addresses all the practical aspects of this verification problem. Finally, we also show that EPLTL extended by propositional quantifiers is also decidable, based on a new generalization of fundamental techniques used in quantified linear propositional logic.

Keywords: Probabilistic Logic, Temporal Logic, Markov chains, Model Checking, Satisfiability problems.

Acknowledgments

First of all, I would like to thank my supervisor Professor Paulo Mateus whose guidance has made this work possible. I would also like to thank my co-authors, especially David Henriques for the long and eye-opening discussions we had throughout my PhD program and Pedro Baltazar for the brief but invaluable initial supervision.

I would like to thank everyone at the Security and Quantum Information Group at IT for their fellowship and support.

I also would like to thank all my research fellows, specially the Atónitos Anónimos group, for the possibility of sharing and discussing my ideas.

I would also like to thank FCT for the funding they provided throughout four years, thanks to the PhD scholarship with reference:

SFRH / BD / 73315 / 2010.

Finally, I would like to thank you, kind reader, for allowing me to show you a labor of hardship and love.

Contents

List of Figures

List of Tables

Introduction

Technological progress moves inexorably, as long as our knowledge of Nature and ourselves becomes more precise and adequate to explain the phenomena we experience. The adequacy of knowledge is particularly relevant. One does hope that in order to explain a certain social phenomena one does not need to explain it using the laws of quantum mechanics; this is a basic example to argue that while Science has a strong reductionist point of view, it also relies on an holistic perspective. The behavior of the system depends on the behaviors of each component, but the interdependence law is as relevant to the study of the system as understanding each individual component.

Even though the behavior of an automated system can, in theory, be replicated by a human, in practice it can not, as an automated system performs many types of operations much faster than a human. As evermore complex systems are needed, the human understanding of their behavior becomes increasingly holistic. However, one can also use automation to distill a degree of understanding about the target automated system, or even automatically design more complex systems.

A system is called *correct* if its behaviors are precisely the ones intended by the designer. Of course, depending on the application of the system, different degrees of correctness are required, which will consequently imply different methods of verification and/or specification.

Software verification is an integral part of software development. It is estimated that 30% to 50% of the costs in the development of a software are used in to validate its behavior; in the case of hardware, these costs are even more exacerbated. This verification consists mainly in code review by other programmers and software testing. However, in some critical applications, a greater degree of confidence in whether the system is correct is required, either because of the high costs involved in the repair of faulty behaviors, or where testing may be impossible or too expensive or even in cases where it is completely impractical to correct the fault, for instance correcting an error in the hardware of a spacecraft far from the Earth's orbit. Model

Checking is an advanced verification technique used successfully in many critical hardware/software systems with the intent of obtaining the utmost certainty concerning the correctness of a system.

Model Checking begins by constructing a *model* of the system capturing in rigorous form the relevant inner workings of the components and their interdependence. Already this modeling process can already uncover a great deal of inconsistencies that remain hidden during the informal specification of the system. One should note however that the analysis performed will be only as useful as quality of the model so obtained. At the same time, we need to formalize the properties that we wish to validate in the model.

Given a model of a system, and a formal specification of the desired property one then proceeds to analyze whether the formal specification of the property holds in the model. This phase is done automatically by a *model-checker*; in fact Model Checking as a software verification technique obtains its name from the mathematical concept behind the algorithm used in this phase. If the model does verify the the property, we can proceed; if it does not, the model-checker is usually able to provide a counter-example to the formal specification, that is a behavior of the model that does not satisfy the specification; with that counter-example one can simulate the actual system and check whether the system does have a faulty behavior. If it does, one then corrects the system and consequently the model. If the system does not have in fact that faulty behavior, then either the model was not adequate, or the specification property needs to be adapted. Either way, the process is restarted. There is also the possibility that the model-checker does not terminate successfully due to memory/time restrictions. In this case, one needs to construct an abstraction of the original model where the model-checker can be run successfully.

In the end, after all the desired properties are verified, one can be absolutely sure that the model does in fact behave appropriately, that is, the model is *correct*. Of course, the information about the degree of correctness of the system is only as good as the adequateness of the model considered.

Automated methodologies can also be used effectively as a tool to design more complex systems; even though verification and design might seem uncorrelated at first glance, they are in fact two faces of the same coin. Suppose that we have some system that is described by a certain model, but the system/model is still only partially built, that is, it allows many more behaviors than the desired ones. Suppose we want to further restrict the behavior of the model by ensuring that the resulting model obeys a certain property. One way of obtaining this more complex model is to consider a suitable composition between the partially defined model and an appropriate structure

representing the formula. The resulting model is then guaranteed to behave strictly within the bounds of the specification. Interestingly enough many model-checkers already function in this way; they verify if a certain property holds in a model by composing the *negation* of the property with the model and testing whether there exists an allowed behavior in the composition. If it does, then there exists an undesired behavior in the original system.

There exists however an important point that has to be addressed. We need to choose a suitable language able to formalize properties relevant for verification/specification. Moreover, we need to show that a certain property can be verified in a model *effectively* and *efficiently*. Since the seminal paper of **Pnuelli**, Temporal Logic has been used as a formal language to specify properties of reactive non-terminating discrete systems.We note that there are several successful applications of classical temporal model-checkers like [82, 39] in the context of protocol analysis that demonstrate the potential inherent in temporal reasoning to this field. Propositional temporal logic allows us to reason about the evolution of the truth values of propositions across time; however, there is no single propositional temporal logic, as different temporal logics allow us to express different classes of behaviors. This is expected, as different perspectives on time should change the set of valid temporal reasonings. We present a short introduction to Temporal logic in Chapter 1.

Logics for reasoning about probabilistic assertions abound, and in recent years have been increasingly combined with classical temporal logics. The first mention of probabilistic and temporal logics applied to the verification of systems comes from **Vardi**. The main drive for combining temporal and probabilistic reasoning is twofold. Not only by considering models with probabilistic components we can tame the presence of non-determinism, but also it allows us to obtain provable and absolute bounds about the security of the system. In Chapter 2, we introduce the fundamental probabilistic logic we use in this work (EPPL - Exogenous Probabilistic Propositional Logic), inspired by **Fagin et al.**, and developed in [78]; in Chapter 3, we approach the most common probabilistic models and discuss the logics used in their verification.

The main motivation for this work was to increase our understanding about reasonings that couple assertions about time and probability. In order to achieve this goal, we study a meta-property of EPPL, which has a direct impact when we consider the temporalized versions of EPPL. However, we will also investigate further the inherent limitations of probabilistic systems, or the limitations in their analysis. All these issues are tackled in the chapters of Part 2. Some of the chapters are adapted versions of the

articles published on these topics by myself and my co-authors.

We summarize the main contributions of this thesis:

- Following the work of **Baltazar** [8], we study the probabilization of quantified linear temporal logic (QLTL), which yields an extension of the most commonly used logic for verification of probabilistic systems (pCTL/pCTL*). We show that the logic is decidable by providing a satisfiability algorithm, and a Hibert calculus which is sound and (weakly) complete. The work was designed to be applied to Markov chains, which are deterministic probabilistic systems; however, we feel that the adaptation of the work to Markov decision processes is entirely viable. This work is presented in Chapter 4.

- In order to study the temporalization of the probabilistic propositional logic EPPL, we show EPPL admits elimination of propositional quantification. This new result is important as it not only gives us a better perspective on EPPL, but it mirrors the results known in the case of propositional logic (PL), its quantification (QPL), and its temporal extensions (LTL and QLTL, respectively). This work is presented in Chapter 5.

- Extending the work of **Baltazar et al.**, we consider a temporalization of EPPL using as temporal logic the μ-calculus. We provide a sound and (weakly) complete Hilbert calculus, and show that the logic μ-EPPL is decidable, and that its model-checking problem is also decidable; we argue that a syntactic restriction of μ-EPPL is useful in the automated design of planning strategies with probabilistic uncertainty. This work, presented on Chapter 6 becomes even more realistic when we consider the results shown in Chapter 8.

- We study the verification problem of EPLTL over Markov chains. Markov chains induce infinite EPLTL models, and as such the decidability of the model-checking problem does not apply; we show that the verification problem of a syntactic restriction of EPLTL is (Turing) equireducible to a well known open problem in Number Theory, the Skolem problem. Nevertheless, by considering a relaxation of the Skolem problem we show that the corresponding relaxed verification problem becomes decidable. This work appears in Chapter 7.

Part I

Introductory Concepts

Chapter 1

Temporal Logic

Temporal reasoning has its roots deep in classical Philosophy, being one of the first examples of modal reasoning. Temporal Logic allows us to reason about the evolution of propositions through time. Although it is in fact often seen as a subarea of Modal Logic, due to the usage of *modes* of reasoning dealing with time, Temporal Logic blossomed into a field on its own in the late seventies. This was due to an extremely important work by **Pnueli** [89], highlighting the relevance of temporal logic to the analysis of non-terminating cyclic programs and protocols. While the analysis of sequential (hopefully) terminating procedures could be done (partially) by the use of Hoare calculus for instance, the analysis of concurrent systems or non-terminating systems like an operating system was not developed in any way. Temporal logic as a subject in our days still is a research field linking the Computer Science community and the Logic community. Today, we have available a plethora of different temporal logics with three different application purposes in mind:

- verification of systems: in this paradigm, the logic is used for specification of a relevant property that the system should verify; after constructing an abstraction of the system, the resulting model is then checked to guarantee that the desired property holds. This paradigm is usually known as Model Checking [5, 7, 15, 26, 28, 30, 54, 57, 68, 104]. The mathematical concept critical to this perspective is the notion of satisfaction relation $M \Vdash \phi$;

- specification of systems: in this paradigm, the system is specified by temporal logic formula that captures all the intended behaviors of the system. Assuming that this formula is satisfiable, there are algorithms

that will completely describe (one of) the (many) possible implementation(s) of the system [16, 55, 107, 91, 70]. The mathematical concept critical to this perspective is the notion of satisfiability, that is whether there exists M such that $M \Vdash \phi$. This is usually linked with the entailment relation, that is, in most common temporal logics $\neg\phi$ is a valid formula ($\models \phi$) if and only if ϕ is satisfiable;

- automated/interactive theorem proving: this paradigm usually allows the user to use more expressive logics specifying both the system and the desired properties in much more detail; the reasoning with appropriate deduction rules allows for checkable proofs, if the system does not verify the desired properties, the procedure can be used to obtain additional hypotheses should be considered [88, 87]. In this case the relevant notion is the existence of a sound calculus, a purely syntactic reasoning tool such that $\vdash \phi$ implies $\models \phi$.

Still, when faced with the intent of guaranteeing that in some system (or model of a system) a certain relevant property holds, there is no clear answer on how to approach the problem. Even though the modeling the system may seem straightforward, one needs to take into account what type of property we would wish to ensure.

Moreover, one also needs to take into account the computational complexity of each of the algorithms used. For instance, while model-checking a transition system against a CTL formula is an *easy* problem, the use of CTL satisfiability (SAT) algorithm may be extremely costly. This may restrict the use of some logics in certain situations.

There are some general guidelines we can consider in order to understand which temporal logic is more adequate for the problem at hand. In this case we follow the review article of **Emerson** [33] and provide some of the most distinguishing factors between temporal logic.

Discrete Time versus Continuous Time

This axis of comparison is particularly interesting; while most temporal logics considered for reasoning about programs or protocols assume that time is discrete, as there exist a/many well defined next step(s) operation in these contexts, sometimes we are interested in considering the length of time that some task requires. Moreover, we may also be interested in systems undergoing a continuous dynamical evolution, acting under discrete control. In these cases a truly continuous temporal logic may be required.

Notable examples of discrete time logic include Propositional Linear Temporal Logic (LTL [89, 95]), Computation Tree Logic (CTL [35]), or the more general μ-Calculus (μ-TL [65, 67]). Examples of continuous time logic are Interval Temporal Logic (ITL [81]) or Metric Temporal Logic (MTL [64]).

Branching Time versus Linear Time

While our perception of time is intuitively linear, and in fact most iterative programs are well described as a single processing path, when dealing with concurrent/distributed/interactive processes, time can be seen as progressing in a branching-like fashion. While linear time logic can be used to verify properties of branching time systems, it has a strong focus on (computational) paths, branching time logics can capture other type of branching time substructures.

The most common linear time logic is LTL; it is mirrored by the branching time logic by CTL; although over standard structures they are expressively incomparable, the more powerful version CTL* [35, 36] encompasses both. For further discussion about the merits of each viewpoint consult [103].

Propositional versus First-order reasoning

First-order reasoning allows extra expressiveness; this fact implies that on one hand, we can specify more complex systems and properties, however even if the logic at hand is decidable, the algorithms' running time is usually prohibitive. Moreover, first-order temporal logic is undecidable and not even semi-decidable. These facts generally point in the direction of propositional based logics, possibly augmented with features relevant for the problem at hand.

However, purely propositional temporal logic, while quite well adapted to the verification of circuits/hardware or control protocols, it is not as suited to the verification of software due to the infinite state space usually required; only in recent years there has been some significant advances in the field. See [52] for a gentle introduction to First-order Temporal Logic.

1.1 Temporal logics used in this work

As in our work we will deal with many temporal logics, it is necessary to establish a general framework and notation in order for the work to remain consistent and reader-friendly throughout the manuscript.

The main difficulty in setting up this framework is the fact that temporal logics can be seen from alternating viewpoints. When approaching Temporal Logic in general we can work with a framework closely related with Modal Logic, using Kripke structures and modal operators; this approach benefits from the fact that transition systems model quite appropriately the behaviors of systems like hardware circuits. However a complementary approach focuses on describing the temporal logic in question as a first-order or second-order logic, with suitable signatures. This perspective is underlined in automata-theoretical views of Temporal Logic, and allows deep connections with Language Theory.

Before presenting the main temporal logics used throughout this work we will present some definitions (adapted to our purposes) relevant for the logics considered thereafter.

Definition 1.1.1 (Kripke Structure)**.** A *Kripke structure* m over the propositional symbols Λ is a tuple $m = (S, R, L)$ such that:

- S is a set, whose members are usually called *states*; we usually assume that S is countable;
- R is a total relation between states, that is $R \subseteq S \times S$ such that $\forall x \exists y R(x, y)$, usually called the *transition relation*;
- L is a map between S and $\{0, 1\}^{\Lambda}$, associating each state to a valuation of propositional symbols, usually called the *labeling function.*

We may require that a distinguished state $s^0 \in S$ is chosen; in this case we will denote the Kripke structure by (S, R, L, s^0); the state s^0 is usually named the *initial state.* Let $\pi \in A^{\omega}$; we denote by $\pi(k)$ the k-th element of π, the k-th suffix of π, by $\pi[k]$ (an element of A^{ω}) and the k-th prefix of π by $\pi|_k$ (an element of A^*). Throughout this work we will consider a countable set of propositional symbols Λ. They will usually be denoted by $p, q, p_1, p_2 \ldots$.

Definition 1.1.2. A *path* $\pi = s_0 s_1 \ldots s_n \ldots$ in a Kripke structure (S, R, L) is an element of S^{ω}, such that for all $i \in \mathbb{N}$, $(\pi(i), \pi(i+1)) \in R$. The *trace* of a path π, represented as $trace(\pi)$, where $\pi = s_0 s_1 \ldots s_n \ldots$ denotes the sequence of valuations associated with the path, that is $trace(\pi) = L(s_0)L(s_1)\ldots L(s_n)\ldots$; as such $trace(\pi) \in (\{0, 1\}^{\Lambda})^{\mathbb{N}}$.

This definition can be generalized for *finite paths*, by considering finite paths to be partial functions from $\mathbb{N} \mapsto S$ or elements of S^*, defined in an

initial segment of the natural numbers. We use the notation from language theory representing a path as an element of S^ω, or S^*, if the path is finite.

We are now ready to proceed with the description of the propositional-based temporal logics used in this work.

1.1.1 Linear time logics

In Linear Temporal Logic, time is seen as a discrete linear order without right endpoint. As it is based on classical propositional logic, we will be able to describe how propositions or events evolve along a structure isomorphic to the natural numbers with its order relation. In a Kripke structure, this is captured by stating that in order for a formula $\phi \in \mathsf{LTL}$ to be true in a Kripke structure $m = (S, R, L, s^0)$ it has to be true along all paths starting in s^0. In this way, we are requiring that all branches of the unwinding (for now take the word unwinding at its informal meaning) of the Kripke structure satisfy the formula ϕ.

We note that LTL as presented here describes the **future** evolution of propositional variables. In fact, LTL was originally formulated with connectives dealing with the past as well as the future [89]. For our work they are not required, and in fact they can be eliminated in most variations of linear time logics.

Definition 1.1.3 (Well-formed Formulae in LTL)**.** The set of *well-formed Linear Temporal formulae over the propositional symbols* Λ is denoted by $\mathsf{LTL}(\Lambda)$. This set is described as follows in Backus-Naur notation:

$$\phi := \top \quad | \quad p \quad | \quad (\phi \wedge \phi) \quad | \quad (\neg\phi) \quad | \quad (\mathsf{X}\phi) \quad | \quad (\phi \mathsf{U} \phi),$$

where $p \in \Lambda$.

The connectives $\wedge$ (conjunction) and $\neg$ (negation) are the same as in classical propositional logic. We included verum ($\top$) for convenience reasons. All the other classical connectives are defined in the usual way. Moreover, using the X (neXt time) and the U (Until) connective we define $(\mathsf{F}\phi) \equiv (\top \mathsf{U} \phi)$ (in the Future), and $(\mathsf{G}\phi) \equiv (\neg(\mathsf{F}(\neg\phi)))$ (Globally). The semantics of LTL, with respect to sequences of valuations, are as follows:

Definition 1.1.4 (Semantics)**.** Let $\sigma \in (\{0,1\}^\Lambda)^\omega$ and $\phi \in \mathsf{LTL}(\Lambda)$. Then the satisfaction relation between a sequence of valuations σ and a formula ϕ is defined inductively as follows:

- $\sigma \Vdash p$ iff $\sigma(0)(p) = 1$,

- $\sigma \Vdash \top$,
- $\sigma \Vdash (\phi_1 \wedge \phi_2)$ iff $\sigma \Vdash \phi_1$ and $\sigma \Vdash \phi_2$,
- $\sigma \Vdash (\neg\phi_1)$ iff $\sigma \nVdash \phi_1$,
- $\sigma \Vdash (\mathsf{X}\phi_1)$ iff $\sigma[1] \Vdash \phi_1$,
- $\sigma \Vdash (\phi_1 \mathsf{U} \phi_2)$ iff there exists $j \in \mathbb{N}$ s.t $\sigma[j] \Vdash \phi_2$ and for all $0 \leq i < j$, $\sigma[i] \Vdash \phi_1$.

A Kripke structure m satisfies ϕ, represented as $m \Vdash \phi$ iff for all paths π over m, $trace(\pi) \Vdash \phi$; in this case m is a model of ϕ. If we are considering Kripke structures with a designated initial state s, then we consider only the paths starting in s. We now list some relevant results concerning LTL. Recall that a Hilbert calculus is *sound* with respect to an entailment relation if $\Gamma \vdash \phi$ implies $\Gamma \models \phi$, and *complete* if $\Gamma \models \phi$ implies $\Gamma \vdash \phi$. If the results only hold in the absence of hypothesis, that is $\Gamma = \emptyset$, we instead refer to the properties by *weakly* sound/complete.

Theorem 1.1.1 ([95, 96, 46])**.** *There exists a Hilbert axiomatization for LTL which is sound and weakly complete for the given semantics. The satisfiability problem is PSPACE-complete.*

An important result concerning Linear Temporal Logic is that LTL with past connectives is expressively equivalent to the first-order logic over discrete linear orders without endpoints, with unary predicate symbols. This result, which was shown in [56], provides also a different way of eliminating past connectives.

Theorem 1.1.2. *LTL is expressively equivalent to the first-order logic over discrete linear orders with left endpoints (FO$[0, <]$).*

However, it was known since the sixties [21] that this first-order logic was weaker than the monadic second-order logic with an appropriate signature, for instance with a successor symbol (MSO(S)). In fact this weakness can be readily found; a simple natural language specification like ***p* must be true on even instants** cannot be expressed using LTL formulae [106]. From the point of view of Logic, these problems are solved using second-order logic, which since **Büchi** was known to be decidable; however it was not clear how one could enhance LTL to deal with these types of statements involving counting. Before presenting the extension of LTL capable of describing such properties, we will need to introduce the connection between LTL and automata over infinite words and therefore, Language Theory.

Definition 1.1.5. Let $\phi \in \mathsf{LTL}(\Lambda)$. The ω-*language* generated by ϕ, represented as $L^\omega(\phi)$ is defined to be $\{\sigma \in (\{0,1\}^\Lambda)^\omega : \sigma \Vdash \phi\}$.

Definition 1.1.6. A *finite state automaton* N is a tuple $N = (S, \Sigma, \delta, s^0, \alpha)$, where:

- S is a finite set, whose members are called *states*,
- Σ is a finite set, which is the *alphabet* and whose members are called *letters*,
- δ is a subset of $S \times \Sigma \times S$, which is the *transition relation*; if the transition relation δ is such that for all states $s \in S$ and letters $a \in \Sigma$, $\delta(s, a, t)$ holds for at most one state t, then the automaton is named *deterministic*.
- s^0 is a member of S, called the *initial state*,
- α is named the *accepting condition*.

A *run* on a finite state automaton N is a sequence $\rho \in \delta^\omega$ such that:

- $pr_1(\rho(0)) = s^0$, where $pr_i(.)$ represents the i-the projection;
- $\forall i \in \mathbb{N} \quad pr_3(\rho(i)) = pr_1(\rho(i+1))$.

The *trace* of a run $\rho = (s^0, a_0, s_1) \dots (s_i, a_i, s_{i+1}) \dots$ is given by $trace(\rho) = a_0 a_1 \dots a_i \in \Sigma^\omega$

The accepting conditions that we will use in this work are:

- Büchi's acceptance condition, denoted by $\mathcal{B}$, where $\alpha \subseteq S$; in this case α represents the *final states*;
- Rabin's acceptance condition, denoted by $\mathcal{R}$, where α is a finite set $\{(E_1, F_1) \dots (E_n, F_n)\}$, with $E_i, F_i \subseteq S$.

A run is called an *accepting run in the Büchi/Rabin accepting condition* if:

- Büchi acceptance condition: a run ρ is accepting in $\alpha = F$ if the set $Inf(\rho) = \{s \in S : s = pr_1(\rho(i))$ for infinitely many indexes $i \in \mathbb{N}\}$ is such that $Inf(\rho) \cap F \neq \emptyset$;
- Rabin acceptance condition: ρ is accepting in $\{(E_1, F_1) \dots (E_n, F_n)\}$, if there exists an index $i \in \{1, \dots n\}$ such that $Inf(\rho) \cap E_i = \emptyset$ and $Inf(\rho) \cap F_i \neq \emptyset$.

The Büchi/Rabin ω-language accepted by N is defined to be $L^{\omega}_{\mathcal{B}/\mathcal{R}}(N) = \{\sigma \in \Sigma^{\omega} : \text{ there exists } \rho \text{ s.t. } trace(\rho) = \sigma \text{ and } \rho \text{ is a } \mathcal{B}/\mathcal{R} \text{ accepting run}\}$.

A Büchi/Rabin automaton is a finite state automaton endowed with a Büchi/Rabin accepting condition. The connection between finite state automata over infinite words and LTL is obtained by the following proposition:

Proposition 1.1.1 ([96, 49]). *Let $\phi \in LTL(\Lambda)$. Then there exists a Büchi automaton $N = (S, \{0,1\}^{\Lambda}, \delta, s^0, F)$ such that $L^{\omega}(\phi) = L^{\omega}(N)$.*

Given that the proof of the above proposition is constructive, we can use the following proposition to show that the satisfiability problem for LTL is decidable.

Proposition 1.1.2 ([96]). *The problem of deciding whether a Büchi automaton accepts some sequence is decidable in NLOG.*

Proposition 1.1.3 (Closure of Büchi automata for complementation, [21, 96, 92]). *Let $N = (S, \{0,1\}^{\Lambda}, \delta, s^0, F)$. Then there exists a Büchi automaton $\overline{N}$ such that $L^{\omega}(\overline{N}) = \overline{L^{\omega}(N)}$ such that $size(\overline{N}) \in 2^{O(log(size(N))size(N))}$.*

Moreover, one can use the satisfiability procedure as a model-checking procedure. Suppose that we wish to verify if ϕ holds in a Kripke structure m. One constructs the Büchi automaton for ϕ, constructs its complement, and then consider the product between m, and the the Büchi automaton for $\neg\phi$. If there exists an accepting sequence in the automaton so obtained, then ϕ does not hold in every path of m.

Languages accepted by a non-deterministic Büchi automaton are named *ω-regular* languages. They are the extensions of regular languages over infinite words. They have many characterizations, but we summarize the most important ones:

Theorem 1.1.3 ([21, 33, 100]). *Let $L \subseteq \Sigma^{\omega}$. Then the following conditions are equivalent:*

- *there exists a closed formula ψ over MSO$[S]$ such that the set of interpretations that satisfy ψ is L,*
- *there exists a non-deterministic Büchi automaton N such that $L = L^{\omega}(N)$,*
- *there exists a non-deterministic Rabin automaton N such that $L = L^{\omega}(N)$,*

- *there exists a deterministic Rabin automaton N such that $L = L^{\omega}(N)$,*
- *$L = \bigcup_{i=1}^{n} A_i B_i^{\omega}$, with A_i, B_i some regular languages.*

One of the ideas to augment LTL in order to capture those naturally occurring alternating properties would be to directly introduce connectives representing finite automata; in fact **Wolper** [106] did as much; however there exists a much more natural way of enhancing LTL: the introduction of propositional quantification. As we will use frequently QLTL in our work, the syntax and semantics are as follows:

Definition 1.1.7 (Quantified Propositional Linear Temporal Logic)**.** The set of *well-formed Quantified Linear Temporal formulae over the propositional symbols* Λ is denoted by $\mathsf{QLTL}(\Lambda)$. This set is described as follows in Backus-Naur notation:

$$\phi := \top \quad | \quad p \quad | \quad (\phi \wedge \phi) \quad | \quad (\neg\phi) \quad | \quad (\mathsf{X}\phi) \quad | \quad (\phi \mathsf{U} \phi) \quad | \quad \exists p.\phi,$$

where $p \in \Lambda$.

As usual we define $\forall p.\phi$ as $\neg(\exists p.(\neg\phi))$. As usual in first-order logic, we say that the variable p is *bound* if it is within the scope an existential quantifier. The set of free propositional symbols of ϕ is denoted by $fpvar(\phi)$ and the set of all propositional symbols in ϕ is denoted by $pvar(\phi)$. In order to stress that ϕ only depends on the free propositional symbols, we may denote ϕ by $\phi(\mathbf{r})$; if $fpvar(\phi) = \emptyset$, then ϕ is a closed formula.

Definition 1.1.8 (Quantified Propositional Linear Temporal Logic Semantics)**.** Let $\sigma \in (\{0,1\}^{\Lambda})^{\omega}$ and $\phi \in \mathsf{LTL}(\Lambda)$. Then the satisfaction relation between a sequence of valuations σ and a formula ϕ is defined inductively as follows:

- $\sigma \Vdash p$ iff $\sigma(0)(p) = 1$,
- $\sigma \Vdash \top$,
- $\sigma \Vdash (\phi_1 \wedge \phi_2)$ iff $\sigma \Vdash \phi_1$ and $\sigma \Vdash \phi_2$,
- $\sigma \Vdash (\neg\phi_1)$ iff $\sigma \nVdash \phi_1$,
- $\sigma \Vdash (\mathsf{X}\phi_1)$ iff $\sigma[1] \Vdash \phi_1$,
- $\sigma \Vdash (\phi_1 \mathsf{U} \phi_2)$ iff there exists $j \in \mathbb{N}$ s.t $\sigma[j] \Vdash \phi_2$ and for all $0 \leq i < j$, $\sigma[i] \Vdash \phi_1$.

- $\sigma \Vdash \exists p.\phi$ iff there exists $\sigma' \in (\{0,1\}^{\Lambda})^{\omega}$ s.t σ' is p-equivalent to σ and $\sigma \Vdash \phi$; a sequence σ is p-equivalent to σ', denoted as $\sigma \sim_p \sigma'$, if $\sigma'(q) = \sigma(q)$ for all propositional symbols $q \in \Lambda$ different from p.

The consequence relations in QLTL are defined as in LTL.

The introduction of propositional quantification transforms LTL into a highly expressive logic.

Example 1.1.1. Consider the following formula: $\mathsf{G}_2 p \equiv \exists q.(q \wedge (\mathsf{X}\neg q) \wedge \mathsf{G}(q \Leftrightarrow \mathsf{XX}q) \wedge \mathsf{G}(q \Rightarrow p))$. We will show that $\pi \Vdash G_2 p$ iff p is true on all even instants, that is, $\pi(2k)(p) = 1$, $k \in \mathbb{N}$.

Suppose that $\pi \Vdash \exists q.(q \wedge (\mathsf{X}\neg q) \wedge \mathsf{G}(q \Leftrightarrow \mathsf{XX}q) \wedge \mathsf{G}(q \Rightarrow p))$; then there exists $\pi' \sim_q \pi$ s.t. $\pi' \Vdash q \wedge (\mathsf{X}\neg q) \wedge \mathsf{G}(q \Leftrightarrow \mathsf{XX}q) \wedge \mathsf{G}(q \Rightarrow p))$. Then:

- $\pi'(0)(q) = 1$, because $\pi' \Vdash q$;
- $\pi'(1)(q) = 0$, because $\pi' \Vdash (\mathsf{X}\neg q)$;
- $\pi'(k) = \pi'(k+2)$, because $\pi' \Vdash \mathsf{G}(q \Leftrightarrow \mathsf{XX}q)$;
- Combining the information, we get that $\pi'(2k)(q) = 1$ and $\pi'(2k+1)(q) = 0$;
- Finally, the last conjuct states that for all $n \in \mathbb{N}$, if $\pi'(n)(q) = 1$ then $\pi'(n)(p) = 1$, which given that $\pi'(2k)(q) = 1$, implies that $\pi'(2k)(p) = 1$ and so $\pi(2k) = 1$, since $\pi \sim_q \pi'$.

For the converse implication, suppose that $\pi(2k)(p) = 1$. Then, consider $\sigma : \mathbb{N} \mapsto \{0,1\}^{\{p,q\}}$ defined as follows:

$$\sigma(n)(p) = \pi(n)(p), \qquad \sigma(n)(q) = 1 \text{ iff } n \text{ is even.}$$

From the definition, we get that $\pi \sim_q \sigma$, and it is a simple matter to see that $\sigma \Vdash q \wedge (\mathsf{X}\neg q) \wedge \mathsf{G}(q \Leftrightarrow \mathsf{XX}q) \wedge \mathsf{G}(q \Rightarrow p))$.

Example 1.1.2. Another interesting feature of QLTL is that it is possible to define the connective U by abbreviation. Consider the following formula, where q is a propositional symbol appearing free in α and β:

$$Until(\alpha, \beta) \equiv \exists q.\mathsf{F}\beta \wedge q \wedge \mathsf{G}(q \Rightarrow (\beta \vee (\alpha \wedge \mathsf{X}q))).$$

Assume that $\pi \Vdash \alpha \mathsf{U} \beta$. Then there exists $n \in \mathbb{N}$ such that $\pi[n] \Vdash \beta$, and $\pi[j] \Vdash \alpha$ for all $0 \leq j < n$. Let N be the least index such that $\pi[n] \Vdash \beta$. Consider σ defined as follows:

$$\sigma(n)(p) = \pi(n)(p), p \neq q, \qquad \sigma(n)(q) = 1, 0 \leq n \leq N, \sigma(n)(q) = 0, n > N.$$

Then, $\sigma \Vdash \mathsf{F}\beta$, as q does not appear in β; moreover $\sigma \Vdash q$; then:

- for any $j > N$, we will have that $\sigma[j] \Vdash q \Rightarrow (\beta \vee (\alpha \wedge \mathsf{X}q)))$, as the premise will fail;
- for $0 \leq j < N$, $\sigma[j] \nVdash \beta$, but $\sigma[j] \Vdash \alpha$ and $\sigma[j] \Vdash \mathsf{X}q$;
- Finally for $j = N$, $\sigma[j] \Vdash \beta$ and so $\sigma[j] \Vdash q \Rightarrow (\beta \vee (\alpha \wedge \mathsf{X}q))$.

Therefore $\sigma \Vdash \mathsf{F}\beta \wedge q \wedge \mathsf{G}(q \Rightarrow (\beta \vee (\alpha \wedge \mathsf{X}q)))$ and as such $\pi \Vdash \exists q.\mathsf{F}\beta \wedge q \wedge \mathsf{G}(q \Rightarrow (\beta \vee (\alpha \wedge \mathsf{X}q)))$, as $\sigma \sim_q \pi$.

For the converse implication, suppose that $\pi \Vdash \exists q.\mathsf{F}\beta \wedge q \wedge \mathsf{G}(q \Rightarrow (\beta \vee (\alpha \wedge \mathsf{X}q)))$; then there exists $\pi' \sim_q \pi$ s.t. $\pi' \Vdash \mathsf{F}\beta \wedge q \wedge \mathsf{G}(q \Rightarrow (\beta \vee (\alpha \wedge \mathsf{X}q)))$. Now, either for all $i \in \mathbb{N}$, $\pi'(i)(q) = 1$ (1), or there exists an index N such that $\pi'(j)(q) = 1$, $0 \leq j \leq N$ and $\pi'(N+1)(q) = 0$ (2).

If (1) holds, then until $\pi'[j] \Vdash \beta$, (which will happen since $\pi' \Vdash \mathsf{F}\beta$), π' must satisfy α, which implies that $\pi \Vdash \alpha \mathsf{U} \beta$. If (2) holds, in the instant N, the second disjunct in the invariant $(\mathsf{G}(q \Rightarrow (\beta \vee (\alpha \wedge \underline{\mathsf{X}q}))))$ will fail, and therefore β must hold. Until then, π' must satisfy α at every instant, therefore $\pi \Vdash \alpha \mathsf{U} \beta$.

Proposition 1.1.4 (QLTL normal forms, [96]). *Let $\phi \in \mathsf{QLTL}(\Lambda)$. Then there exists $\psi_1, \psi_2 \in \mathsf{LTL}(\Lambda)$ such that:*

$$\models \phi \Leftrightarrow Q_1 p_1 \ldots Q_n p_m \psi_1, \quad \textit{(prenex normal form)}$$
$$\models \phi \Leftrightarrow \exists q_1 \ldots \exists q_m . \psi_1, \quad \textit{(existential prenex normal form)},$$

where $Q_i \in \{\exists, \forall\}$.

Theorem 1.1.4 (Equivalency between Büchi Automata and QLTL formulae, [96]). *Let N be a Büchi automaton over the alphabet $\{0,1\}^\Lambda$; then there exists a QLTLformula ϕ such that $L^\omega(\phi) = L^\omega(N)$. Furthermore, for any formula $\phi \in \mathsf{QLTL}(\Lambda)$, there exists a Büchi automaton N over the alphabet $\{0,1\}^\Lambda$ such that $L^\omega(\phi) = L^\omega(N)$.*

Theorem 1.1.5 (Complexity of the satisfiability problems for QLTL, [96]). *The satisfiability problem for a formula in prenex normal form ϕ, where the innermost and the outermost quantifiers are existential is [n]-EXPSPACE-Complete, where n is the number of alternations. If $n = 0$, then ϕ is in existential normal form and the complexity is PSPACE-Complete.*

If one needs to produce a witness for the satisfiability of ϕ, the complexity jumps one level.

1.1.2 Branching Time Tree Logics

The fundamental theory of branching time temporal logics is very rich, and in some aspects, it is still in active development. The core idea is that the future instant of a computation is not completely determined. This nondeterminism can have different roots: an interaction with the environment or concurrent processes running in parallel, for instance. While LTL is able to analyze non deterministic models, by considering LTL semantics for Kripke structures, a LTL formula always must be interpreted as a set of infinite traces, therefore any LTL verification of a Kripke structure must only focus on trace properties.

The Computation Tree Logic CTL, heavily developed by **Clarke** [34], appeared as the first logic for analyzing the branching properties of the computation tree. It is a highly used logic in the Model Checking area, and in practical verification tools. We will list some important results about CTL, and about its extension, CTL* [35]. This extension is particularly useful as it allows us to relate CTL and LTL.

Definition 1.1.9. Let Λ be a countable set of propositional symbols. The set of CTL *well-formed formulae* over the propositional symbols Λ is defined as:

$$\phi := \top \quad | \quad p \quad | \quad (\phi \wedge \phi) \quad | \quad (\neg\phi) \quad | \quad (\mathsf{EX}\phi) \quad | \quad (\mathsf{E}(\phi \mathsf{U} \phi)) \quad | \quad (\mathsf{EG}\phi),$$

where $p \in \Lambda$.

We define the usual abbreviations for the propositional connectives. Moreover, the three temporal connectives $\mathsf{EX}\phi$ (there Exist a path such that in the neXt state, ϕ holds), $\mathsf{E}(\phi_1 \mathsf{U} \phi_2)$ (there Exists a path such that ϕ_1 holds in every state Until ϕ_2 holds), and $\mathsf{EG}\phi$ (there Exists a path such that ϕ holds Globally), allows us to define $\mathsf{AX}\phi \equiv \neg\mathsf{EX}\neg\phi$, $\mathsf{EF}\phi \equiv \mathsf{E}(\top \mathsf{U} \phi)$, $\mathsf{AG}\phi \equiv \neg\mathsf{EF}\neg\phi$, $\mathsf{AF}\phi \equiv \neg\mathsf{EG}\neg\phi$ and $\mathsf{A}(\phi_1 \mathsf{U} \phi_2) \equiv \neg(\mathsf{E}(\neg\phi_2 \mathsf{U} \neg(\phi_1 \vee \phi_2)) \vee \mathsf{EG}\neg\phi_2)$.

The semantics for Kripke structures is given below:

Definition 1.1.10. Let m be a Kripke structure over the propositional symbols Λ, s a state in m, and ϕ a CTL formula. The satisfaction relation between a Kripke structure $m = (S, R, L)$, a state s and the formula ϕ is defined inductively below:

- $m, s \Vdash \top$,
- $m, s \Vdash p$ iff $L(s)(p) = 1$,

- $m, s \Vdash \neg\phi$ iff $m, s \nVdash \phi$,
- $m, s \Vdash \phi_1 \wedge \phi_2$ iff $m, s \Vdash \phi_1$ and $m, s \Vdash \phi_2$,
- $m, s \Vdash \mathsf{EX}\phi$ iff there exists a path $\pi = s_0 s_1 \ldots$ over m such that $\pi(0) = s$, and $m, \pi(1) \Vdash \phi$,
- $m, s \Vdash \mathsf{EG}\phi$ iff there exists a path $\pi = s_0 s_1 \ldots$ over m such that $\pi(0) = s$, and $m, \pi(i) \Vdash \phi$, for all $i \geq 0$,
- $m, s \Vdash \mathsf{E}(\phi_1 \mathsf{U} \phi_2)$ iff there exists a path $\pi = s_0 s_1 \ldots$ over m such that $\pi(0) = s$, there exists j such that $m, \pi(j) \Vdash \phi_2$, and $m, \pi(k) \Vdash \phi_1$, for all $0 \leq k < j$.

The consequence relation $\models$ for Kripke structure semantics usually is defined by $\Gamma \models \phi$ iff any Kripke structure m satisfying all the elements of Γ also satisfies ϕ.

There is an alternative semantics, which is equivalent in the case of CTL and CTL^*; given a Kripke structure, $m = (S, R, L)$, we construct the unwinding of m, which is also a (infinite) Kripke structure; the semantics for CTL and CTL^* are then defined over the generated computation tree, as in Definition 1.1.10.

Definition 1.1.11 (Unwinding of a Kripke structure)**.** Let $m = (S, R, L)$ be a Kripke structure over the propositional symbols Λ. Then, the *unwinding of* m based on the state s, represented as $T_s(m)$ is the following Kripke structure (S_T, R_T, L):

- $S_T \subseteq S^*$ is such that $s \in S_T$ and if $x.s_i \in S_T$, then $x.s_i.y \in S^*$ for all $y \in S$ such that $R(s_i, y)$ holds;
- $R_T \subseteq S_T \times S_T$ is such that $R_T(x, x.t)$ iff $x \in S_T$ and $x.t \in S_T$;
- $L_T : S_T \mapsto \{0, 1\}^\Lambda$ is such that $L_T(x.t) = L(t)$.

We now note some important differences between LTL and CTL, when interpreted over Kripke structures.

Example 1.1.3. Consider the following Kripke structure:

It is clear that $m, s_1 \Vdash \mathsf{AGEF}p$. However, the LTL candidates $\mathsf{GF}p$ or even $\mathsf{FG}p$ are not satisfied by m. In fact, it is possible to prove that there does not exist any LTL formula capable of expressing $\mathsf{AGEF}p$. In fact, LTL has

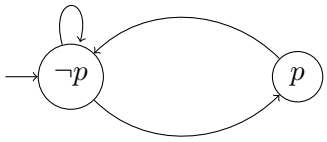


Figure 1.1: A Kripke structure m such that satisfying $\mathsf{AGEF}p$.

no capabilities for expressing the existence of a path verifying some interesting condition in a Kripke structure, as its Kripke semantics enforce that ϕ holds in a structure, if it holds along **all** paths. However, now consider the following Kripke structure:

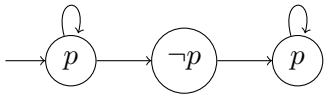


Figure 1.2: A Kripke structure m such that satisfying the LTL formula $\mathsf{FG}p$ but not satisfying the CTL formula $\mathsf{AFAG}p$.

While in fact these examples do not prove that LTL and CTL are incomparable in expressiveness, it is in fact possible to show that these intuitions are correct. However, it is still an open area of research understanding precisely when does a CTL formula has an equivalent version in LTL and vice-versa [17].

The logic CTL is axiomatizable, decidable, and its model-checking problem is P-complete, which coupled with the fact that it expresses many elementary properties easily (namely those about existence of paths) made CTL a staple in verification of hardware in the nineties. We summarize the important results related to CTL in the following theorem:

Theorem 1.1.6 ([34, 35, 7])**.** *There exists a Hilbert-style axiomatization for CTL, which is sound and weakly complete. The satisfiability problem for CTL is $\mathsf{EXPTIME}$-complete, and given a Kripke structure m and a CTL formula ϕ, testing whether $m \Vdash \phi$ is P-complete.*

There exists an extension of both CTL and LTL that solves these expressiveness issues. The logic CTL^* allows arbitrary conjugation of temporal connectives ($\mathsf{X}, \mathsf{F}, \mathsf{G}, \mathsf{U}$), without the restriction that a temporal connective must be followed by a path connective A, E. However, a temporal connective

eventually must be in the scope of a path connective. The syntax for CTL^* is given as follows:

Definition 1.1.12. Let Λ be a countable set of propositional symbols. The set of CTL^* *well-formed state formulae* over the propositional symbols Λ is defined in Backus Naur form by:

$$\begin{aligned} \phi_p &:= \phi_{st} \quad | \quad (\phi_p \wedge \phi_p) \quad | \quad (\neg\phi_p) \quad | \quad (\mathsf{X}\phi_p) \quad | \quad (\phi_p \mathsf{U} \phi_p), \\ \phi_{st} &:= p \quad | \quad (\phi_{st} \wedge \phi_{st}) \quad | \quad (\neg\phi_{st}) \quad | \quad (\mathsf{A}\phi_p) \quad | \quad (\mathsf{E}\phi_p), \end{aligned}$$

where $p \in \Lambda$.

The semantics for CTL^* can be given directly for Kripke structures or for their unwinding, just like in CTL. We remind the reader that CTL^* only comprises state formulae. We require two types of satisfaction relations, one for state formulae, and another for path formulae.

Definition 1.1.13. Let Λ be a countable set of propositional symbols, m a Kripke structure over Λ, and ϕ a CTL^* formula (state formula). The *satisfaction relation* between the Kripke structure m, a state s and ϕ, represented as $m, s \Vdash \phi$, is defined inductively, below:

- $m, s \Vdash p$ iff $L(s)(p) = 1$,
- $m, s \Vdash \phi_1 \wedge \phi_2$ iff $m, s \Vdash \phi_1$ and $m, s \Vdash \phi_2$,
- $m, s \Vdash \neg\phi_1$ iff $m, s \nVdash \phi_1$,
- $m, s \Vdash \mathsf{A}\phi_1$ iff $m, \pi \Vdash \phi_1$, for all paths over m such that $\pi(0) = s$, where ϕ_1 is a path formula,
- $m, s \Vdash \mathsf{E}\phi_1$ iff there exists a path over m such that $\pi(0) = s$ and $m, \pi \Vdash \phi_1$, where ϕ_1 is a path formula.

Moreover, the satisfaction relation between the Kripke structure m, a path π over m and ϕ, a path formula, represented as $m, \pi \Vdash \phi$, is also defined inductively:

- $m, \pi \Vdash \phi$ iff $m, \pi(0) \Vdash \phi$, where ϕ is a state formula,
- $m, \pi \Vdash \phi_1 \wedge \phi_2$ iff $m, \pi \Vdash \phi_1$ and $m, \pi \Vdash \phi_2$,
- $m, \pi \Vdash \neg\phi_1$ iff $m, \pi \nVdash \phi_1$,
- $m, \pi \Vdash \mathsf{X}\phi_1$ iff $m, \pi[1] \Vdash \phi_1$,

- $m, \pi \Vdash \phi_1 \mathsf{U} \phi_2$ iff there exists $j \in \mathbb{N}$ s.t $m, \pi[j] \Vdash \phi_2$ and for all $0 \leq i < j$, $m, \pi[i] \Vdash \phi_1$.

The consequence relation $\models$ for Kripke structure semantics usually is defined by $\Gamma \models \phi$ iff any Kripke structure m satisfying all the elements of Γ also satisfies ϕ.

With these semantics, we can now freely combine formulae from CTL and LTL; naturally, a formula like $\mathsf{A}(\mathsf{FG}p) \wedge \mathsf{AGEF}q$ cannot be expressed neither in CTL nor LTL. The practical usefulness of CTL*might be argued for, as it allows mixing of both linear properties and branching properties, however, there are steep complexity costs associated to the main algorithms.

Theorem 1.1.7 ([35, 90])**.** *The satisfiability problem for CTL* is 2-EXPTIME-complete, and given a Kripke structure m and a CTL formula ϕ, testing whether $m \Vdash \phi$ is PSPACE-complete. There exists a sound and complete axiomatization for CTL*.*

There exists one further well known formalism that can encapsulate these branching time logics. This extension appears naturally when one notices that many of the temporal connectives can be written as fixed points, like for instance $\phi \mathsf{U} \psi \Leftrightarrow \psi \vee (\phi \wedge \mathsf{X}(\phi \mathsf{U} \psi))$. However, it is not an extension in the strictest sense, as it is in fact an extension of propositional dynamic logic. However, with some natural adaptations of Kripke structures, it is in fact possible to show that the μ-Calculus (μ-TL [65, 67]) is in fact capable of expressing any property expressible in CTL*.

Moreover, there is a relevant property with extremely practical applications that all these branching logics share. A bisimulation relation between two Kripke structures m_1, m_2, modeling two systems, intuitively asserts that the behaviors of m_1 are indistinguishable from the behaviors of m_2. A simulation relation is only one sided, that is, if m_2 simulates m_1, then any behavior in m_1 can be mimicked in m_2. These relations form the cornerstone of a technique widely used in Model Checking named abstraction.

For instance, if some concrete protocol is modeled by a Kripke structure m_1, which is very large, one might hope that there exists a bisimulation between m_1 and a smaller, more manageable Kripke structure m_2. Since the structures are bisimilar, the observable behaviors are the same. Hopefully, if our logic is bisimulation-invariant, any formula that holds in m_2, automatically holds in m_1. The branching logics presented so far are in fact bisimulation-invariant.

While the presented in this document does not make use of bisimulation relations, as it is focused on developing interesting and useful probabilistic

temporal logics, it is important to be aware of the properties that these fundamental logics are endowed with. In fact, μ-TL captures **precisely** all the branching time properties that are bisimulation-invariant!

We define the syntax and semantics of μ-TL with respect to a Kripke structure endowed with many transition relations; we then list some of the more important results concerning μ-TL, with their respective references.

Definition 1.1.14. A *multimodal Kripke structure* m over Λ and transition relations over T is a tuple $m = (S, \{R_t\}_{t \in T}, L)$, such that:

- S is a set, whose members are usually called *states*; we usually assume that S is countable;
- R_t is a total relation between states for each $t \in T$, that is $R_t \subseteq S \times S$ such that $\forall x \exists y R_t(x, y)$, usually called the *transition relation*; we usually assume that T is finite.
- L is a map between S and $\{0, 1\}^\Lambda$, associating each state to a valuation of propositional symbols, usually called the *labeling function*.

We note that we might wish to have one designated initial state, as in the case of simple Kripke stuctures.

Definition 1.1.15 (μ-TL syntax, from Clarke [28])**.** Let Λ be the set of propositional symbols, and $X = \{Q_1, \ldots, Q_n, \ldots\}$ the set of *relational variables*, and T a set of transition relations. Then, the set of μ-TL *well-formed formulae* is defined inductively as follows:

- A propositional symbol p is a well-formed formula;
- A relational variable symbol Q_1 is a well-formed formula;
- Given ϕ and ψ, well formed formulae, all $\neg\phi$, $\phi \wedge \psi$ and $\phi \vee \psi$ are well formed formulae;
- If $t \in T$, and ϕ is a well formed formula then $[t]\phi$ and $\langle t \rangle \phi$ are well formed formulae;
- If $Q \in X$, and ϕ is a well formed formula and all occurrences of Q not under the scope of μ are under an even number of negations, then $\mu Q.\phi$ and $\nu Q.\phi$ are well formed formulae.

If all occurrences of relational variables in a μ-TL formula ϕ are under the scope of some fixed point operator, then ϕ is named a *sentence*. If a formula ϕ has as free variables $Q_1, \ldots, Q_n$, we represent it as $\phi(Q_1, \ldots, Q_n)$.

The modalities $[t]\phi$ and $\langle t\rangle\phi$ are both used for one step reasoning; $[t]\phi$ asserts that ϕ holds in a state for every t transition and $\langle t\rangle\phi$ asserts that it is possible to make a t transition towards a state where ϕ holds. The operators μ and ν will be interpreted as least and greatest fixed points, allowing a formula to assert properties about the distant future. An *assignment* $\rho : X \mapsto \mathcal{P}(S)$ interprets the relational variables; we say that two assignments ρ, ρ' are Q_i-equivalent, represented as $\rho \sim_{Q_i} \rho'$, if $\rho(Q_j) = \rho'(Q_j)$ for all $j \neq i$, that is if the assignment agrees in every variable but Q_i.

Definition 1.1.16. Let $m = (S, \{R_t\}_t \in T, L)$ be a multimodal Kripke structure over Λ with transitions relations over T. The denotation of a formula $\phi \in \mu\text{-}\mathsf{TL}$ in m, given an assignment $\rho : X \mapsto \mathcal{P}(S)$, represented as $[\![\phi]\!]_{m,\rho}$ is a map towards $\mathcal{P}(S)$ given recursively by:

- $[\![p]\!]_{m,\rho} = \{s \in S : L(s)(p) = 1\}$,
- $[\![Q]\!]_{m,\rho} = \rho(Q)$,
- $[\![\neg\phi_1]\!]_{m,\rho} = S[\![\phi]\!]_{m,\rho}$,
- $[\![\phi_1 \wedge \phi_2]\!]_{m,\rho} = [\![\phi_1]\!]_{m,\rho} \cap [\![\phi_1]\!]_{m,\rho}$,
- $[\![\phi_1 \vee \phi_2]\!]_{m,\rho} = [\![\phi_1]\!]_{m,\rho} \cup [\![\phi_1]\!]_{m,\rho}$,
- $[\![[t]\phi]\!]_{m,\rho} = \{s \in S : \forall u \in S \text{ if } (s,u) \in R_t \text{ then } u \in [\![\phi]\!]_{m,\rho}\}$,
- $[\![\langle t\rangle\phi]\!]_{m,\rho} = \{s \in S : \exists u \in S \text{ s.t. } (s,u) \in R_t \text{ and } u \in [\![\phi]\!]_{m,\rho}\}$,
- $[\![\mu Q.\phi]\!]_{m,\rho} = \bigcap\{S' \subseteq S : [\![\phi]\!]_{m,\rho'} \subseteq S', \text{ where } \rho' \sim_Q \rho \text{ s.t. } \rho'(Q) = S'\}$,
- $[\![\nu Q.\phi]\!]_{m,\rho} = \bigcup\{S' \subseteq S : S' \subseteq [\![\phi]\!]_{m,\rho'}, \text{ where } \rho' \sim_Q \rho \text{ s.t. } \rho'(Q) = S'\}$.

If $[\![\phi]\!]_{m,\rho} = S$, then we say that the multimodal Kripke structure m and assignment ρ *satisfies* ϕ, represented as $m, \rho \Vdash \phi$. In the case of multimodal Kripke structures with designated initial state, we just require that the initial state belongs to $[\![\phi]\!]_{m,\rho}$.

If for any multimodal Kripke structure $m = (S, \{R_t\}_T, L)$ and assignment ρ, $[\![\phi]\!]_{m,\rho} = S$, then ϕ is a *valid* formula. Moreover, the consequence relation is defined in the usual way: $\Gamma \models \phi$ holds if for any multimodal Kripke structure m and assignment ρ that satisfies every $\gamma \in \Gamma$, $m, \rho \Vdash \phi$.

The requirement of an even number of negations is a necessary and sufficient condition in order for the fixed points to exist. Using DeMorgan's laws, and dualities of both modalities ($\neg[t]\phi \equiv \langle t\rangle\neg\phi$) and fixed points operators ($\neg\mu Q.\phi(Q) \equiv \nu Q.\neg\phi(\neg Q)$), one can eliminate negations from bound variables, with the remaining negations only applied to propositional symbols. Therefore if $\rho \sim_Q \rho'$ and $\rho(Q) \subseteq \rho'(Q)$ then $[\![\phi]\!]_{m,\rho} \subseteq [\![\phi]\!]_{m,\rho'}$, which can be proven by structural induction, provided that there only negations applied to propositional symbols; this fact is enough to show that the (least/greatest) fixed points exist, by the Knaster-Tarski fixed point theorem [32].

Theorem 1.1.8. *Let ϕ be a CTL* formula over the propositional symbols Λ; then there exists ψ, a sentence in μ-TL, over Λ and a single transition $\{a\}$ such that $m \Vdash \phi$ iff $m \Vdash \psi$.*

Theorem 1.1.9 ([67, 28])**.** *The satisfiability problem for μ-TL is EXPTIME-complete, and given a Kripke structure m and a CTL formula ϕ, testing whether $m \Vdash \phi$ is in NP $\cap$ co-NP. Furthermore, there exists a sound and complete axiomatization for μ-TL, which is presented in Table 1.1.*

Axioms

[**Ax1**]	all PL tautologies
[**Ax2**]	$([a]\varphi_1) \wedge ([a]\varphi_2) \Leftrightarrow ([a](\varphi_1 \wedge \varphi_2))$
[**Ax3**]	$((((\langle a\rangle\varphi_1) \wedge ([a]\varphi_2)) \Rightarrow (\langle a\rangle(\varphi_1 \wedge \varphi_2)))$
[**Ax4**]	$((\langle a\rangle\bot) \Leftrightarrow \bot)$
[**Ax5**]	$((\lceil\varphi\rceil^{\xi}_{\mu\xi.\varphi}) \Rightarrow (\mu\xi.\varphi))$

Rules of Inference

[**MP**]	$\varphi_1, (\varphi_1 \Rightarrow \varphi_2) \vdash^{\mu\text{-TL}} \varphi_2$
[**B**]	if $\vdash^{\mu\text{-TL}} \varphi$ then $\vdash^{\mu\text{-TL}} [a]\varphi$
[$\boldsymbol{\mu}$]	$(\lceil\varphi_1\rceil^{\xi}_{\varphi_2}] \Rightarrow \varphi_2) \vdash^{\mu\text{-TL}} (\mu\xi.(\varphi_1 \Rightarrow \varphi_2))$

Table 1.1: A Hilbert calculus for μ-TL where we use the notation $\lceil\varphi\rceil^{\xi}_{\psi}]$ to denote the formula that can be obtained by replacing the propositional variable ξ by the formula ψ.

However, all these formalisms cannot capture completely the full power of Monadic Second-order Logic with two (or more) successor functions [44, 72], even though on standard structures μ-TL already matches its expressiveness. In fact, only by adding some type of operator capable of counting, like in QLTL versus LTL, one can in fact reach the expressiveness of monadic second-order logic [72, 79]. Although we will not consider propositional quantification over branching time models in our work, for completeness sake we chose to mention some important references concerning CTL and CTL* endowed with propositional quantification.

Surprisingly enough, quantified CTL and CTL* are equally expressive, with decidable model-checking algorithms, in both Kripke semantics and the computation tree semantics. Moreover, over the tree semantics, quantified CTL is still decidable, although with steep complexity costs. However, all these logics share a common property with QLTL: any formula can be rewritten into existential normal form, whose problems have much more amenable complexity.

Chapter 2

Probabilistic Logic

Probability and Logic are subjects that do not seem at first sight to mesh well. After all, Logic usually deals with absolute certainties and deductions, which are formal in nature, whereas Probability is inherently numeric, and built for reasoning with uncertainty. This problem is exacerbated if one considers the multiple interpretations of Probability. However, we note that the most common formulation of Probability is based on Kolgomorov's Axioms, and many efforts were made in the 20^{th} century to formalize different aspects of Probabilistic reasoning.

More recently, there has been increasing interest in developing logics with a probabilistic component. Instead of dealing with Probability at a fundamental level, a probabilistic component is introduced in classical logics to enhance the applicability of pure logical reasoning to the context of real world applications. A practical scenario, even if fully deterministic in its nature (assuming the existence of fully deterministic real world scenarios), has to be brought onto a suitable level of abstraction which is dependent on the features one wishes to analyze. In any case, this abstraction process has to introduce non-determinism, even if the system is perfectly deterministic, in order to reduce the complexity present in the real world scenario. Consider the following (un)real scenario: there are two processes running on a processor. The processor assigns execution time to each process in a completely deterministic fashion, although fairly complicated. If we want to study the interleaving of two processes, we can abstract the system by considering all the possible interleavings of the processes, ignoring the deterministic algorithm run by the processor. However, that generates a non-deterministic model.

Nondeterminism is essential for abstraction. However, it comes at a great

cost. The behavior of the real world system will always be a subset of the behavior of the abstract model, because the latter is meant to ignore some of the inner workings of the former. This implies that a property may not hold in the abstract model, but in fact the real world system does verify it. For instance, consider the example given above. In the abstract model it will be possible that even though the second process can start running, because the shared memory is unlocked, the processor never chooses to assign execution time to it; this type of issues led to CTL Model Checking using fairness constraints, that is, the behaviors of the abstract model are restricted to a *fair* component, that approximates the behaviors of the abstract model to the real world system.

The need for a probabilistic component is then justified; if nondeterminism is useful as it allows better abstractions, one may need to control the amount of choice, in order to restrict extremely adversarial behavior. This can be done in some cases by using probabilistic reasoning, allowing the characterization of likely/unlikely behaviors; if a model is not secure because there exists a counterexample, one always wonders whether the counterexample is systemic, and likely to occur on the real world system, or it is in fact irrelevant; the probabilistic component quantifies the importance of these counterexamples. Moreover, probabilistic logics are also necessary to deal with systems that are inherently probabilistic, like randomized algorithms or protocols.

In this work, our interest in probabilistic logics is a practical one. We need to enrich temporal logics with the ability to perform quantitative reasoning, so that we can analyze systems that are inherently probabilistic, or systems that can be seen as probabilistic in nature. There are many probabilistic logics that are able to do so, and unlike temporal logic, there is not a systematic way to approach their expressiveness. There is however an abstract framework that allows for the probabilization of a logic system, that describes given a logic what can be its probabilistic version. This abstract framework emerged from earlier work [78], and in fact it is extremely relevant for the results in Chapters 4 and 5. It is also closely related to the fundamental work by **Fagin et al** [38] that proposed a powerful logic for reasoning with probabilities. A weaker version of this logic is also extremely well-documented and used for its practical applications by the Model Checking community [48].

In Section 2.1.3, we shall present an abridged version of the definitions concerning the Probabilization operator by **Baltazar** [8], which helps to introduce the setting for the main probabilistic logic EPPL, described in Section 2.2 where some new results regarding EPPL are presented.

2.1 Preliminaries

2.1.1 Probability Spaces

In this subsection we introduce the necessary definitions about probability spaces. A suitable reference for this subsection is the book by **Ash et al.** [2].

Definition 2.1.1. A *semi-ring* S over X is a subset of $\mathcal{P}(X)$ such that:

- $\emptyset \in S$,
- S is closed over finite intersections,
- for all $A, B \in S$ there exists $\{C_1, \dots C_n\}$ with $C_i \in S$ such that $A - B = \bigcup_{i=1}^{n} C_i$.

Definition 2.1.2. A *ring* R over X is a subset of $\mathcal{P}(X)$ such that:

- $\emptyset \in R$,
- R is closed over finite unions,
- R is closed under relative complements.

Every semi-ring induces a canonical ring of sets:

Proposition 2.1.1. *Let S be a semi-ring over the set X. Then $R(S) = \{A \subseteq X : A = \cup_{i=1}^{n} A_i, A_i \in S\}$ is a ring over X.*

Definition 2.1.3 (σ-Algebra)**.** A *σ-algebra $\mathcal{F}$ over a set X* is such that $\mathcal{F} \subseteq \mathcal{P}(X)$ such that $X \in \mathcal{F}$ and $\mathcal{F}$ is closed under complement and countable unions. Elements of $\mathcal{F}$ are named *measureable sets* or *events* in the case of probability spaces.

Every ring induces a canonical σ-algebra, and the σ-algebra induced by a semi-ring S is the same as the σ-algebra induced by $R(S)$:

Proposition 2.1.2. *Let S be a semi ring over the set X. Then $\sigma(S)$ is the intersection of all σ-algebras containing all elements of S.*

Definition 2.1.4 (Probability Space)**.** A *probability space* $P = (\Omega, \mathcal{F}, \mu)$ is a tuple such that:

- Ω is a set, usually named *sample space*;
- $\mathcal{F}$ is a σ-algebra over Ω, named the *event space*;

- $\mu : \mathcal{F} \mapsto [0;1]$ is a *probability measure*, that is $\mu(\Omega) = 1$, which also satisfies $\mu(\bigcup_{i\in I} E_i) = \sum_i \mu(E_i)$ for any countable collection of disjoint elements of $\mathcal{F}$.

Definition 2.1.5 (Pre-measure)**.** Let S be a semi-ring over X. A *pre-measure* $\mu_0 : S \mapsto [0;1]$ is such that $\mu_0(\emptyset) = 0$, $\mu_0(X) = 1$ and for every countable sequence $\{A_i\}_{i\in\mathbb{N}}$ of pairwise disjoint sets such that $\bigcup_{i\in\mathbb{N}} A_i \in S$, $\mu_0(\bigcup_{i\in\mathbb{N}} A_i) = \sum_{i\in\mathbb{N}}(\mu_0(A_i))$.

All these definitions come together in an extremely useful theorem that states that any pre-measure over a semi-ring S can be uniquely extended to a probability measure over the σ-algebra generated by the semi-ring. This theorem allows us to define a probability measure by assigning probability to just a semi-ring, instead of the whole σ-algebra.

Theorem 2.1.1 (Carathéodory extension theorem)**.** *Let S be a semi-ring over X and let $\mu_0 : S \mapsto [0;1]$ be a pre-measure. Then there exists a unique probability measure $\mu : \sigma(S) \mapsto [0;1]$ such that $\mu|_S = \mu_0$.*

2.1.2 Ordered Real Closed Fields

Reasoning about algebraic operations with probabilities can be done in the framework of real closed fields. A real closed field is a field which shares the same algebraic properties with the real numbers.

Definition 2.1.6. Let $\mathbb{K}$ be a field. Then an *ordered* field $O = (\mathbb{K}, \leq)$ is such that:

- $\leq$ is a total order on $\mathbb{K}$,
- $\forall x, y \quad x \leq y \Rightarrow x + z \leq y + z$,
- $\forall x, y \quad 0 \leq x \wedge 0 \leq y \Rightarrow 0 \leq x.y$,

A *real closed* field $(\mathbb{K}, \leq)$ is an ordered field such that $\{x \in \mathbb{K} : 0 \leq x\} = \{x.x : x \in \mathbb{K}\}$ and any odd degree polynomial has a root in $\mathbb{K}$.

Let Σ_{ORCF} be the following first-order signature: $\{0, 1, -, +, ., =, <\}$. As usual, we will not represent the multiplication operation ., and abbreviate $\underbrace{y.\ldots.y}_{n}$ by y^n.

[**AxF1**]	$\forall x \forall y \forall z\ x + (y + z) = (x + y) + z$	(associativity of $+$)
[**AxF2**]	$\forall x \forall y\ x + y = y + x$	(commutativity of $+$)
[**AxF3**]	$\forall x\ x + 0 = x$	(existence of neutral element for $+$)
[**AxF4**]	$\forall x\ x + (-x) = 0$	(existence of symmetric element)
[**AxF5**]	$\forall x \forall y \forall z\ x.(y.z) = (x.y).z$	(associativity of $.$)
[**AxF6**]	$\forall x \forall y\ x.y = y.x$	(commutativity of $.$)
[**AxF7**]	$\forall x\ x.1 = x$	(existence of neutral element for $.$)
[**AxF8**]	$\forall x\ x.0 = 0$	(existence of absorbing element $.$)
[**AxF9**]	$\forall x\ ((\neg x = 0) \Rightarrow (\exists y\ x.y = 1))$	(existence of inverse)
[**AxF10**]	$\forall x \forall y \forall z\ x.(y + z) = x.y + x.z$	(distributivity)
[**AxF11**]	$\neg(0 = 1)$	(fields with at least two elements)
[**Ax=**]	All the equality axioms	
[**AxRCF1**]	$\forall x \exists y\ (x = y^2) \vee (-x = y^2)$	
[**AxRCF2**]	$\forall x_0 \ldots \forall x_{n-1} \exists y\ y^n + x_{n-1}y^{n-1} + \ldots + x_1 y + x_0 = 0$	(for each odd natural number n)
[**AxORCF3**]	$\forall x \forall y\ (x < y) \Leftrightarrow (\exists z\ x + z^2 = y)$	

Table 2.1: The first-order theory of Ordered Real Closed Fields.

Definition 2.1.7. The the axioms in Table 2.1 axiomatize the first-order theory of ordered real closed fields, denoted by $\Theta^{\vdash}_{\mathsf{ORCF}}$ (Θ_{ORCF} denotes the axiomatization).

We note that Axiom **AXRCF2** is in fact an Axiom schema.

Proposition 2.1.3 ([10])**.** *The axiomatization Θ_{ORCF} axiomatizes precisely the real closed fields.*

Theorem 2.1.2 ([10, 99])**.** *The theory $\Theta^{\vdash}_{ORCF}$ admits quantifier elimination, that is, for any first-order formula φ in the signature Σ_{ORCF} such that $fv(\varphi) = \{x_1, \ldots, x_n\}$, there exists a first-order formula ψ without quantifiers such that $fv(\psi) = fv(\varphi)$ such that:*

$$\Theta_{ORCF} \vdash \varphi \Leftrightarrow \psi.$$

Remark 2.1.1 (Complexity of the quantifier elimination method, [12, 10])**.** When Tarski in in 50s proved that the theory of ordered real closed fields admitted quantifier elimination, the procedure presented was of non-elementary complexity. However, there given recent advances since the 70s, it was proven that the quantifier elimination procedure has complexity doubly exponential in the number of variables in the formula.

Theorem 2.1.3. *The theory $\Theta^{\vdash}_{ORCF}$ is complete and decidable.*

Remark 2.1.2 ([12, 10])**.** If ψ is a quantifier free formula in the signature Σ_{ORCF}, its existential closure $\exists\psi$ can be decided in PSPACE. If one needs to provide a satisfying assignment for ψ, the complexity becomes EXPSPACE.

The fundamental example of a real closed field is the real closure of the rational numbers, which will be denoted by $\mathbb{R}_{\mathsf{Alg}}$. In this field, we are able to perform algebraic reasoning involving probabilities, that is, our terms may consider products and sums of probabilities, and comparison of terms thus formed. Let φ be first-order formula over the signature Σ_{ORCF} with logical variables from a countable set Z, and let $\rho : Z \mapsto \mathbb{R}_{\mathsf{Alg}}$. We denote by $\rho \Vdash^{\mathsf{EAlg}} \varphi$ the first-order satisfaction relation between the field of real algebraic numbers $\mathbb{R}_{\mathsf{Alg}}$ (when seen as a first-order structure), the assignment ρ and the formula φ; while the presentation of the Probabilization of a Logic system can be done without a fixed domain, we chose to focus upon $\mathbb{R}_{\mathsf{Alg}}$ because probability theory is usually done with real numbers, and $\mathbb{R}_{\mathsf{Alg}}$ is the subset of the real numbers closed for the properties we wish to capture.

We note that the ordered real closed fields theory is not easily extended without the loss of many important properties. For instance, in the seminal work on quantifier elimination of real closed fields by **Tarski**, it was argued that the sine function could not be added to the theory, as it would allow to define the notion of an integer within the theory and therefore would lead directly to an undecidable theory. He also proposed the study of the introduction of the exponential function, however it is still not known whether the theory of the exponential real closed fields is decidable or not, and its decidability hinges on deep results in Transcendental Number Theory [73]. Another possibility would be to consider the field of computable real numbers, which is still a real closed field; however, simple comparisons like $c_1 = c_2$ for computable reals c_1 and c_2 are in general undecidable, which greatly limits the model-checking procedure.

2.1.3 The Probabilization of a Logic system

The main drive for the probabilization of a logic system is to gain the ability to quantify the likelihood of some property; instead of *for all concretizations φ holds*, or *there exists a concretization such that φ holds*, we are interested in stating quantitative assertions: *with probability* 1 *φ holds*, or *the probability of φ to hold is* 0.9. We wish however to retain the ability of making absolute statements about a property ϕ. There are some issues that need to be solved right from the start.

- We will restrict ourselves to logical systems which are *classical* in na-

ture, that is there exists in them *negation* and *disjunction* connectives which behave classically: $m \Vdash \neg\phi$ iff $m \nVdash \phi$ and $m \Vdash \phi \vee \psi$ iff $m \Vdash \phi$ or $m \Vdash \psi$. This is due to the fact that we wish to represent a property ϕ by the set of base models that satisfy it, which has to be a *measurable event*; however, unions and complements of events are also events, and as such we will require to have connectives in the logic that are able to mimic probabilistic reasoning.

- We need to choose what type of operations can be performed with probabilities. This is a relevant subject, with practical implications. In PSAT [48], one usually only considers linear combinations of probabilities, however, as already was pointed out by **Fagin et al.**, that choice makes extremely difficult the representation of independence between events, which is a central concept in Probability Theory. Independence of events A and B is defined by $P(A|B) = \frac{P(A \cap B)}{P(B)}$, with $P(B) > 0$, which can not be expressed as a linear combination of probabilities. Therefore, our choice was to allow reasoning about any algebraic combination of probabilities.

In this section we present the probabilistic scaffolding for the logics presented in Section 2.2 and Chapter 4. Let $\mathcal{L} = (\mathsf{L}, \mathsf{M}, \Vdash)$ be a satisfaction system, where L is the set of *well-formed formulae* of the logic $\mathcal{L}$, M is a set of *models*, and $\Vdash \subseteq \mathsf{L} \times \mathsf{M}$; we assume that L and M are not empty. Moreover, as we discussed earlier, we assume that if $\psi_1, \psi_2 \in \mathsf{L}$ then (1) $\psi_1 \vee \psi_2 \in \mathsf{L}$, (2)$\neg\psi_1 \in \mathsf{L}$, (3) $m \Vdash \psi_1 \vee \psi_2$ iff $m \Vdash \psi_1$ or $m \Vdash \psi_2$ and (4) $m \Vdash \neg\psi_1$ iff $m \nVdash \psi_1$. We define the syntax of the *Probabilization* of $\mathcal{L}$:

Definition 2.1.8. Let $\mathcal{L} = (\mathsf{L}, \mathsf{M}, \Vdash)$ be a satisfaction system obeying conditions (1)-(4). The Probabilization of $\mathcal{L}$, represented as $\mathcal{P}(\mathcal{L}) = (\mathsf{L}^p, \mathsf{M}^p, \Vdash^p)$ is defined below. The set of *well-formed formulae* L^p is given by Backus-Naur form:

$$t ::= x \quad | \quad (t + t) \quad | \quad (tt) \quad | \quad r \quad | \quad \int \beta \qquad \text{(probabilistic terms)}$$

$$\delta ::= \underbrace{\Box\beta}_{\text{global atoms}} \quad | \quad \sim \delta \quad | \quad \delta \cap \delta \quad | \quad \underbrace{t \leq t}_{\text{comparison atoms}} \qquad \text{(formulae)}$$

where $\beta \in \mathsf{L}$, $r \in \mathbb{R}_{\mathsf{Alg}}$, a real algebraic number, $x \in Z$, where Z is the set of *algebraic variables*.

A probabilistic model $(\mathcal{P}, \rho) \in \mathsf{M}^p$ is such that:

- $V \subseteq \mathsf{M}$, named the *sample space*,
- $\mathcal{F}$ is a σ-algebra over M such that $V \in \mathcal{F}$, and for all formulae $\beta \in \mathsf{L}$, $\{v \in V : v \Vdash \beta\} \in \mathcal{F}$, is named the *event space*,
- ν is a probability measure over $\mathcal{F}$, with the following provisos: $\nu(\{v \in V : v \Vdash \beta\}) \in \mathbb{R}_{\mathsf{Alg}}$, for any $\beta \in \mathsf{L}$ and $\nu(V) = 1$; it is named the *probability measure*,
- $\rho : Z \mapsto \mathbb{R}_{\mathsf{Alg}}$ is an *assignment* of real algebraic values.

The denotation of a probabilistic term $[\![t]\!]_{\mathcal{P},\rho}$ is a map that assigns a real algebraic value to any term t and it is given as follows:

- $[\![x]\!]_{(\mathcal{P},\rho)} = \rho(x)$,
- $[\![r]\!]_{(\mathcal{P},\rho)} = r$,
- $[\![\int \beta]\!]_{(\mathcal{P},\rho)} = \nu(\{v \in V : v \Vdash \beta\})$,
- $[\![(t_1{+}t_2)]\!]_{(\mathcal{P},\rho)} = [\![t_1]\!]_{(\mathcal{P},\rho)}{+}[\![t_2]\!]_{(\mathcal{P},\rho)}$ and $[\![(t_1 t_2)]\!]_{(\mathcal{P},\rho)} = [\![t_1]\!]_{(\mathcal{P},\rho)}{\cdot}[\![t_2]\!]_{(\mathcal{P},\rho)}$.

The satisfaction relation between a Probabilistic model $\mathcal{P}, \rho$, and a formula $\delta \in \mathsf{L}^p$ is defined inductively as follows:

- $\mathcal{P}, \rho \Vdash^p \Box\beta$ iff for all $v \in V, v \Vdash \beta$,
- $\mathcal{P}, \rho \Vdash^p (t_1 \leq t_2)$ iff $[\![t_1]\!]_{(\mathcal{P},\rho)} \leq [\![t_2]\!]_{(\mathcal{P},\rho)}$,
- $\mathcal{P}, \rho \Vdash^p (\sim\delta)$ iff $\mathcal{P}, \rho \not\Vdash^p \delta$,
- $\mathcal{P}, \rho \Vdash^p (\delta_1 \cap \delta_2)$ iff $\mathcal{P}, \rho \Vdash^p \delta_1$ and $\mathcal{P}, \rho \Vdash^p \delta_2$.

Remark 2.1.3. We note that $\mathcal{P}$ is not quite a probability space, as usually the σ-algebra of a probability space is a set of subsets of the sample space. In our case, we chose to work with a σ-algebra over the set of all models; this choice was motivated to allow us to use always the same σ-algebra for any possible sample space; any result proven can be easily adapted to the classical definition of a probability space.

Remark 2.1.4. We use the common classical abbreviations for conjunction $\cap$, equivalence $\Leftrightarrow$, implication $\supset$, and all the remaining comparison predicates $(<, >, \geq, =)$. Moreover, given that we assume the existence of classical negation and disjunction at the inner level formulae, we also use abbreviations for conjunction $\wedge$, equivalence $\leftrightarrow$, implication $\rightarrow$ and verum/falsum $\top/\bot$.

Remark 2.1.5. If a formula δ does not contain terms of the form $\int \beta$, or global atoms like $\Box\beta$, we say that δ is an *algebraic formula*. These formulae belong to a syntactic extension of the first-order language of ordered real closed fields due to the introduction of terms r which are representable in the theory. Let $\Sigma_{\mathsf{eORCF}} = \Sigma_{\mathsf{ORCF}} \cup \{r\}_{r\in\mathbb{R}_{\mathsf{Alg}}}$, where r is a constant term. Consider also $\Theta_{\mathsf{eORCF}} = \Theta_{\mathsf{ORCF}} \cup \{\forall x.x = r \Leftrightarrow \gamma_r(x)\}_{r\in\mathbb{R}_{\mathsf{Alg}}}$, where $\gamma_r(x)$ is a existentially quantified first-order formula over the signature Σ_{ORCF}, representing uniquely the real algebraic number r. Then not only Θ_{eORCF} is a conservative extension of Θ_{ORCF}, but also for any first-order formulae ϕ over Σ_{eORCF} there exists a translation of ϕ where each constant r is substituted by a fresh variable x_r, and the whole formula is appended by $x_r = \gamma_r(x_r)$.

Note that we can also choose $\gamma_r(x)$ to be quantifier free, due to Theorem 2.1.2.

If we prohibit the use of comparison predicates, we are considering the *global* version of the base logic, which was also described in [78]; these formulae allow us to reason with allowed behavior, whereas comparison predicates express the likelyhood of the behaviors.

Remark 2.1.6. In Section 2.2 we introduce EPPL as the probabilization of classical propositional logic PL; in Chapter 4, the probabilistic temporal logic PQLTL is introduced as the probabilization of Quantified Linear Temporal Logic QLTL.

We usually separate the model $(\mathcal{P}, \rho)$ in its two components, the probabilistic model $\mathcal{P}$ and the assignment ρ.

This probabilization of a logic system allows for any elementary algebraic reasoning, that is, we can express assertions like:

- The event β_1 is more likely than the event β_2: $\int \beta_1 > \int \beta_2$.
- If β holds always, then its probability is 1: $\Box\beta \supset \int \beta = 1$.
- The probability of β_1 plus the probability of β_2 squared is greater than the square root of the probability of β_3 minus $\frac{1}{2}$: $\int \beta_1 + \int \beta_2 \int \beta_2 > x \cap xx = \int \beta_3 + (-1)\frac{1}{2} \cap x \geq 0$.
- Moreover, we can also express independence between events, as the multiplication of probabilistic terms is possible: in order to express that β_1 and β_2 are independent, we simply assert $\int \beta_1 \wedge \beta_2 = \int \beta_1 \int \beta_2$.

However, in some cases, we may wish to consider a less expressive probabilistic logic, that only allows linear combinations of probabilistic terms.

These logics will not require the machinery of real closed fields, therefore instead of their satisfaction algorithms solving systems of polynomial inequations, they will simply need to solve linear systems.

Definition 2.1.9. Let $\mathcal{L} = (\mathsf{L}, \mathsf{M}, \Vdash)$ be a satisfaction system obeying conditions (1)-(4). The Linear Probabilization of $\mathcal{L}$, represented as $\mathcal{LP}(\mathcal{L}) = (\mathsf{L}^{lp}, \mathsf{M}^{lp}, \Vdash^{lp})$ is defined below. The set of *well-formed formulae* L^{lp} is given by Backus-Naur form:

$$t ::= \quad | \quad x \quad | \quad (t+t) \quad | \quad q \quad | \quad q \int \beta \qquad \text{(probabilistic terms)}$$

$$\delta ::= \underbrace{\Box\beta}_{\text{global atoms}} \quad | \quad \sim\delta \quad | \quad \delta \cap \delta \quad | \quad \underbrace{t \leq t}_{\text{comparison atoms}} \qquad \text{(formulae)}$$

where $\beta \in \mathsf{L}$, $r \in \mathbb{Q}$, $x \in Z$ a rational variable .

A probabilistic model $(\mathcal{P}, \rho) \in \mathsf{M}^{lp}$ is such that:

- $V \subseteq \mathsf{M}$, named the *sample space*,
- $\mathcal{F}$ is a σ-algebra over M such that $V \in \mathcal{F}$, and for all formulae $\beta \in \mathsf{L}$, $\{v \in V : v \Vdash \beta\} \in \mathcal{F}$, is named the *event space*,
- ν is a probability measure over $\mathcal{F}$, with the following provisos: $\nu(\{v \in V : v \Vdash \beta\}) \in \mathbb{Q}$, for any $\beta \in \mathsf{L}$ and $\nu(V) = 1$; it is named the *probability measure*,
- $\rho : Z \mapsto \mathbb{Q}$ is an *assignment* of rational values.

The denotation of a probabilistic term $[\![t]\!]_{\mathcal{P},\rho}$ is a map that assigns a rational value to any term t and it is given as follows:

- $[\![x]\!]_{(\mathcal{P},\rho)} = \rho(x)$,
- $[\![q]\!]_{(\mathcal{P},\rho)} = q$,
- $[\![q \int \beta]\!]_{(\mathcal{P},\rho)} = q\nu(\{v \in V : v \Vdash \beta\})$,
- $[\![(t_1 + t_2)]\!]_{(\mathcal{P},\rho)} = [\![t_1]\!]_{(\mathcal{P},\rho)} + [\![t_2]\!]_{(\mathcal{P},\rho)}$.

The satisfaction relation between a model $\mathcal{P} = (V, \mathcal{F}, \nu)$ an assignment $\rho : Z \mapsto \mathbb{Q}$, and a formula $\delta \in \mathsf{L}^{lp}$ is defined inductively as follows:

- $\mathcal{P}, \rho \Vdash^{lp} \Box\beta$ iff for all $v \in V, v \Vdash \beta$,

- $\mathcal{P}, \rho \Vdash^{lp} (t_1 \leq t_2)$ iff $[\![t_1]\!]_{(\mathcal{P},\rho)} \leq [\![t_2]\!]_{(\mathcal{P},\rho)}$,
- $\mathcal{P}, \rho \Vdash^{lp} (\sim\delta)$ iff $\mathcal{P}, \rho \nVdash^{lp} \delta$,
- $\mathcal{P}, \rho \Vdash^{lp} (\delta_1 \cap \delta_2)$ iff $\mathcal{P}, \rho \Vdash^{lp} \delta_1$ and $\mathcal{P}, \rho \Vdash^{lp} \delta_2$.

2.2 Exogenous Probabilistic Propositional Logic

Herein, we review the syntax and semantics for EPPL introduced in [78]. For the sake of readability, we present some already known results concerning EPPL. We note that this logic has arisen as a version of the fundamental probabilistic logic in [38], and it is particularly useful as a building block for extended logics that require some degree of probabilistic reasoning, like [76, 77, 24]. Furthermore, it also arises as the end result of the probabilization of classical propositional logic, a according to the definitions presented in the previous section.

The logic EPPL allows for reasoning about probability spaces over classical valuations. We assume that $\Lambda = \{p_1, \ldots, p_n, \ldots\}$ is a countable set of propositional variables; unless there is some restriction in place, we assume that Λ is infinite. Moreover, we assume that Z is a countable infinite set of logical variables.

Definition 2.2.1. The *well formed formulae* in $\mathsf{EPPL}(\Lambda)$ are given below in Backus-Naur notation:

$$\beta ::= p \quad | \quad (\beta \wedge \beta) \quad | \quad (\neg\beta) \quad | \quad \top \qquad \text{(basic formulae)}$$

$$t ::= x \quad | \quad (t + t) \quad | \quad (tt) \quad | \quad r \quad | \quad \int \beta \qquad \text{(probabilistic terms)}$$

$$\delta ::= \underbrace{\Box\beta}_{\text{global atoms}} \quad | \quad \sim\delta \quad | \quad \delta \cap \delta \quad | \quad \underbrace{t \leq t}_{\text{comparison atoms}} \qquad \text{(formulae)}$$

where $p \in \Lambda$, $\top$ is constant true (*verum*), $x \in Z$ and $r \in \mathbb{R}_{\mathsf{Alg}}$ is an algebraic real number.

Basic formulae β are classical propositional formulae. These basic propositional formulae can be used to build two different types of global atoms: either by using the connective $\Box$ (which can be interpreted as "necessarily") and obtaining a global atom, or by first producing *probabilistic terms*, using the $\int$ connective; these terms can be further combined with algebraic expressions formed using an infinite countable set of real variables Z; finally

these probabilistic terms can be compared with the connective $\leq$, producing a comparison atom.

We can then reason classically with global or comparison atoms achieved by the processes described above. The abbreviations on the global level for disjunction $\cup$, implication $\supset$, equivalence $\Leftrightarrow$ will be also used for classical global reasoning, using also $\Diamond\beta \equiv \sim\Box\neg\beta$ (which can then be interpreted as "possibly") and all the usual predicate abbreviations for $=, <, >, \geq$. We further note that $pvar(\delta) \subseteq \Lambda$ represents the set of propositional symbols used in the construction of δ; the set of (algebraic) variables in a formula δ is represented by $var(\delta)$. The notation $\delta(p, \mathbf{r})$ is meant to convey the fact that p appears as a propositional symbol in δ, while $\mathbf{r}$ represents all other (possibly none) propositional symbols appearing in δ.

Before presenting the models of EPPL formulae, we will need to introduce some simplifying notation. If $v \in \{0,1\}^{\Pi}$ and $v' \in \{0,1\}^{\Pi\cup\{q\}}$ such that $v'(q) = 1$ for some propositional variable q not already in Π and $v'(p) = v(p)$ for all $p \in \Pi$, we may write v' as $v, q \mapsto 1$. Furthermore $\overline{v}^p$ represents the valuation such that $dom(\overline{v}^p) = dom(v)$, $\overline{v}^p(p) \neq v(p)$ and $v(q) = \overline{v}^p(q)$, for all other propositional symbols; the notation $\overline{V}^p$ is the extension towards sets, that is $\overline{V}^p = \{\overline{v}^p : v \in V\}$. Note that in general, $V \cap \overline{V}^p \neq \emptyset$.

Even if the syntax of EPPL does in fact follows Definition 2.1.8, one must however take some care when defining the semantics. In Definition 2.1.8, one needs to choose a σ-algebra over the set of all valuations, and then, we need to make sure that all basic formulae are measureable in the chosen σ-algebra. If the set of propositional symbols is finite, then the natural choice for the σ-algebra is the powerset of the set of all valuations. Moreover, if one assigns a probability to all the singleton sets, the probability space is completely defined. When considering a denumerable set of propositional symbols, this approach is not as useful, as the σ-additivity of the measure is of no use since propositional formulae will correspond to uncountable unions of singleton sets.

Consider $v \in \{0,1\}^{\Pi}$, such that $\Pi \subseteq \Lambda$ and Π is a finite set. The sets of the following form are usually named *cylinders*:

$$B_v = \{w \in \{0,1\}^{\Lambda} : w|_{dom(v)} = v\}.$$

The set of all cylinders $\mathcal{B} = \{B_v : dom(v) = \Pi, \#\Pi < +\infty, \Pi \subseteq \Lambda\} \cup \{\emptyset\}$ is a semi-ring. Then, Carathéodory's extension theorem [61, 23] allow us to conclude that if one has a pre-measure $\mu_{\mathcal{B}}$ defined on the semi-ring, there exists a measure μ defined on the σ-algebra generated by $\mathcal{B}$, such that its restriction to the semi-ring $\mathcal{B}$ is $\mu_{\mathcal{B}}$. In fact, if we are dealing with probability

spaces, the induced measure is unique. We denote this σ-algebra to be $\sigma(\{0,1\}^\Lambda)$. Moreover, $\sigma(\{0,1\}^\Lambda)$ is also the Borel σ-algebra when $\mathcal{B}$ is seen as the basis of the product topology over $\{0,1\}$ [23].

Definition 2.2.2. A EPPL *model* $\mathcal{P}$ is a tuple $\mathcal{P} = (V, \sigma(\{0,1\}^\Lambda), \nu)$ satisfying the following conditions:

- the first component named the *sample space* is such that $V \subseteq \{0,1\}^\Lambda$,
- the second component is the Borel σ-algebra generated by the cylinders, such that $V \in \sigma(\{0,1\}^\Lambda)$,
- the third component, named the *probability*, is a probability measure $\nu : \sigma(\{0,1\}^\Lambda) \mapsto [0;1]$ over the standard σ-algebra, verifying $\nu(V) = 1$ and $\nu(B_v) \in \mathbb{R}_{\mathsf{Alg}}$ for any cylinder B_v. Note that this condition also ensures that $V \neq \emptyset$.

The set of all EPPL models over Λ will be denoted as $\mathfrak{M}(\mathsf{EPPL}(\Lambda))$. However, if Λ is finite, the σ-algebra can be set as $\mathcal{P}(\Lambda)$, still keeping the proviso that $\nu(V) = 1$.

Definition 2.2.3. Given a probabilistic term t and a EPPL model $\mathcal{P} = (V, \sigma(\{0,1\}^\Lambda), \nu)$ over Λ and an assignment $\rho : Z \mapsto \mathbb{R}_{\mathsf{Alg}}$ the *denotation* of t in $\mathcal{P}$ and ρ, represented as $[\![t]\!]_{(\mathcal{P},\rho)}$ or as $[\![t]\!]_{\mathcal{P}}$ if the term does not contain variables, is given inductively as follows:

- $[\![x]\!]_{(\mathcal{P},\rho)} = \rho(x)$,
- $[\![r]\!]_{(\mathcal{P},\rho)} = r$,
- $[\![\int \beta]\!]_{(\mathcal{P},\rho)} = \nu(\{v \in V : v \Vdash \beta\})$,
- $[\![(t_1{+}t_2)]\!]_{(\mathcal{P},\rho)} = [\![t_1]\!]_{(\mathcal{P},\rho)}{+}[\![t_2]\!]_{(\mathcal{P},\rho)}$ and $[\![(t_1 t_2)]\!]_{(\mathcal{P},\rho)} = [\![t_1]\!]_{(\mathcal{P},\rho)}.[\![t_2]\!]_{(\mathcal{P},\rho)}$.

The satisfaction relation $\Vdash$ (without superscripts) between a valuation $v \in \{0,1\}^\Lambda$ and a basic formula β is the satisfaction relation in classical propositional logic; note that the set $\{v \in V : v \Vdash \beta\}$ is always measureable in $\sigma(\{0,1\}^\Lambda)$, since it is an union of cylinders intersected with a measureable set V. Moreover, $\nu(\{v \in V : v \Vdash \beta\}) = \nu(\{v \in \{0,1\}^\Lambda : v \Vdash \beta\})$, since $\nu(V) = 1$.

We can now define the satisfaction relation for our logic. Usually we will represent it as $\Vdash^{\mathsf{EPPL}(\Lambda)}$, however, whenever the set of propositional variables can be understood from context, we drop it from the superscript.

Definition 2.2.4. The satisfaction relation between a EPPL model $\mathcal{P} = (V, \sigma(\{0,1\}^\Lambda), \nu)$ over Λ, an assignment $\rho : Z \mapsto \mathbb{R}_{\mathsf{Alg}}$, and a formula $\delta \in \mathsf{EPPL}(\Lambda)$ is defined inductively as follows:

- $\mathcal{P}, \rho \Vdash^{\mathsf{EPPL}(\Lambda)} \Box\beta$ iff for all $v \in V, v \Vdash \beta$,
- $\mathcal{P}, \rho \Vdash^{\mathsf{EPPL}(\Lambda)} (t_1 \leq t_2)$ iff $[\![t_1]\!]_{(\mathcal{P},\rho)} \leq [\![t_2]\!]_{(\mathcal{P},\rho)}$,
- $\mathcal{P}, \rho \Vdash^{\mathsf{EPPL}(\Lambda)} (\sim\delta)$ iff $\mathcal{P}, \rho \nVdash^{\mathsf{EPPL}(\Lambda)} \delta$,
- $\mathcal{P}, \rho \Vdash^{\mathsf{EPPL}(\Lambda)} (\delta_1 \cap \delta_2)$ iff $\mathcal{P}, \rho \Vdash^{\mathsf{EPPL}(\Lambda)} \delta_1$ and $\mathcal{P}, \rho \Vdash^{\mathsf{EPPL}(\Lambda)} \delta_2$.

The notion of *semantic entailment* can now be defined: if $\Delta \subseteq \mathsf{EPPL}(\Lambda)$ and $\gamma \in \mathsf{EPPL}(\Lambda)$, then Δ entails γ (represented as $\Delta \models^{\mathsf{EPPL}(\Lambda)} \gamma$) if and only if for any models $\mathcal{P}$ and assignments ρ such that $\mathcal{P}, \rho \Vdash \delta$, for all $\delta \in \Delta$, then $\mathcal{P}, \rho \Vdash \gamma$. If $\emptyset \models^{\mathsf{EPPL}(\Lambda)} \gamma$, we say that γ is a *valid* formula.

Example 2.2.1. Let $\Lambda := \{p_1, p_2, p_3\}$. We present some assertions in order to give some insight into EPPL formulae:

- The formula $\int p_1 \wedge p_2 \geq \int p_1 \wedge p_3$ expresses the assertion that p_1 and p_2 is more likely to be true than p_2 and p_3.
- The formula $\int p_1 \wedge p_2 = \int p_1 . \int p_2$ expresses the assertion that events p_1 and p_2 are independent.
- The formula $(\Box p_1 \cup \Box\neg p_1) \supset \int p_1 . \int p_1 - \int p_1 = 0$ is a valid formula. Suppose that a model satisfies the left-hand side of the implication. Then either all valuations v satisfy p_1 or all valuations do not satisfy p_1. Therefore either the event p_1 occurs almost surely or it has null measure.
- The formula $(\Box p_1 \cup \Box\neg p_1) \Leftrightarrow \int p_1 . \int p_1 - \int p_1 = 0$ is not a valid formula. Suppose that there exists a valuation such that $v(p_1) = 1$, but the valuation has measure 0; moreover all other valuations verify $v(p_1) = 0$. Then the right-hand side of the equivalence would hold, as $\int p_1 = 0$, but neither $\Box p_1$ nor $\Box\neg p_1$ would hold.

As it can be seen from the previous example, we may use the connective $\Box$ to reason absolutely, without being restricted to speak about events up to almost sure equivalency. In fact in [78], the authors construct EPPL as an extension of a Global Logic (GL); as we mentioned, this logic can be obtained

by eliminating all the comparison atoms in a formula. The logic so obtained is not modal in nature (although closely related), as the logic does not allow nesting of $\Box$ connectives. Furthermore, unlike [38], we note that we do not allow quantification over variables $z \in Z$ in EPPL, although we believe it would not impact negatively the main results presented in this work.

Theorem 2.2.1. *The Hilbert calculus presented in Table 2.2 is sound and (weakly) complete with respect to the EPPL semantic entailment relation.*

Axioms			
[**CTaut**]	$\vdash^{\mathsf{EPPL}}$	$\Box\beta$	for each valid propositional formula β
[**GTaut**]	$\vdash^{\mathsf{EPPL}}$	δ	for each EPPL instantiation δ of a tautological propositional formula;
[**Lift** $\Rightarrow$]	$\vdash^{\mathsf{EPPL}}$	$\Box(\beta_1 \Rightarrow \beta_2) \supset (\Box\beta_1 \supset \Box\beta_2)$	
[**Eqv** $\top$]	$\vdash^{\mathsf{EPPL}}$	$\Box\top \Leftrightarrow (\Box(\beta_1 \vee \neg\beta_1))$	
[**Prob**]	$\vdash^{\mathsf{EPPL}}$	$(\int \varphi = 1)$	for each classical propositional valid formula φ;
[**ORCF**]	$\vdash^{\mathsf{EPPL}}$	$(t_1 \leq t_2)$	for each instantiation of a valid analytical inequality;
[**FAdd**]	$\vdash^{\mathsf{EPPL}}$	$(\int(\beta_1 \vee \beta_2) = \int \beta_1 + \int \beta_2 - \int(\beta_1 \wedge \beta_2)))$	
[**Mon**]	$\vdash^{\mathsf{EPPL}}$	$((\int (\beta_1 \Rightarrow \beta_2) = 1) \supset (\int \beta_1 \leq \int \beta_2))$	
Inference rules			
[**MP**]	$\delta_1, (\delta_1 \supset \delta_2) \vdash^{\mathsf{EPPL}} \delta_2$.		

Table 2.2: A Hilbert calculus for EPPL

The logic EPPL is decidable, and the satisfiability algorithm runs in PSPACE.

Definition 2.2.5. Let $\mathcal{P} = (V, \sigma(\{0,1\}^{\Lambda}), \nu)$ be a EPPL model over Λ. Then, the *reduct of* $\mathcal{P}$ in Π is defined as $\mathcal{P}|_{\Pi} = (V', \mathcal{P}(\{0,1\}^{\Pi}), \nu')$, where $\Pi \subseteq \Lambda, \#\Pi < +\infty$, is defined as follows:

- $V' = \{v \in \{0,1\}^{\Pi} : \text{ exists } w \in V \text{ s.t. } w|_{\Pi} = v\}$;
- $\nu'(B_v) = \nu(B_v)$ for any valuation v defined over any subset of Π.

Lemma 2.2.1 (Lemma of absent variables)**.** *Let $\delta \in \mathit{EPPL}(\Lambda)$ and $\mathcal{P} = (V, \sigma(\{0,1\}^{\Lambda}), \nu) \in \mathfrak{M}(\mathit{EPPL}(\Lambda))$, and let $\rho : Z \mapsto \mathbb{R}_{\mathit{Alg}}$. Then,*

$$\mathcal{P}, \rho \Vdash^{\mathit{EPPL}(\Lambda)} \delta \text{ iff } \mathcal{P}|_{pvar(\delta)}, \rho \Vdash^{\mathit{EPPL}(pvar(\delta))} \delta.$$

Proof. Let $\mathcal{P}|_{pvar(\delta)} = (V', \mathcal{P}(\{0,1\}^{pvar(\delta)}), \nu')$; the proof follows by structural induction in δ.

- δ is $(\Box\beta)$: then, $\mathcal{P}, \rho \Vdash^{\mathsf{EPPL}(\Lambda)} \Box\beta$ iff for all $v \in V$, $v \Vdash \beta$, iff for all $v \in V'$, $v \Vdash \beta$ iff $\mathcal{P}|_{pvar(\delta)}, \rho \Vdash^{\mathsf{EPPL}(pvar(\delta))} \Box\beta$. In the second equivalence we use the fact that omitted variables are irrelevant in classical propositional logic.

- δ is $(t_1 \leq t_2)$: we show that $[\![t_1]\!]_{\mathcal{P},\rho} = [\![t_1]\!]_{\mathcal{P}|_{pvar(\delta),\rho}}$. The non-obvious case is when t_1 is $\int \beta$. Note that

$$\begin{aligned}\nu(\{v \in V : v \Vdash \beta\}) &= \sum_{v \in \{0,1\}^{pvar(\delta)}, v \Vdash \beta} \nu(B_v \cap V) \\ &= \sum_{v \in \{0,1\}^{pvar(\delta)}, v \Vdash \beta} \nu'(B_v \cap V|_{pvar(\delta)}),\end{aligned}$$

  where the last equality is due to the definition of $\mathcal{P}|_{pvar(\delta)}$.

The step cases are direct applications of the induction hypothesis.

□

This lemma allows us to focus on finite probabilistic spaces. The following lemma, which will be useful later on, shows that both global atoms and terms of the form $\int \beta$ can be substituted, leaving only algebraic equalities/inequalities; in order to do so, we need to enrich the assignment ρ in order to codify the probability space $\mathcal{P}$. Recall that an algebraic formula is a first-order formula constructed using the signature of ordered real closed fields. Thanks to **Tarski**'s result on quantifier elimination, it is known that any formula constructed with this signature can be decided in the first-order theory of real closed fields.

The following lemma is a possible approach for showing that EPPL is decidable.

Lemma 2.2.2. *Let γ be a $\mathsf{EPPL}(\Lambda)$ formula and let $\mathcal{P} = (V, \mathcal{P}(pvar(\gamma)), \nu) \in \mathfrak{M}(\mathsf{EPPL}(pvar(\gamma)))$. Then, there exists an elementary algebraic formula without quantifiers δ^γ and a map $\rho^\gamma_\mathcal{P} : var(\delta^\gamma) \cup Z \mapsto \mathbb{R}_{\mathsf{Alg}}$ extending ρ such that:*

$$\mathcal{P}, \rho \Vdash^{\mathsf{EPPL}(pvar(\gamma))} \gamma \text{ iff } \rho^\gamma_\mathcal{P} \Vdash^{\mathsf{EAlg}} \delta^\gamma.$$

Proof. We start by providing the candidate translation δ^γ of the formula γ. We assume w.l.o.g. that γ is in disjunctive normal form (DNF). Furthermore, we also assume that all the classical propositional formulae β at the

local level are in disjunctive normal form, and moreover, that each conjunct has $\#(pvar(\gamma))$ distinct elements. In this way, the expression for β is the disjunction of the valuations over $pvar(\gamma)$ that satisfy β. The translation operator, represented as $(.)^-$, will be built on the structure of γ. Let x_{v_i} and $x_{\nu(v_i)}$ be fresh variables, with $v_i \in \{0,1\}^{pvar(\gamma)}$

- $(\gamma_1 \cup \gamma_2)^- \equiv (\gamma_1^- \vee \gamma_2^-)$,
- $(\gamma_1 \cap \gamma_2)^- \equiv (\gamma_1^- \wedge \gamma_2^-)$,
- $(\Diamond\beta)^- = (\Diamond \bigvee_i \beta_{v_i})^- \equiv (\bigvee_i (x_{v_i} = 1))$,
- $(t_1 \leq t_2)^- \equiv (t_1)^{--} \leq (t_2)^{--}$,
- $(\sim \Diamond\beta)^- = (\sim \Diamond \bigvee_i \beta_{v_i})^- \equiv (\bigwedge_i (x_{v_i} = 0))$,
- $(t_1 > t_2)^- \equiv (t_1)^{--} > (t_2)^{--}$.

The auxiliary translation $(.)^{--}$ is needed to deal with probabilistic terms.

- $(r)^{--} \equiv r$ (see Remark 2.1.5),
- $(t)^{--}(x)^{--} \equiv x$,
- $(t_1 + t_2)^{--} \equiv t_1^{--} + t_2^{--}$,
- $(t_1.t_2)^{--} \equiv t_1^{--}.t_2^{--}$,
- $(\int \beta)^{--} = (\int \bigvee_i \beta_{v_i}) \equiv \sum_i x_{v_i}.x_{\nu(v_i)}$.

Recall that in last item, we are only considering the disjunction of the valuations that satisfy β. Set $\delta^\gamma = (\gamma)^-$.

The assignment $\rho^\gamma_{\mathcal{P}}$ will extend the assignment ρ with the added information about the probability space $\mathcal{P}$ contained in the assignment of the added variables; $\rho^\gamma_{\mathcal{P}}(x_{v_i}) = 1$ if $v_i \in V$ and it is 0 otherwise; furthermore $\rho^\gamma_{\mathcal{P}}(x_{\nu(v_i)})$ is equal to $\nu(B_{v_i} \cap V) = \nu(B_{v_i})$. Finally, $\rho^\gamma_{\mathcal{P}}(x) = \rho(x)$ for all variables x that were not introduced in the process. The proof of the lemma follows by induction on the structure of γ.

We will start by showing that $\mathcal{P}, \rho \Vdash^{\mathsf{EPPL}(pvar(\gamma))} \Diamond\beta$ iff $\rho^\gamma_{\mathcal{P}} \Vdash^{\mathsf{EAlg}} (\bigvee_i (x_{v_i} = 1))$.

If $(\mathcal{P}, \rho) \Vdash^{\mathsf{EPPL}} \Diamond\beta$, then there has to exist a valuation $v \in V$ such that $v \Vdash \beta$, and therefore v is disjunct in the DNF of β. Then $\rho^\gamma_{\mathcal{P}}(x_v) = 1$, and so $\rho^\gamma_{\mathcal{P}} \Vdash^{\mathsf{EAlg}} (\bigvee_i (x_{v_i} = 1))$.

Consider now that $\rho^\gamma_{\mathcal{P}} \Vdash^{\mathsf{EAlg}} (\bigvee_i (x_{v_i} = 1))$. Then, there exists j such that $\rho^\gamma_{\mathcal{P}}(x_{v_j}) = 1$; then $v_j \in V$ and therefore $(\mathcal{P}, \rho) \Vdash^{\mathsf{EPPL}} \Diamond\beta$. The proof concerning $\sim \Diamond\beta$ is similar.

We now show that $[\![t]\!]_{(\mathcal{P},\rho)} = (t)^{--}$. The only difficult case is the one concerning $[\![\int \beta]\!]_{(\mathcal{P},\rho)}$. Note that $[\![\int \beta]\!]_{(\mathcal{P},\rho)} = \sum_i [\![\int \beta_{v_i}]\!]_{(\mathcal{P},\rho)}$ since β_{v_i} correspond to disjoint cylinders. Furthermore, $[\![\int \beta_{v_i}]\!]_{(\mathcal{P},\rho)} = \nu(V \cap B_{v_i}) = \nu(\{v_i\})$. If the cylinder is non-empty, $\rho^\gamma_{\mathcal{V}}(x_{v_i}) = 1$, and so $x_{v_i} x_{\nu(v_i)} = \nu(\{v_i\})$. If the cylinder is empty, then $x_{v_i} = 0$ and so $x_{v_i} x_{\nu(v_i)} = \nu(\{v_i\}) = 0$.

The step cases are trivial. □

This lemma is enough to prove that the satisfiability problem for EPPL is decidable. Given a EPPL formula γ, we can construct its corresponding formula δ^γ, which is a first-order formula over Σ_{ORCF}, and by applying the decidability procedure of the theory Θ_{ORCF} to its existential closure, we then know whether γ is satisfiable. However, this method can be improved upon, as the method described involves an exponential blowup when translating γ to δ^γ. As mentioned in Remark 2.1.2, since the decidability of existential theory of the real closed fields lies in PSPACE, this would imply that the satisfiability problem would be in EXPSPACE. However, in [8], it is presented a more efficient approach that avoids the exponential blowup. In order to do so, we need two small model theorems, one for the global component of EPPL, and another for the probabilistic component. The size of a EPPL formula δ is the number of symbols used in writing δ.

Lemma 2.2.3 (Small model theorem for global formulae, [8])**.** *Suppose that $\delta \in$ EPPL is a satisfiable global formula; then there exists a model $\mathcal{P}' = (V', \mathcal{P}(\{0,1\}^{pvar(\delta)}, \nu')$ such that $\#V \leq |\delta|$ such that $\mathcal{P}' \Vdash^{EPPL(pvar(\delta)} \delta$.*

Sketch. Assume that δ is written in disjunctive normal form. Then,

$$\delta \equiv \bigcup_i (\Diamond\beta_i^1 \cap \ldots \Diamond\beta_i^{j-1}) \cap (\sim \Diamond\beta^j \cap \ldots \sim \Diamond\beta_i^k).$$

If δ is satisfiable, at least one of the disjuncts is satisfiable; each disjunct has at most size $|\delta|$; a diamond formula $\Diamond\beta$ enforces the existence of a valuation satisfying β; a negated $\Diamond$ formula, $\sim \Diamond\beta$ prohibits the existence of valuations satisfying β; however, there are at most $|\delta|$ diamond formulas, therefore, the worst possible case is that one has to choose a different valuation for each $\Diamond\beta^l$ formula. So, if δ is satisfiable, there has to exist a model satisfying δ with at most $|\delta|$ valuations. The probabilistic component is irrelevant for global formulae. □

Lemma 2.2.4 (Small model theorem for probabilistic formulae, [8])**.** *Suppose that $\delta \in \mathsf{EPPL}(\Lambda)$ is a satisfiable formula, without global atoms; then there exists a model $\mathcal{P}' = (V', \mathcal{P}(\{0,1\}^{pvar(\delta)}, \nu')$ and an assignment ρ such that $\#V \leq |\delta|$ and $\mathcal{P}', \rho \Vdash^{\mathsf{EPPL}(pvar(\delta)} \delta$.*

Sketch. Suppose that δ is satisfiable. Then there exists a EPPL model over $pvar(\delta)$, $\mathcal{P}$, and an assignment $\rho : Z \mapsto \mathbb{R}_{\mathsf{Alg}}$ such that $\mathcal{P}, \rho \Vdash^{\mathsf{EPPL}(pvar(\delta)} \delta$. We construct another EPPL model $\mathcal{P}' = (V', \mathcal{P}(\{0,1\}^{pvar(\delta)}, \nu')$ with $\#V' \leq |\delta|$ as follows. Start by collecting all the $\int \beta$ terms from δ. Then consider the following linear system, over variables x_v, where $v \in \{0,1\}^{pvar(\delta)}$:

$$\begin{aligned}
\sum_{v \Vdash \beta_1} x_v &= [\![\int \beta_1]\!]_{\mathcal{P},\rho} \\
\sum_{v \Vdash \beta_2} x_v &= [\![\int \beta_2]\!]_{\mathcal{P},\rho} \\
&\cdots \\
\sum_{v \Vdash \beta_n} x_v &= [\![\int \beta_n]\!]_{\mathcal{P},\rho} \\
\sum_{v} x_v &= 1.
\end{aligned}$$

This system has a non-negative solution, since δ is satisfiable, given by assigning x_v the value $\nu(B_v)$. However, from Linear Programming, it is known that if a linear system of $k+1$ equations has a non-negative solution, then there exists another solution where only $k+1$ variables take positive values [25]. Then, we just need to take the valuations with positive values in order to build our sample space V'. Moreover, the probability of each of those valuations is also known, therefore the probability measure ν' is also defined. Since the new model $\mathcal{P}'$ assigns exactly the same value to all the probabilistic terms as $\mathcal{P}$ did, $\mathcal{P}', \rho \Vdash^{\mathsf{EPPL}(pvar(\delta))} \delta$. □

Let δ be a EPPL formula. The global atoms occuring in δ will be denoted by $gatom(\delta)$; the probabilistic comparison terms are denoted by $patm(\delta)$, and $atm(\delta) = patm(\delta) \cup gatm(\delta)$. Let Π be a set of propositional symbols in bijective correspondence λ with $atm(\delta)$, that is $\lambda : atm(\delta) \mapsto \Pi$ is a bijection. One can consider the classical propositional formula obtained by substituting all the atoms for propositional symbols, which we denote by $\lambda(\delta)$. If this PL formula is not satisfiable, then δ is also unsatisfiable. Consider a valuation $v \in \{0,1\}^{\Pi}$; then we can consider $\lambda^{-1}(\gamma_v)$, the EPPL formula induced

by the propositional valuation v. This formula is a conjunction of global atoms and comparison predicates. Let $global(\lambda^{-}1(\gamma_v))$ represent the conjunction of global atoms and $prob(\lambda^{-}1(\gamma_v))$ the conjunction of comparison predicates.

Theorem 2.2.2. *Algorithm 1 is correct; δ is satisfiable if and only if the algorithm returns a model satisfying δ. Its complexity is EXPSPACE, when providing a satisfable model, and PSPACE when providing a yes/no answer.*

In order to only check for satisfiability, the complexity is PSPACE. The real closed fields solver has only to output whether there exists a solution, and as it receives a polynomial formula on δ, due to the Lemma 2.2.4 and Remark 2.1.2, the algorithm runs in PSPACE. However, in order to present the probabilistic model, the real closed fields solver has to run in EXPSPACE, so the complexity of the satisfiability problem becomes EXPSPACE.

The model-checking procedure is straightforward. Regarding the probabilistic part of the formula, one simply evaluates each term $\int \beta$ by adding the probabilities of all the valuations in the model that satisfy β, and then one checks which comparison predicates do hold in the model. Regarding the global part, one simply checks which global atoms do hold in the model; using information from both the global part and the probabilistic part, one verifies directly whether the whole formula holds in the model. Regarding the complexity of the algorithm, if one assumes that the additions and multiplications are done in constant time, one obtains that the time complexity of the procedure is $O(|\delta|.|\mathcal{P}, \rho|)$. For a concrete presentation of the algorithm, consult [8], Algorithm 5.

Algorithm 1: Satisfiability algorithm for EPPL

Input: EPPL formula δ
Output: EPPL model $\mathcal{P} = (V, \mathcal{P}(\{0,1\}^{pvar(\delta)}, \nu)$ and assignment ρ or *no model*

compute $atm(\delta), \lambda : \Pi \mapsto atm(\delta), \lambda(\delta)$;
foreach $v \in \{0,1\}^{\Pi}$ *s.t.* $v \Vdash \lambda(\delta)$;
do
  compute $\lambda^{-1}(\gamma_v), global(\lambda^{-1}(\gamma_v)), prob(\lambda^{-1}(\gamma_v))$;
  foreach $V \subseteq \{0,1\}^{pvar(\delta)}$ *s.t.* $\#V < 2|\delta| + 1$;
  do
    if $V \Vdash global(\lambda^{-1}(\gamma_v))$ **then**
      foreach $\int \beta \in prob(\lambda^{-1}(\gamma_v))$;
      do
        $\int \beta \mapsto \sum_{u \in V, u \Vdash \beta} x_u$
      end
      foreach $u \in V$ **do**
        $prob(\lambda^{-1}(\gamma_v)) \mapsto prob(\lambda^{-1}(\gamma_v)) \cap (x_u \geq 0)$
      end
      compute
      $prob(\lambda^{-1}(\gamma_v)) \mapsto prob(\lambda^{-1}(\gamma_v)) \cap (\sum_{u \in V} x_u = 1)$;
      compute $\kappa \mapsto \exists RCFSolver(prob(\lambda^{-1}(\gamma_v)))$;
      if κ *has a solution* **then**
        compute *organize* κ *constructing* $(V, \mathcal{P}(\{0,1\}^{pvar(\delta)}, \nu)$ *and assignment* ρ;
        return $(V, \mathcal{P}(\{0,1\}^{pvar(\delta)}, \nu), \rho$
      end
    end
  end
end
return *no model*

Chapter 3

Temporal and Probabilistic Reasoning

In this chapter, we review the more common techniques used that combine temporal and probabilistic reasoning. There has been a growing interest in the combination of the two fields since the emergence of Model Checking as a verification technique. The combination of temporal and probabilistic reasoning is justified by the following needs:

- protocols or algorithms with a probabilistic component: many algorithms are designed to have a discrete probabilistic component, namely by the use of coin tosses; if one does not analyze the resulting system from the point of view of probability theory, one is left with a non-deterministic system which will not model adequately the intended use of the protocol.
- modeling and quantifying uncertain behavior of an error-prone evolving system: failure of communications, unintended interactions with the environment, component malfunctions and similar events can be modeled initially by non-deterministic transitions. However, using gathered statistical information, one can more adequately model the system by considering probabilities of errors on each component, which can then be used to obtain quantifiable bounds on the adequate behaviors of the whole system.

In the 80s, **Vardi et al** suggested a method for qualitative verification of Markov chains, which was later extended to effective quantitative verification, which has become the staple in Temporal Probabilistic verification. Work in this field focuses mainly on probabilistic models like Markov chains,

Markov decision process and probabilistic automata, using primarily as a specification language a probabilistic adaptation of CTL/CTL*. The main focus of all these techniques is to reason with/model the probabilistic space generated by the evolutions of the system, that is probabilistic temporal reasoning [68, 26, 57, 102, 5, 7].

However, in recent years, there has been some work developed in the opposite side [69, 63, 11, 1]. Instead of looking at these widely used modeling concepts from the point of view of what are the probabilities of evolutions, one can also see them as dynamical systems of probability distributions. This perspective is closer to the usual approach in the theory of stochastic processes and it offers a complementary approach to the same concepts.

Herein, we will be addressing systems which are discrete in both time and state space. Systems with continuous time and continuous probabilistic distributions are best approached from the area of stochastic differential equations, and while one could use Interval Temporal Logic to address systems with continuous time and discrete state space, we chose not do so.

This Chapter is divided in the following sections: in the first section, we define the more common models used when describing systems with a probabilistic component. In Section 3.2, we present the logics that are commonly used to reason with probabilistic systems.

3.1 Probabilistic systems

The fundamental concept when studying probabilistic systems is that of a *stochastic process*. A stochastic process is a collection of random variables $\{X_t\}_{t\in T}$ over a probabilistic space $(\Omega, \mathcal{F}, \mu)$, with values in a measureable set S, indexed over a totally ordered set T. In our case, we are interested in stochastic processes taking values in a countable discrete set S, representing the states of the system, and taking the index set as $\mathbb{N}$.

Definition 3.1.1. Let $\{X_n\}_{n\in\mathbb{N}}$ be a stochastic process over $(\Omega, \mathcal{F}, \mu)$ taking values in a countable set S. The stochastic process is said to have the *Markov property* if:

$$P(X_{n+1} = s | X_1 = s_1, \dots X_n = s_n) = P(X_{n+1} = s | X_n = s_n),$$

whenever $P(X_1 = s_1 \dots X_n = s_n) > 0$.

The evolution of a stochastic process with the Markov property is completely driven by the present state, that is, the future (probabilistic) state of the system is given solely by a law depending only of the present state

of the system. However, it is necessary to impose an additional restriction, in order to guarantee that the evolution law is always the same throughout the index set.

Definition 3.1.2. Let $\{X_n\}_{n\in\mathbb{N}}$ be a stochastic process over $(\Omega, \mathcal{F}, \mu)$ taking values in a countable set S. The stochastic process is said to be *time homogeneous* if:

$$P(X_{n+1} = s | X_n = r) = P(X_n = s | X_{n-1} = r),$$

for all $n \geq 1$.

A Markov chain is then defined to be a stochastic process indexed over the natural numbers, taking values over a countable set, which is time homogeneous and has the Markov property. However, we present an alternative definition as a weighted transition system, with states labeled with propositional properties. We restrict our attention to rational-valued Markov chains. Many of the results presented in this work can be adapted to Markov chains with entries in $\mathbb{R}_{\mathsf{Alg}}$, but we chose to work with simpler models.

Definition 3.1.3. A *Markov chain* $\mathcal{M}$, (state) labeled over Λ is a tuple $\mathcal{M} = (S, M, d^0, L)$ such that:

- S is a countable set, named the *state space*,
- $M \in \mathbb{Q}^{\#S^2}$, where $M_{i,j} \geq 0$, and $\sum_j M_{i,j} = 1$, is a (row-)stochastic matrix, named the *transition matrix*,
- $d^0 \in \mathbb{Q}^{\#S}$ where $d^0_j \geq 0$, and $\sum_j d^0_j = 1$ is the *initial distribution*,
- $L : S \mapsto \{0,1\}^\Lambda$ is the *labeling function*.

It is also acceptable to define a Markov chain with the initial distribution focused on a distinguished single state, which is then named the *initial state*. In that case, we represent a Markov chain instead as $\mathcal{M} = (S, M, s^0, L)$. A Markov chain can also be represented as a graph, with vertexes labeled with valuations, and edges labeled with probabilities. Moreover the probabilistic distribution over states S in the instant n, represented as μ^n can be given by $\mu^{n+1} = \mu^n.M$ and $\mu^0 = d^0$, which then can be rewritten as $\mu^{n+1} = \mu^0.M^n$. In the probabilistic Model Checking community, a Markov chain is usually named a *probabilistic deterministic transition system*.

Example 3.1.1. In Figure 3.1.1, we represent a model of the IPv4 zeroconf protocol, which is used for establishing an unique IP address on a local home network, taken from [7]. In this scenario there are some household objects that have network access, and should communicate between each other; the network has to be "plug-and-play" and self configuring; one of the main issues is how each of the objects obtain an unique identifier in order to communicate. The protocol works as follows: each time a new object is to be connected to the network, it generates a new IP address randomly, over a fixed number of addresses (65024). However, if there are m objects already connected to the network, with probability $q = m/65204$, it will generate an already in use IP address. This must not happen. In order to avoid these problems, the host object sends a probe to n objects asking who is using the generated IP address. If an object receives a probe and the IP address generated is in use, it responds to the host, warning them of the possible collision. In this case, the host restarts the protocol. If however, the all the probes are lost, (and each has a probability of p of being lost), it may happen that the host believes no one is using the address; in this case the IP address collision occurs.

The Markov chain represented models this protocol. The protocol begins in state s_0, and randomly generates an IP address. If there is no collision (probability $1 - q$), it moves to the state s_4, where it will then use the new address in state s_6. However, if there is a collision, the host is then on state s_1; in this case, if the first probe is received informing of the colision (probability $1 - p$), the host restarts the protocol. If that does not happen, with probability p he will wait for the next probe, moving to state s_2; the case of state s_2 and s_3 is similar; however, in our represent we used 3 probes, and as such, if there are no warning of collision in state s_3, the host will obtain an used IP address, in state s_5 and s_7.

Example 3.1.2. Consider Figure 3.1.2 a description of a Markov chain inspired by the major political parties in Portugal. Portugal has two large political parties $\mathsf{PSD}, \mathsf{PS}$ which alternate across legislative elections, with CDS usually connoted with more right-wing positions, and BE and CDU associated with left-wing viewpoints. The initial distribution is obtained from the recent legislative elections by eliminating the results of the other political parties, and while the transition probabilities are essentially placeholders representing the percentage of voters that shift from one party to another, the example is nevertheless relevant to argue about two complementary aspects about Markov chains. The state space of the Markov chain

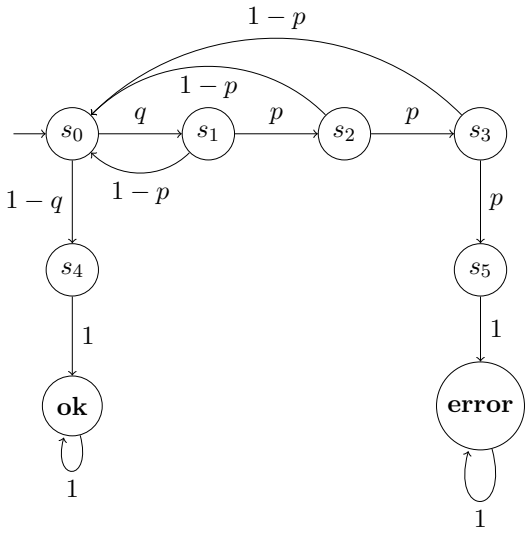


Figure 3.1: A Markov chain model for the IPv4 zeroconf protocol

is $\{\mathsf{CDS}, \mathsf{PSD}, \mathsf{PS}, \mathsf{CDU}, \mathsf{BE}\}$, and the initial distribution given as a vector is $(0.168, 0.392, 0.278, 0.102, 0.06)$.

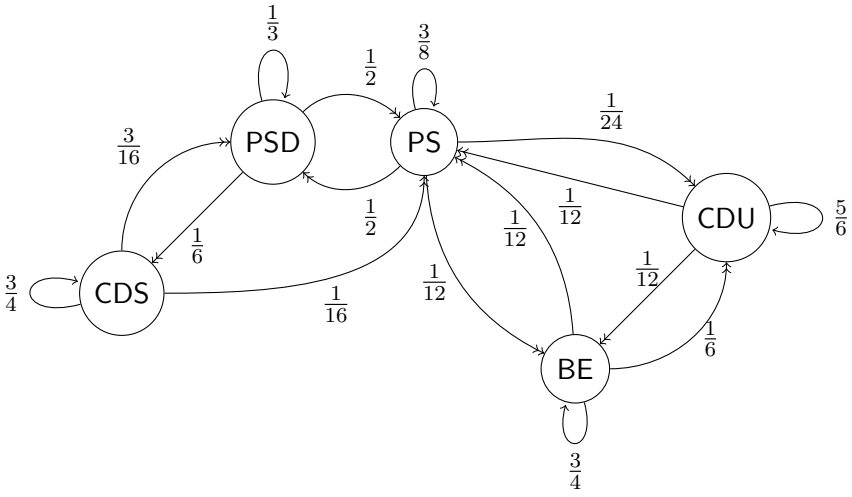


Figure 3.2: A Markov chain inspired by the major political parties in Portugal

A straightforward simulation of the evolution of the initial distribution shows that μ_i tends towards a distribution μ_F such that $\mu_F(i) > 0$ for any political party i. However, one can also see that for this Markov chain that the left-wing political party BE will never be the most voted party, and it will be even further away from an absolute majority of votes, no matter how many elections one considers. More rigorously, the formula $\int \mathsf{BE} \geq \frac{1}{2}$ never holds in any instant of time, that is $\mathsf{F} \int \mathsf{BE} \geq \frac{1}{2}$ is not satisfied in this scenario. In this scenario, we are arguing about (absolute) evolution of probabilities. The fact that BE has always a positive percentage of votes does not mean that eventually it will win the election, as in fact it will never will.

However, now consider the other possible viewpoint; consider evolutions across the transition system, that is, infinite sequences of political parties. Most (in fact almost all) evolutions will eventually reach the state BE, that is the formula $\mathsf{F}\ \mathsf{BE}$ represents a set of infinite sequences with probability 1; this probability does not change if we chose different but non-zero transition probabilities. In this case are making statements about the probability of evolutions. These two perspectives are complementary, and different problems will require different approaches.

The most common viewpoint in analyzing Markov chains in the Model Checking community is to analyze how much probability each temporal property has; the system is evolving, without non-determinism, without environmental circumstances unaccounted for in the probabilistic weights, and in each instant the system is in a definite state; in that state it may transition to other states with different probabilities, but it is always on one state at a time; however, two evolutions paths may have different probabilities. For instance, in Example 3.1.1, the finite path $s_0 s_1$ is assigned probability q, since the probability of starting in s_0 is 1, and the conditional probability of being in next time instant in s_1, given that one is in s_0 is q. The path $s_0 s_1 s_2 s_3 s_5 \mathbf{error}^\omega$ is assigned probability $q.p.p.p$, accordingly.

Definition 3.1.4. Let $\mathcal{M} = (S, M, s^0, L)$ be a Markov chain over a finite set S. Then $\mathcal{P}^{\mathcal{M}} = (S^\omega, \mathcal{F}^{\mathcal{M}}, \mu^{\mathcal{M}})$ is the *probability space over paths induced by the Markov chain* $\mathcal{M}$.

- The σ-algebra is the Borel σ-algebra generated from the cylinders $B^{\mathcal{M}}_\pi = \{\sigma \in S^\omega : \sigma|_{\#\pi} = \pi\}$, where $\pi \in S^*$;
- The probability measure $\mu^{\mathcal{M}}_{s^0}$ is defined, using Caratheodory's Extension as discussed in the semantics of EPPL, by assigning measure to

all the cylinders. If $\pi = s_{i_1} s_{i_2} \dots s_{i_k} \in S^*$, $\mu_{s^0}^{\mathcal{M}}(B_\pi^{\mathcal{M}}) = 0$ if $s_{i_1} \neq s^0$ and $\mu_{s^0}^{\mathcal{M}}(B_\pi^{\mathcal{M}}) = \prod_{j=1}^{k-1} M_{i_j, i_{j+1}}$.

Remark 3.1.1. If the Markov chain $\mathcal{M} = (S, M, d^0, L)$ is presented with an initial distribution d^0, then the probability measure $\mu_{d^0}^{\mathcal{M}}$ is defined with a slight modification. One consider the Markov chain $\mathcal{M}' = (S \cup \{s^0\}, M', s^0, L)$, where $M'_{s^0, s_i} = d^0(s_i)$ and $M'_{s_j, s^0} = 0$ and its associated probability measure $\mu_{s^0}^{\mathcal{M}'}$. Then $\mu_{d^0}^{\mathcal{M}}(A) = \mu_{s^0}^{\mathcal{M}'}(s^0.A)$.

With this probability measure over paths, we can then quantify the likelihood of temporal properties. The most commonly used logics in this context are probabilistic versions of CTL and CTL^*, where the absolute path quantifiers A/E are replaced with quantitative quantifiers $\mathsf{P}_{>r}$. The logics will be presented in Section 3.2.

Given a probability space induced by a Markov chain, one needs to understand which sets of paths are in fact events in the probability space. By construction, any cylinder is measureable. Now consider the set $A = \{\sigma \in S^\omega : \exists n \text{ s.t } \sigma(n) = s\}$, associated with the property $\mathsf{F}s$; this set is also measureable, as it is the (disjoint) infinite union of cylinders $\{B_{\pi.s}\}_{\pi \in (S - \{s\})^*}$; the set $B = \{\sigma \in S^\omega : \forall n \text{ s.t } \sigma(n) = s\}$ is also measureable, as it is the infinite intersection of cylinders $\{B_{\pi.s}\}_{\pi \in \{s\}^*}$. It was proven by **Landweber** [71] that any ω-regular language defines a measureable set of events.

Given that a ω-regular language is measureable, the only hard question remaining is whether we can compute its probability. The full construction is given in Section 3.2; herein, we focus on simple reachability properties, namely the computation of the probability of each state to satisfy the event $\mathsf{F}A$; let S be a set of states, and M a transition matrix; for each state $s \in S$, we wish to compute the probability of $P(s) = \mu_s^{\mathcal{M}}(\{\sigma \in S^\omega : \exists n \text{ s.t. } \sigma(n) \in A\}$.

This computation proceeds as follows [27]:

- Partition S as: $U_0 = \{s \in S : A \text{ is not reachable from } s\}$, $U_1 = \{s \in S - A : \text{ the measure of reaching a state in } A \text{ starting in } s \text{ is } 1\}$, and $U_? = S - (U_0 \cup U_1)$.
- Solve the following linear recursion:

$$P(s) = \begin{cases} 1, \text{ if } s \in U_1 \\ 0, \text{ if } s \in U_0 \\ \sum_{s' \in S} M_{s,s'}.P(s'), \text{ otherwise.} \end{cases}$$

- The recursion generates the following linear system, over variables $P(s), s \in U_?$, which has an unique solution [7]:

$$P(s) = \sum_{s' \in U_?} M_{s,s'}.P(s') + \sum_{s' \in U_1} M_{s,s'}. \tag{3.1}$$

Remark 3.1.2. The probabilistic propositional logic EPPL can be used to reason about the probability space over paths induced by a Markov chain, in the same way that propositional logic can be used to reason about paths in a transition system; this remark serves as an advanced example of EPPL and a gentle introduction to the more common (and powerful) logics used to reason with probabilities of evolutions. Suppose that we are given a Markov chain $\mathcal{M} = (S, M, s^0, L)$, where $S = \{s_1, \ldots, s_n\}$ is a finite set. Moreover, for the purpose of this remark suppose that the labeling function $L : S \mapsto \{0,1\}^S$ is such that $L(s)(s) = 1$ and $L(s)(r) = 0$, for all $s \neq r$.

Now we shall construct a EPPL model m that mimics $\mathcal{P}^M$. The propositional symbols that we shall consider are $\Lambda = S \times \mathbb{N}$. The sample set V is obtained in the following way: a valuation $v \in \{0,1\}^\Lambda$ is in V if for all $k \in \mathbb{N}$, there exists exactly one $i \in \{1, \ldots, n\}$ such that $v((s_i, k)) = 1$. A valuation $v \in V$ corresponds to an element of S^ω.

The probability measure ν is given by assigning measure to the cylinders, which by Caratheodory's Extension theorem assigns measure to the whole σ-algebra.

Instead of defining the probability measure in the cylinders B_w, for a generic finite valuation w, we instead just define the probability measure of valuations w whose domain is $S \times \{0, 1, \ldots, m\}$ for some $m \in \mathbb{N}$. The probability measure for the other finite valuations can be readily obtained by finite summation. Now:

- if there exists $k \in \mathbb{N}$ such that for some $i, j \in \{1, \ldots n\}, i \neq j, w((s_i, k)) = 1$ and $w((s_j, k)) = 1$, then $\nu(B_w) = 0$. This guarantees that we are only in a single state at any instant in time.

- in the case $dom(w) = S \times \{0\}$, then, if $w((s_0, 0)) = 1$ then $\nu(B_w) = 1$, else $\nu(B_w) = 0$.

- finally, if $dom(w) = S \times \{0, \ldots, m+1\}$, consider w' the finite valuation $dom(w') = S \times \{0, \ldots, m\}$ that can be extended to w, s_i the state such that $w'((s_i, m)) = 1$ and s_j the state such that $w((s_j, m+1)) = 1$. In this case, $\nu(B_w) = \nu(B_{w'}).M_{s_i,s_j}$.

Temporal bounded formulae can be rewritten as a propositional formulae:

- The simple temporal formula $s_1 \wedge \mathsf{X}\mathsf{X}s_3$ is rewritten as $(s_1, 0) \wedge (s_3, 2)$, which can be further rewritten as $(s_1, 0) \wedge (\bigvee_{j=1}^{n}(s_j, 1)) \wedge (s_3, 2)$ or even $\bigvee_{j=1}^{n}(s_1, 0) \wedge (s_j, 1) \wedge (s_3, 2)$; the probability of $\mu^{\mathcal{M}}(\{\sigma : S^{\omega} : \sigma \Vdash s_1 \wedge \mathsf{X}\mathsf{X}s_3\})$, that is the probability of the set of evolutions that satisfy $s_1 \wedge \mathsf{X}\mathsf{X}s_3$ can be computed by summing up the probability of each finite path $s_1 \wedge \mathsf{X}s_j \wedge \mathsf{X}\mathsf{X}s_3$ or equivalently the probability of all valuations that satisfy $\bigvee_{j=1}^{n}(s_1, 0) \wedge (s_j, 1) \wedge (s_3, 2)$.

- The temporal formula $\neg\mathsf{X}s_1$ can be rewritten as $\neg(s_1, 1)$ or $\bigvee_{k=2}^{n}(s_k, 1)$ or even $\bigvee_{j=1,k=2}^{n}(s_j, 0) \wedge (s_k, 1)$.

- A bounded until $s_1 \mathsf{U}^{\leq n} s_2$, that is, an until connective that has to be satisfied in at most n time steps, can also be rewritten as a propositional formula: $\bigvee_{i=0}^{n}(s_2, i) \wedge (\bigwedge_{j=0}^{i-1}(s_1, j))$.

- The temporal formula $(\mathsf{X}s_2) \wedge (\mathsf{X}s_3)$ which is unsatisfiable in this context. also can be translated to the propositional formula $(s_2, 1) \wedge (s_3, 1)$, which according to the above definition, does not belong to V, and its measure is 0.

- More complex bounded temporal formulae can also be translated into propositional formulae, however the translation is not as straightforward. One has to first discover the maximum path length that is relevant for the formula at hand, consider all the finite paths that satisfy the formula, translate each of them into a propositional formula representing a valuation, and finally consider the disjunction of all the propositional formulae obtained.

However, unbounded temporal formulae can not be represented as propositional formulae, as they express relations between an infinite number of paths. For instance $\mathsf{F}s_1$ is satisfiable by the length 1 path s_1, by the length 2 paths $s_j s_1$, for some $j \neq 1$, and so on. Therefore, one can only hope to translate $\mathsf{F}s_1$ to an infinitary version of propositional logic. In that case, one could rewrite $\mathsf{F}s_1$ as:

$$\bigvee_{k \in \mathbb{N}} \Big(\bigvee_{f:\{0,\ldots,k\} \mapsto S - \{s_1\}} \Big(\bigwedge_{i=0}^{k-1} (f(i), i) \Big) \wedge (s_1, k+1) \Big).$$

In this way, we can see linear temporal logic in this context as a way of allowing us to access more complex elements in the σ-algebra $\sigma(\{0,1\}^{\Lambda})$

than just finite unions and intersections of cylinders. Note however that $\nu(\{v \in V : V \Vdash \bigvee_{k\in\mathbb{N}}(\bigvee_{f:\{0,\dots k\}\mapsto S-\{s_1\}}(\bigwedge_{i=0}^{k}(f(i),i)) \wedge (s_1,k+1))\}$ is still equal to $\mu^{\mathcal{M}}(\{\sigma : \sigma \Vdash \mathsf{F}s_1\}$.

The Markov chain formalism does not have non-deterministic transitions; this can be an extremely significant restriction. Non-determinism is useful to model interactions with the environment, for instance reading input from a keyboard, and it is also required to model adequately distributed probabilistic algorithms; in the first case, one can try to model choice using a statistical approach, but statistical information might be unavailable; in the second case, it is of paramount importance to separate the probabilistic component of the process from the interleaving of the processes.

The non-deterministic model that we present is the Markov decision process, which is a direct generalization of Markov chains. Instead of transitioning between states using a fixed probabilistic evolution rule, one is offered a choice between actions; given the choice, a probabilistic evolution rule is chosen; the following example adapted from [7], depicted in Figure 3.1 is quite enlightening: a player is able to choose from two coins, one is a fair coin, and the other is a biased coin, with 3/4 probability of landing heads. The goal of the player is to obtain in two tosses two heads. Is the statement "the player will win the game with probability p" intuitively verifiable?

Intuition says that *a priori*, without knowing what coins will the player choose, it must be impossible to assign a probability; intuition also points out however, that the best choices, the choices that maximize his probability of winning, are to choose always the biased coin; in that case the probability should be $\frac{3}{4}\frac{3}{4} = \frac{9}{16}$; it also points out that if the player wishes to lose the game, he should pick always the fair coin, because then the probability of winning is then $\frac{1}{2}\frac{1}{2} = \frac{1}{4}$.

Definition 3.1.5 (Markov decision process)**.** A *Markov decision process* is a tuple $\mathfrak{M} = (S, \Sigma, T, s^0, L)$ such that:

- S is a countable set of *states*;
- Σ is a finite set of *actions*;
- $T : S \times \Sigma \times S \mapsto [0;1] \cap \mathbb{Q}$, named the *transition relation* is such that $\sum_{s'\in S} T(s,a,s') \in \{0,1\}$, for all actions a and states s; when $\sum_{s'\in S} T(s,a,s') = 1$, we say that action $a \in \Sigma$ is *enabled* in state s;
- The state s^0 is the *initial state*;

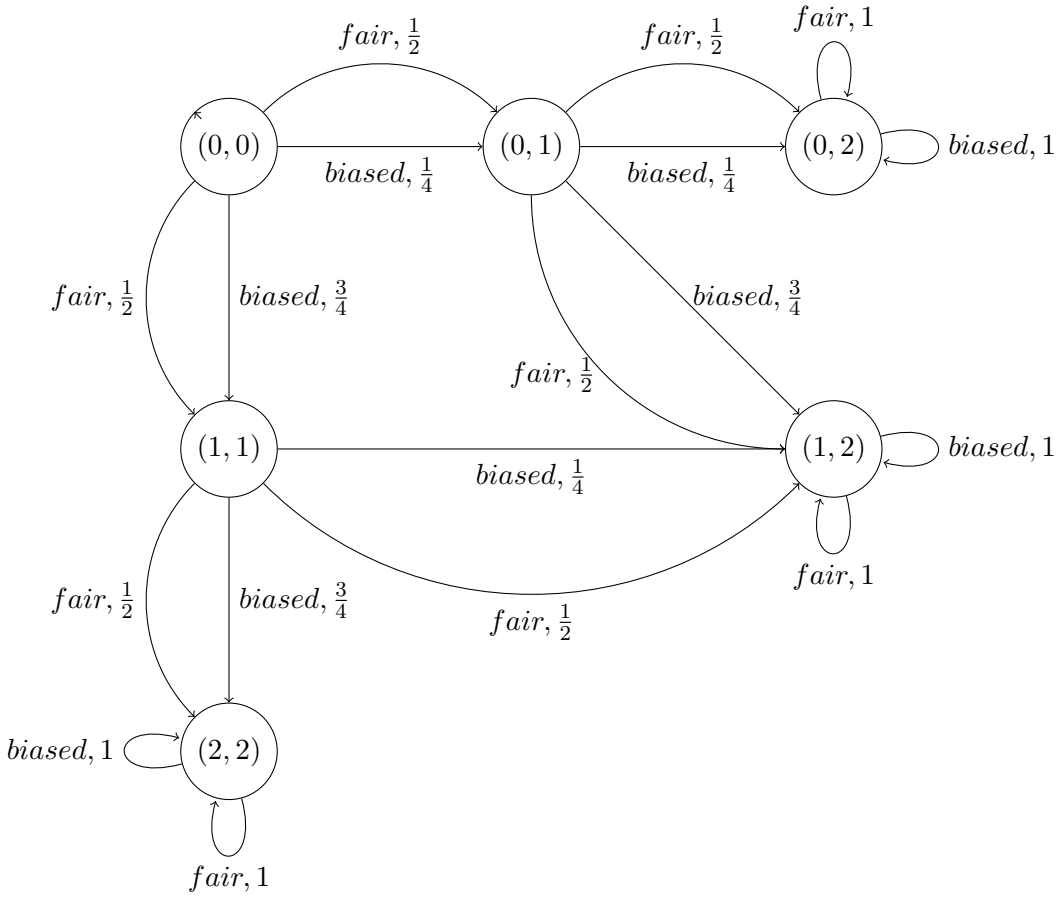


Figure 3.3: A Markov decision process representing two coin tosses of fair and biased coins. The labeling of the states represents the number of heads and the number of tosses, respectively

- $L : S \mapsto \{0,1\}^{\Lambda}$ is the *labeling function*.

A Markov chain is a special case of a Markov decision process, where Σ is a singleton, and there is always one action enabled. In order to be able to define probability of sets of paths, given a Markov decision process, we need some technique in order to eliminate non-determinism; the concept required is named the *scheduler*, although *adversary*, or *policy* also appear in the literature, depending on the intended applications of Markov decision processes. The *scheduler* has access to to current state where the Markov decision process is currently in, and also to the *history* of states passed

by; given that information, it chooses an action from Σ; given a scheduler, we can then define the probability space induced by the Markov decision process operating under it. Markov decision processes are usually named probabilistic nondeterministic transition systems (PNTS) in the verification community.

A *path* in a Markov decision process (S, Σ, T, s^0, L) is an element of $(S \times \Sigma)^\omega$ such that if $\sigma = (s_0, a_0)(s_1, a_1)\ldots(s_n, a_n)\ldots$, then action a_i is enabled on s_i . The definition for finite paths is similar. The set of all path in a Markov decision process is denoted by $Path^{\mathfrak{M}}$.

Definition 3.1.6 (Scheduler)**.** A scheduler for a Markov decision Process $\mathfrak{M} = (S, \Sigma, T, s^0, L)$ is a function $\mathfrak{S} : S^+ \mapsto \Sigma$ such that $\mathfrak{S}(s_0 s_1 \ldots s_n)$ is enabled on s_n. A path $\sigma = (s_0, a_0)(s_1, a_1)\ldots(s_n, a_n)\ldots$ is $\mathfrak{S}$-*path* if $a_i = \mathfrak{S}(s_0 \ldots s_i)$ for all $i \in \mathbb{N}$.

The definition is similar for finite paths. Given a scheduler, a path can be uniquely represented by the sequence of states, as $\sigma = (s_0, a_0)\ldots(s_n, a_n)\ldots$ may be rewritten as $\sigma = (s_0, \mathfrak{S}(s_0)\ldots(s_n, \mathfrak{S}(s_0 \ldots s_n))$; a scheduler induces an unique probability space on the σ-algebra over $\mathfrak{S}$-paths in $Path^{\mathfrak{M}}$ or equivalently over S^ω.

Definition 3.1.7. Let (S, Σ, T, s^0, L) be a Markov decision process, and $\mathfrak{S} : S^+ \mapsto \Sigma$ a scheduler for the Markov decision process $\mathfrak{M}$. Then, the *probability space induced by the scheduler* $\mathfrak{S}$, $\mathcal{P}^{\mathfrak{M}}_{\mathfrak{S}} = (S^{\mathfrak{M}}_{\mathfrak{S}}, \mathcal{F}, \mu^{\mathfrak{M}}_{\mathfrak{S}})$ is defined as follows:

- The sample set $S^{\mathfrak{M}}_{\mathfrak{S}} = S^\omega$;
- The σ-algebra is the Borel σ-algebra generated from the cylinders $B^{\mathfrak{M}}_\pi = \{\sigma \in S^{\mathfrak{M}}_{\mathfrak{S}} : \sigma|_{\#\pi} = \pi\}$, where $\pi \in S^*$;
- The probability measure $\mu^{\mathfrak{M}}_{\mathfrak{S}}$ is defined by assigning measure to the cylinders; if $\pi = (s_0, \mathfrak{S}(s_0))(s_1, \mathfrak{S}(s_0 s_1))\ldots(s_n, \mathfrak{S}(s_0 s_1 \ldots s_n))$ is a $\mathfrak{S}$-path, then:

$$\mu^{\mathfrak{M}}_{\mathfrak{S}}(B^{\mathfrak{M}}_{s_0 s_1 \ldots s_n}) = \prod_{i=0}^{n-1} T(s_i, \mathfrak{S}(s_0 \ldots s_i), s_{i+1}).$$

A scheduler also induces a Markov chain; however, given that the choices of the action at each step are dependent on the history of actions, one has to consider an infinite state Markov chain, where the states of the induced Markov chain are sequences of states of the Markov decision process:

Definition 3.1.8. Let $\mathfrak{M} = (S, \Sigma, s^0, L)$ be a Markov decision process and $\mathfrak{S} : S^* \mapsto \Sigma$ a scheduler for $\mathfrak{M}$. Then the *Markov chain induced by the scheduler* $\mathfrak{S}$, $\mathcal{M}_{\mathfrak{S}}^{\mathfrak{M}} = (s_0.S^*, T', s_0, L')$, where the transition matrix and the labeling function are as follows:

- the transition matrix is defined by $T'_{\pi,\pi.s} = T(\pi(\#(\pi)), \mathfrak{S}(\pi), s)$;
- the labeling function is defined by $L'(\pi.s) = L(s)$.

Given a ω-regular property φ, the analysis of Markov decision processes focuses on the discovering what are the extrema probabilities. Suppose that φ is a "good" property, one that we are interested that our system should have. In that case, if we prove that the scheduler that minimizes the probability of φ still assigns probability $1 - \epsilon$, for some small value of ϵ, we may be confident that our system does satisfy the property, **whatever** may happen. If φ represent an undesirable property, and if the scheduler that maximizes the probability of φ only assigns it probability ϵ, we are ensured that the system will behave adequately most of the time.

Definition 3.1.9. Let $\mathfrak{M} = (S, \Sigma, T, s^0, L)$ be a Markov decision process and φ a ω-regular property from the σ-algebra generated from the cylinders. Then, we define the *upper/lower probability*, denoted as $P_{\top}^{\mathfrak{M}}(\varphi)/P_{\bot}^{\mathfrak{M}}(\varphi)$ of φ as:

$$P_{\top}^{\mathfrak{M}}(\varphi) = sup_{\mathfrak{S}} \quad \mu_{\mathfrak{S}}^{\mathfrak{M}}(\varphi), \quad P_{\bot}^{\mathfrak{M}}(\varphi) = inf_{\mathfrak{S}} \quad \mu_{\mathfrak{S}}^{\mathfrak{M}}(\varphi).$$

The question that remains is how can these extrema probabilities be computed. Clearly the set of all schedulers is too large for a brute force approach; however, let us focus on reachability properties: suppose that there exists a subset of states A that we wish to reach; we are then concerned with the probability of the event $\mathsf{F}A$; as we know, the scheduler at each state has to choose an action, given the history; if we are interested in maximizing the probability of satisfying $\mathsf{F}A$, intuition says that the action one should choose is the action that maximizes this probability, since on one hand we only need to reach A, and so the path we have taken in order to reach A does not seem to be relevant. In fact, intuition is also correct here. There exists an important class of schedulers, named *memoryless schedulers*, which are enough for the purposes of reachability.

Definition 3.1.10. Let $\mathfrak{M} = (S, \Sigma, T, s^0, L)$ be a Markov decision process. A *memoryless scheduler* $\mathfrak{S}^m : S^* \mapsto \Sigma$ is a scheduler such that $\mathfrak{S}^m(s_0 \dots s_n) = \mathfrak{S}^m(s_n)$.

Theorem 3.1.1. *Let $\mathfrak{M} = (S, \Sigma, T, s^0, L)$ be a Markov decision process and $A \subseteq S$. Then the extrema probabilities are such that:*

$$P^{\mathfrak{M}}_{\top}(\mathsf{F}A) = sup_{\mathfrak{S}} \quad \mu^{\mathfrak{M}}_{\mathfrak{S}}(\mathsf{F}A) = max_{\mathfrak{S}^m} \quad \mu^{\mathfrak{M}}_{\mathfrak{S}}(\mathsf{F}A),$$
$$P^{\mathfrak{M}}_{\bot}(\mathsf{F}A) = inf_{\mathfrak{S}} \quad \mu^{\mathfrak{M}}_{\mathfrak{S}}(\mathsf{F}A) = max_{\mathfrak{S}^m} \quad \mu^{\mathfrak{M}}_{\mathfrak{S}}(\mathsf{F}A).$$

Note that the set of memoryless schedulers is finite. Moreover, the Markov chain induced by a memoryless scheduler is also finite, as it can be directly obtained by eliminating in each state of the Markov decision all the actions not prescribed by the memoryless scheduler. This means that by simply considering each memoryless scheduler one can compute $P^{\mathfrak{M}}_{\top}(\mathsf{F}A)/P^{\mathfrak{M}}_{\bot}(\mathsf{F}A)$, by the methods discussed for Markov chains; assuming that there are two enabled actions in each state, that still means an exponential search on the size of the Markov decision process. However, there exists a very used technique named *value iteration* which guarantees polynomial time complexity, instead of the testing of an exponential number of memoryless schedulers.

Proposition 3.1.1 ([27])**.** *Let $\mathfrak{M} = (S, \Sigma, T, s^0, L)$ be a Markov decision process, and $A \subseteq S$. For each state $s \in S$, consider the Markov decision process $\mathfrak{M}_s = (S, \Sigma, T, s, L)$. Then, both $P^{\mathfrak{M}_s}_{\top}(\mathsf{F}A)$ and $P^{\mathfrak{M}_s}_{\bot}(\mathsf{F}A)$ can be computed in polynomial time on the size of $\mathfrak{M}$.*

As in the case of Markov chains, we delay the presentation of more general ω-regular properties to Section 3.2. We leave a simple example that illustrates the fact that memoryless schedulers are not enough to compute the extrema probabilities of more general ω-regular properties.

Example 3.1.3. Consider the Markov decision process $\mathfrak{M}$ represented in Figure 3.1.3. The property one wishes to consider is $\varphi = (\mathsf{Fs} \wedge \mathsf{Ft})$. In this case, it is easy to see that $P^{\mathfrak{M}}_{\top}(\varphi) = sup_{\mathfrak{S}} \quad \mu^{\mathfrak{M}}_{\mathfrak{S}}(\varphi) = 1$, as the scheduler that alternates between action α and action β when the Markov decision process is on the state s_0 clearly achieves probability 1. However, any memoryless scheduler must choose on state s_0 an unique action. If it chooses action α/β, then the component Ft/Fs can only be satisfied with probability 0. Therefore, $sup_{\mathfrak{S}^m} \quad \mu^{\mathfrak{M}}_{\mathfrak{S}}(\varphi) = 0$.

3.2 Logics for Probabilistic systems

The main logics used for the analysis of the probabilistic systems presented herein (or common variants) are adaptations of CTL and CTL^*. There are

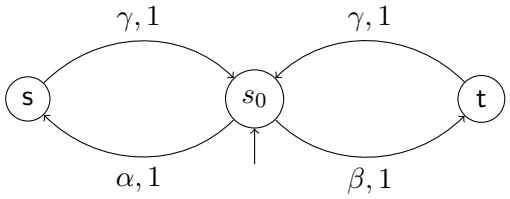


Figure 3.4: A Markov decision process where the extrema probabilities can not be computed by considering memoryless schedulers

multiple tools, widely used in practical problems, that are able to verify whether a probabilistic system verifies a certain property in a reasonable amount of time. The syntax for the logics is directly obtainable from replacing the absolute path quantifiers A/E by probabilistic path quantifiers $\mathsf{P}_{\gtreqless}$, with $\gtreqless \in \{>, \geq, <, \leq\}$; instead of asserting that all paths satisfy some property, we assert that the measure of the paths is $\gtreqless r$.

We present the syntax of the probabilistic adaptation of CTL and CTL*, named pCTL and pCTL*; the semantics of the pCTL are directly obtainable from the semantics of pCTL*.

Definition 3.2.1 (pCTL syntax)**.** Let Λ be a countable set of propositional symbols. The syntax of pCTLis given below in Backus-Naur form:

$$\psi ::= \top \quad | \quad p \quad | \quad \psi \wedge \psi \quad | \quad \neg\psi \quad | \quad \mathsf{P}_{\gtreqless r}(\Psi) \qquad (state formulae),$$
$$\Psi ::= (\mathsf{X}\psi) \quad | \quad (\psi \mathsf{U} \psi) \qquad (path formulae),$$

where $r \in \mathbb{Q}$, $p \in \Lambda$ and $\gtreqless \in \{>, \geq, \leq, <\}$. The set $\mathsf{pCTL}(\Lambda)$ is the set of state formulae over the propositional symbols Λ.

Definition 3.2.2 (pCTL* syntax)**.** Let Λ be a countable set of propositional symbols. The syntax of pCTL*is given below in Backus-Naur form:

$$\psi ::= \top \quad | \quad p \quad | \quad \psi \wedge \psi \quad | \quad \neg\psi \quad | \quad \mathsf{P}_{\gtreqless r}(\Psi) \qquad (state formulae),$$
$$\Psi ::= \psi \quad | \quad \Psi \wedge \Psi \quad | \quad \neg\Psi \quad | \quad (\mathsf{X}\Psi) \quad | \quad (\Psi \mathsf{U} \Psi) \qquad (path formulae),$$

where $r \in \mathbb{Q}$, $p \in \Lambda$ and $\gtreqless \in \{>, \geq, \leq, <\}$. The set $\mathsf{pCTL}^*(\Lambda)$ is the set of state formulae over the propositional symbols Λ.

The semantics for pCTL* given a Markov chain $\mathcal{M} = (S, M, s^0, L)$ over the propositional symbols Λ can then be given:

Definition 3.2.3. Let $\mathcal{M} = (S, M, s^0, L)$ be a Markov chain over the propositional symbols Λ, $\psi \in \mathsf{pCTL}^*(\Lambda)$ and $s \in S$. The (state/path) satisfaction relation between $\mathcal{M}$, s/σ and ψ/Ψ is defined inductively below:

- $\mathcal{M}, s \Vdash \top$;
- $\mathcal{M}, s \Vdash p$ iff $L(s)(p) = 1$;
- $\mathcal{M}, s \Vdash \psi_1 \wedge \psi_2$ iff $\mathcal{M}, s \Vdash \psi_1$ and $\mathcal{M}, s \Vdash \psi_2$;
- $\mathcal{M}, s \Vdash \neg\psi$ iff $\mathcal{M}, s \nVdash \psi$;
- $\mathcal{M}, s \Vdash \mathsf{P}_{\lessgtr r}(\Psi)$ iff $\mu_s^{\mathcal{M}}(\sigma \in S^\omega : \sigma \Vdash \Psi) \lessgtr r$, for $\lessgtr \in \{>, <, \geq, \leq\}$;
- $\mathcal{M}, \sigma \Vdash \psi$ iff $\mathcal{M}, \sigma(0) \Vdash \psi$;
- $\mathcal{M}, \sigma \Vdash \Psi_1 \wedge \Psi_2$ iff $\mathcal{M}, \sigma \Vdash \Psi_1$ and $\mathcal{M}, \sigma \Vdash \Psi_2$;
- $\mathcal{M}, \sigma \Vdash \neg\Psi$ iff $\mathcal{M}, \sigma \nVdash \Psi$;
- $\mathcal{M}, \sigma \Vdash \mathsf{X}\Psi$ iff $\mathcal{M}, \sigma[1] \Vdash \Psi$;
- $\mathcal{M}, \sigma \Vdash \Psi_1 \mathsf{U} \Psi_2$ iff there exists $j \geq 0$ such that $\mathcal{M}, \sigma[j] \Vdash \Psi_2$ and for all $0 \leq i < j$, $\mathcal{M}, \sigma[i] \Vdash \Psi_1$.

We now briefly address the model-checking problem for pCTL^* over Markov chains. In order to do so, we reduce the computation of the probability of an arbitrary ω-regular property in a Markov chain to the computation of the probability of a reachability property over a modified Markov chain; this problem is already tractable thanks to Equation 3.1. We require an important definition:

Definition 3.2.4 (Terminal Strongly Connected Component)**.** Let $\mathcal{M} = (S, M, s^0, L)$ be a Markov chain. A *terminal strongly connected component* C is a subset of S such that:

- $\forall s_1, s_2$ s.t. $M_{s_1,s_2} > 0$ if $s_1 \in C$ then $s_2 \in C$;
- $\forall s_1, s_2 \in C$ there exists a finite path $\pi = s_1 s_{i_1} \dots s_{i_k} s_2$, with $s_{i_j} \in C$, and all transitions can be made with non-zero probability.

Terminal strongly connected components (TSCC) generalize absorbing states, as once we are in a state of a terminal strongly connected component, one can not leave it; moreover, any two states are reachable. This means that any probability that enters a TSCC remains there and it is eventually homogenized, converging to a stationary distribution in the TSCC.

Lemma 3.2.1. *Let $\mathcal{M} = (S, M, s^0, L)$ be a Markov chain. Then for all $s \in S$:*

$$\mu_s^{\mathcal{M}}(\{\sigma \in S^\omega : \exists C \subseteq S \text{ a TSCC of } \mathcal{M} \text{ s.t. } \forall c \in C \text{ infinitely often } \sigma^i = c\}) = 1$$

The model-checking algorithm for pCTL^* over Markov chains is quite straightforward. First, one builds the parse tree for the pCTL^* formula δ given. We then enrich the labeling function $L : S \mapsto \{0,1\}^\Lambda$ with information about each state subformula $\delta_i \in sub(\delta)$. This is done by traversing the parse tree of δ by a bottom up approach. Afterwards, if the initial state is labeled with δ, the Markov chain satisfies δ, otherwise it does not, and one can extract a counterexample.

The only difficult case is how to deal with $\mathsf{P}_{\lessgtr r}(\Psi)$ subformulae, where Ψ is a path subformula. We have already seen how can one solve the issue when Ψ represents a reachability property; we now sketch how to deal with a generic path formula Ψ. We assume that the labeling function of the Markov chain is already extended to consider any state subformulae of Ψ.

The first step is to translate Ψ to a deterministic Rabin automaton $R = (Q, \Sigma, \tau, s^0, \alpha)$. This can be done, using Theorem 1.1.3. Afterwards, we need to compute the product Markov chain between the model Markov chain $\mathcal{M}$ and the Rabin automaton representing Ψ. Finally, we label each state in each terminal strongly connected component that satisfies the (lifted) accepting condition α of the Rabin automaton with *accept*. Then $\mu_s^{\mathcal{M}}(\sigma \in S^\omega : \sigma \Vdash \Psi) \lessgtr r$ if and only if $\mu_s^{\mathcal{M}\times\mathcal{R}}(\sigma \in (S \times Q)^\omega : \sigma \Vdash \mathsf{F}acc) \lessgtr r$.

Theorem 3.2.1 ([27, 7])**.** *Let $\mathcal{M} = (S, M, s^0, L)$ be a Markov chain and $\psi \in \mathsf{pCTL}^*$. The model-checking problem is decidable and the algorithm outlined above has polynomial time complexity in the size of $\mathcal{M}$ and double exponential complexity in the size of ψ.*

The semantics for the probabilistic logic pCTL^* can also be given with respect to Markov decision processes.

Definition 3.2.5. Let $\mathfrak{M} = (S, \Sigma, T, s^0, L)$ be a Markov decision process over the propositional symbols Λ, $\psi \in \mathsf{pCTL}^*(\Lambda)$ and $s \in S$, and $\mathfrak{S}^{all}$ the set of all schedulers for $\mathfrak{M}$. The (state/path) satisfaction relation between $\mathcal{M}$, s/σ and ψ/Ψ is defined inductively below:

- $\mathfrak{M}, s \Vdash \top$;
- $\mathfrak{M}, s \Vdash p$ iff $L(s)(p) = 1$;
- $\mathfrak{M}, s \Vdash \psi_1 \wedge \psi_2$ iff $\mathfrak{M}, s \Vdash \psi_1$ and $\mathcal{M}, s \Vdash \psi_2$;

- $\mathfrak{M}, s \Vdash \neg\psi$ iff $\mathcal{M}, s \nVdash \psi$;
- $\mathcal{M}, s \Vdash \mathsf{P}_{\lessgtr r}(\Psi)$ iff $\mu^{\mathfrak{M}}_{\mathfrak{S}}(\sigma \in S^{\omega} : \sigma \Vdash \Psi) \lessgtr r$ for all schedulers $\mathfrak{S} \in \mathfrak{S}^{all}$;
- $\mathcal{M}, \sigma \Vdash \psi$ iff $\mathcal{M}, \sigma(0) \Vdash \psi$;
- $\mathcal{M}, \sigma \Vdash \Psi_1 \wedge \Psi_2$ iff $\mathcal{M}, \sigma \Vdash \Psi_1$ and $\mathcal{M}, \sigma \Vdash \Psi_2$;
- $\mathcal{M}, \sigma \Vdash \neg\Psi$ iff $\mathcal{M}, \sigma \nVdash \Psi$;
- $\mathcal{M}, \sigma \Vdash \mathsf{X}\Psi$ iff $\mathcal{M}, \sigma[1] \Vdash \Psi$;
- $\mathcal{M}, \sigma \Vdash \Psi_1 \mathsf{U} \Psi_2$ iff there exists $j \geq 0$ such that $\mathcal{M}, \sigma[j]\Psi_2$ and for all $0 \leq i < j$, $\mathcal{M}, \sigma[i] \Vdash \Psi_1$.

In this case, the model-checking problem is also decidable, and proceeds in a way akin to the model-checking algorithm for Markov chains. We will not discuss it herein, as it will not be useful in our work. Interested readers can consult the article [27], or the detailed description in [7].

Both pCTL and pCTL* were designed as logics for verification, and it is unknown whether their satisfiability problem is decidable. The known results by **Brazdil et al.** [20] show that only the qualitative (all comparisons are made to threshold values $0, 1$) fragment of pCTL is decidable; it is also known that some formulae may only have infinite models, like for instance $\mathsf{P}_{>0}(\mathsf{G}(\neg p \wedge \mathsf{P}_{>0}(\mathsf{F}p)))$.

3.3 Related Probabilistic systems

In this section we discuss informally some other models for probabilistic temporal analysis. All of the models presented herein can be seen as Markov decision processes with additional requirements. One of the extensions are *partially observable Markov decision processes*; even though the concept can be seen as a generalization of a Markov decision process, we prefer to see it as restriction on the behavior of the scheduler; in partially observable Markov decision processes, the scheduler is unable to correctly know in which state the Markov decision process is really in. Instead, the choices of the scheduler can only be made with respect to an equivalence relation between the states of the Markov decision process.

This model intends to capture situations when after choosing an action α, the system can not ascertain directly in which state it end up. This scenario is common when the environment is partially defined in the labeling of the states, but the system has no direct knowledge of it. While the theory of

partially observable Markov decision processes is quite rich, most of the interesting problems are undecidable.

These undecidability results also apply unfortunately to most problems in a natural extension of non-deterministic automata: *probabilistic automata*. Probabilistic automata are tuples $(S, \Sigma, \{M_\sigma\}_{\sigma\in\Sigma}, s^0, \alpha)$, where S is a finite set of states, Σ is the alphabet, $\{M_\sigma\}_{\sigma\in\Sigma}$ is a collection of transition matrices, one for each letter of the alphabet, s^0 is the initial state, and α represents the accepting condition. In the case of probabilistic automata for languages over finite words, α is a subset of S, representing accepting states. For each $\lambda \in [0; 1]$, one defines the language accepted by the probabilistic automaton P by $L_\lambda(P) = \{\omega_1 \dots \omega_n \in \Sigma^* : (x_0)^T.M_{\omega_1}.\dots.M_{\omega_n}.(x_F)^T > \lambda\}$, where x_0 is the vector representing the initial state, with point mass in s^0, and x_F is a vector representing the accepting states, such that all entries are 0, except the ones corresponding with accepting states.

Probabilistic automata can be viewed as particular cases of partially observable Markov decision processes, where the equivalence relation between states is maximal; it can be proven that the problem of deciding whether $L^\lambda(P)$ is empty is undecidable [6, 86]. Note also that probabilistic automata can also be viewed as an extension of Markov chains where the alphabet is a singleton. We note that the decidability of the emptiness of $L^\lambda(P)$ in the case of Markov chains is unknown; in Chapter 7 we present our contributions on this problem.

We have already addressed to a great extent the uses of the probabilization of temporal logic in order to reason and validate probabilistic systems. However, there is also a complementary avenue to be explored. We can consider instead the temporalization of a probabilistic logic. This perspective allows for a complementary view of probabilistic systems; instead of focusing on the probability of evolutions, we reason about the evolution of probabilities. As mentioned in the beginning of this chapter, this view enables us to reason about the dynamical systems of probabilities induced by the probabilistic system.

The work in temporalization of a logic system has its roots on **Finger et al.** [40], and was recently applied to the probabilistic propositional logic EPPL in [9]. The temporalizations presented there have many interesting properties and are extended in Chapter 6; the temporalizations of EPPL considered both linear and branching time; so instead of reasoning over a linear sequence of propositional valuations, or a tree labeled with propositional valuations, one considers a sequence of probability distributions, or a tree labeled with probabilistic distributions. The logic EPLTL introduced in [9] is interpreted over sequences of probability distributions; as

we have seen a Markov chain induces a sequence of probability distributions and so EPLTL can be used to specify properties for further validation; we could also argue that a Markov decision process induces a tree of probability distributions and as such we could use EPCTL or EPLTL to reason about the induced tree. However, the undecidability of the emptiness problem for probabilistic automata implies that this avenue is greatly restricted as it was already noted in [8], see Theorem 4.3.7.

We now present the syntax and semantics of EPLTL.

Definition 3.3.1. The well formed formulae in EPLTL over the propositional symbols Λ are given below in Backus-Naur notation:

$$\begin{aligned} \beta &:= p \quad | \quad (\beta \wedge \beta) \quad | \quad (\neg\beta) \\ t &:= r \quad | \quad (t+t) \quad | \quad (t.t) \quad | \quad z \quad | \quad \int \beta \\ \delta &:= \underbrace{\sim \delta \quad | \quad \delta \cap \delta \quad | \quad \mathsf{X}\delta \quad | \quad \delta \mathsf{U} \delta}_{\text{temporal formulae}} \quad | \quad \underbrace{t \leq t}_{\text{comparison formulae}} \end{aligned}$$

where $p \in \Lambda$, $z \in Z$ and $r \in \mathbb{R}_{\mathsf{Alg}}$.

A EPLTL model is simply a sequence of EPPL models, much like an LTL model is a sequence of PL models. In this case we recall that a EPPL model is a probability space and an assignment from the set of algebraic variables Z to the real algebraic number field.

Definition 3.3.2. Let $\delta \in \mathsf{EPLTL}(\Lambda)$, $\pi : \mathbb{N} \mapsto \mathfrak{M}(\mathsf{EPPL}(\Lambda)) \times \mathbb{R}_{\mathsf{Alg}}^{Z}$ a EPLTL model. The satisfaction relation between π and δ is defined inductively on the structure of δ:

- $\pi \Vdash^{\mathsf{EPLTL}} t_1 \leq t_2$ iff $\pi(0) \Vdash^{\mathsf{EPPL}} t_1 \leq t_2$;
- $\pi \Vdash^{\mathsf{EPLTL}} (\sim \delta)$ iff $\pi \nVdash^{\mathsf{EPLTL}} \delta$;
- $\pi \Vdash^{\mathsf{EPLTL}} (\delta_1 \cap \delta_2)$ iff $\pi \Vdash^{\mathsf{EPLTL}} \delta_1$ and $\pi \Vdash^{\mathsf{EPLTL}} \delta_2$;
- $\pi \Vdash^{\mathsf{EPLTL}} (\mathsf{X}\delta_1)$ iff $\pi[1] \Vdash^{\mathsf{EPLTL}} \delta_1$;
- $\pi \Vdash^{\mathsf{EPLTL}} (\delta_1 \mathsf{U} \delta_2)$ iff there exists $j \in \mathbb{N}$ s.t. $\pi[j] \Vdash^{\mathsf{EPLTL}} \delta_2$ and for all $0 \leq k < j$, $\pi[k] \Vdash^{\mathsf{EPLTL}} \delta_1$.

We note that the algebraic variables have a purely local nature, that is $(x = 1) \cap (\mathsf{X}(x = 2))$ is a satisfiable formula. The logic has a finite model theorem, in the sense that the model π can be given as a finite sequence

followed by a loop. This means that any finite set of formulae will not be able to accurately describe a Markov chain, as a Markov chain can only be modeled as an infinite sequence of different distributions. The satisfiability procedure for EPLTL presented in [9] is quite straightforward; however we will present a generalization in Chapter 8. The model-checking problem when considering EPLTL models generated by Kripke structures labeled with EPPL models is also decidable; however when considering EPLTL models generated by Markov chains, the problem was never adequately researched. In the branching time logic EPCTL the results proven were similar, although model-checking EPCTL formulae against probabilistic automata was already known to be undecidable, see [8], in Remark 4.3.7.

Part II

Advances in Temporal and Probabilistic Reasoning

Chapter 4

Decidability and Complexity for ω-Regular Properties of Stochastic Systems

This chapter concerns an article published in the Logic Journal of IGPL in 2012 by David Henriques, myself, Paulo Mateus and Pedro Baltazar about the probabilization of (quantified) linear temporal logic. Our main goal on this work was threefold.

- obtain an example of the probabilization of a logic according to the Definition 2.1.8;
- compare the logic obtained to the unnested fragment of pCTL^* in order to understand if the satisfiability algorithm obtained from this probabilization can shed some light in the satisfiability problem for pCTL^*;
- extend the expressibility of the unnested fragment of pCTL^* which is the source of most used specifications; this extension is twofold. Unlike pCTL^*, we can reason with any algebraic operation between probabilities, and the temporal language used is able to express any ω-regular language unlike pCTL^*. For more information concerning $\mathsf{pCTL}/\mathsf{pCTL}^*$, see Section 3.2.

We recall that we will be dealing with the probability of evolutions described as linear temporal properties. The $\mathsf{pCTL}/\mathsf{pCTL}^*$ semantics are defined originally over Markov chains or probabilistic deterministic transition systems (PDTS); in order to extend this work to cope with Markov decision

processes or probabilistic non-deterministic transition systems (PNTS), one would need to adapt the probabilization operator basic concepts; instead of operations over elements of the real algebraic number field, one would have to consider algebraic operations over intervals, due to the fact that in Markov decision processes each scheduler assigns a probability to a temporal property.

We do not present the full article, as most of the preliminary definitions were already introduced. Furthermore, we will also replace any mention of the Appendix published with the paper by proper references in this work; we will also introduce additional comments in the last section in order to discuss further generalizations of the work.

4.1 Introduction

Temporal logics, such as CTL, LTL and CTL*, are widely used to reason about distributed and dynamic systems, with multiple applications to diverse fields such as software and hardware verification [16, 28, 43, 89], biological systems [13, 54], or even Philosophy [31, 53]. Despite their expressiveness, these logics are not suited for quantitative reasoning about common probabilistic systems, which has been an active research subject. Because of this, alternative semantics over Markov processes have been proposed and thoroughly studied. Logics like pLTL [30, 101, 104], pCTL or pCTL* [3, 27, 51, 58], have emerged and are nowadays commonly used for these purposes. These logics have been used more as a specification language than as a proper "logic", as they are mainly used to specify properties intended to be model-checked against some Markovian model.

The satisfiability problem for probabilistic logics was also studied, although with lesser scope. **Aziz et al** presented a probabilistic version of CTL* (PCTL*) [3] and mentioned the need for further study of its satisfiability problem, although not many results were presented thereafter; **Brázdil et al.** [20] solved the satisfiability problem for qualitative pCTL, while the satisfiability problem for full pCTL remains open. **Ognjanovic** [83] also proposed a probabilistic logic and presented a calculus for it, although with infinitary inference rules. We will discuss the related work in more detail in the conclusion of this work.

We introduce a logic to reason about probabilities of paths of a Markov chain, following the line of research on probabilistic temporal logic (pLTL) introduced in [101, 104]. We consider two enrichments to pLTL. First, we deal with probabilities of sets of paths that can be expressed in quantified

linear temporal logic (QLTL), see Definitions 1.1.7 and 1.1.8. This logic was introduced in [94] as an enrichment of LTL in order to capture the full expressibility of ω-regular languages. Secondly, we consider semialgebraic sets over probabilized QLTL formulae (i.e., the set of polynomial inequations formed by probabilistic QLTL expressions, algebraic real numbers and variables). We call this language *Probabilistic Quantified Linear Temporal Logic* (PQLTL).

We note that other probabilistic logics (pCTL, pCTL*) allow higher-order probabilities (events that are defined by probabilistic assertions). Our logic, while not allowing for this nesting of probability operators, is still rich enough to specify very relevant properties, such as probabilistic fairness or almost-sure termination but also many previously disregarded requests easily expressible in natural language.

For example, while requests of the form "with probability of 99%, the process eventually reaching the critical state implies that a flag will always be raised" can easily be expressed in pCTL, other reasonable requests such as "The probability of the process entering region 5 is at least double the probability of entering regions 1 to 4" or "The probability of reaching `Undecided` is at least 10% less than reaching either `Accept` or `Reject`" are just not expressible in this logic (or any other logic the authors are aware of). PQLTL is able to express this kind of assertions.

Moreover, the introduction of quantification over propositional symbols allows reasoning about events that only happen with some periodicity. This is a very useful feature when dealing with Markov chains. Indeed, the limit behavior of aperiodic chains is well known, but limit properties of periodic Markov chains are hard to study. By considering only transitions constant modulo the period of the chain, we can eventually reduce a periodic chain to a set of aperiodic subchains, which we know how to handle.

We derive algorithms for the decidability of PQLTL, and a complete Hilbert calculus. The algorithms are presented in two different versions: a weak SAT algorithm that just decides whether a certain formula is satisfiable, and a strong SAT algorithm that provides a model for a given satisfiable formula.

Concerning the SAT algorithm, the strong version lies in [n+2]-EXPSPACE whereas the weak version is in [n+1]-EXPSPACE, where n is the alternation depth of the quantifiers in the input formula. The weak SAT algorithm for the existential fragment, which has all the expressive power of PQLTL, is in EXPSPACE.

While PQLTL has theoretically interesting properties, the SAT algorithm proposed is very hard, which severely limits its application. How-

ever, if we restrict ourselves to *linear* inequalities without quantification, we still have relevant quantitative expressiveness while significantly reducing the requirements of the SAT algorithm. We call this more "practical" logic by pLTL^+. Indeed, we show that the SAT problem for pLTL^+ is PSPACE-complete, which may be surprising, since the SAT algorithm for simple non-probabilistic LTL already lies in this complexity class. Moreover, pCTL^*without nesting of the probability operator is a sublanguage of pLTL^+.

Capitalizing in the SAT algorithms, we derive a weakly complete calculus for PQLTL, and consequently for all the sublogics considered. We illustrate the calculus with a toy example using a PONG$^{\mathrm{TM}}$ game. We leave less academic applications of the calculus for future work.

We also present a model-checking algorithm, however the algorithm is straightforwardly obtained by applying well-know reduction of QLTL formulae to deterministic Rabin automata together with the SAT algorithm for the existential theory of the real numbers. This result is a simple generalization of an already existing automata-theoretical algorithm for pCTL^* [27], and so it will be postponed to the end of the Chapter.

The structure of the paper is the following; in Section 4.2 we introduce PQLTL, namely by providing its syntax and semantics. Both SAT algorithms for the logic are introduced in Section 4.3. In Section 4.4, we present the sublogic containing only linear inequalities for which we obtain a PSPACE SAT algorithm. The complete calculi for all the logics considered are developed in Section 4.5, where we also illustrate its use with a simple example. Finally, we draw some conclusions and future work.

4.2 Probabilization of quantified linear temporal logic

4.2.1 PQLTL Syntax

The construction of PQLTL is the following: a set of formulae is taken at a base level - *basic formulae* - and another set is built over it at an higher level - *global formulae*. A set of *probabilistic terms* is also considered.

Definition 4.2.1. The *well-formed formulae* in PQLTL over the proposi-

tional symbols Λ are presented below in Backus-Naur normal form.

$$\begin{array}{lll}
\gamma := p \mid (\neg\beta) \mid (\beta \Rightarrow \beta) \mid (\mathsf{X}\beta) \mid \beta\mathsf{U}\beta & & \\
\beta := \gamma \mid \neg\beta \mid \exists p.\beta & & \text{basic formulae} \\
t := z \mid 0 \mid 1 \mid \int\beta \mid (t+t) \mid (t \cdot t) & & \text{probabilistic terms} \\
\delta := (t \leq t) \mid (\sim \delta) \mid (\delta \supset \delta) & & \text{global formulae}
\end{array}$$

where $p \in \Lambda$ and $z \in Z$.

Basic formulae are simply QLTL formulae over a finite set Λ of propositional symbols, allowing for classical quantified temporal reasoning over them. The usual abbreviations for falsum $\bot$, disjunction $(\beta_1 \vee \beta_2)$, conjunction $(\beta_1 \wedge \beta_2)$, equivalence $(\beta_1 \leftrightarrow \beta_2)$ and universal quantification $\forall p.\beta$, as well as for **f**uture $(\mathsf{F}\beta)$ and **g**lobally $(\mathsf{G}\beta)$ are henceforth used freely.

Probabilistic terms permit quantitative reasoning over the set of algebraic real numbers by introducing a set of algebraic real variables Z which, together with addition, multiplication, 0, 1 and the equality relation of global formulae, allow the representation of any algebraic real number. *Measure terms*, terms of the form $(\int \beta)$ denote the probability of satisfying β.

Global formulae are built by taking comparison formulae $(t_1 \leq t_2)$ as *atoms* and building an analog of the propositional language over them. As in the basic case, we will assume the analogs of usual abbreviations for global falsum $\bot$, global disjunction $(\delta_1 \cup \delta_2)$, global conjunction $(\delta_1 \cap \delta_2)$ and global equivalence $(\delta_1 \Leftrightarrow \delta_2)$. The comparison operators $\{=, \neq, \geq, <, >\}$ will also be used as usual.

When no ambiguity arises, we shall drop the parenthesis.

4.2.2 PQLTL Semantics

In order to define the semantics for PQLTL, we need to build upon the semantics for QLTL. These semantics are detailed in the Definition 1.1.8 and follow the presentation by **Sistla et al** in [96].

The models of PQLTL are pairs, where the first element is a discrete time Markov chain with states labeled by valuations over Λ and the second element is an assignment over Z to the set of algebraic real numbers. Markov chains were defined in Definition 3.1.3.

As we have already discussed in Section 3.1, there is a uniquely induced probability measure for each PDTS $\mathcal{M}$ and state s over sets of paths that depart from s. The measure is defined over the sets of all paths departing from s with common prefixes (*cylinders*):

$$\mu_s^{\mathcal{M}}(\{\pi \in S^\omega : \pi|_k = ss_1...s_k\}) = M_{s,s_1} \times ... \times M_{s_{k-1},s_k},$$

. This is enough to fully define the measure [60]. Moreover, the same argument can be made to construct an uniquely induced probability measure for each PDTS over infinite sequences of valuations, using the labeling function L. This can be done by assigning measure ν to finite sequences of valuations (valuation cylinders); in this case:

$$\nu_s^{\mathcal{M}}(\{\omega \in (\{0,1\}^\Lambda)^\omega : \omega|_k = v...v_k\} = \mu_s^{\mathcal{M}}(\{\pi \in S^\omega : \pi|_k \in L^{-1}(v)...L^{-1}(v_k)\}),$$

which can be described as a finite union of path cylinders.

Given a model $(\mathcal{M} = (S, M, d^0, L), \rho : Z \to \mathbb{R}_{\mathsf{Alg}})$, the denotation of probabilistic terms is as follows:

- $[\![z]\!]_{\mathcal{M},\rho} = \rho(z)$; $[\![0]\!]_{\mathcal{M},\rho} = 0$; $[\![1]\!]_{\mathcal{M},\rho} = 1$;
- $[\![t_1 + t_2]\!]_{\mathcal{M},\rho} = [\![t_1]\!]_{\mathcal{M},\rho} + [\![t_2]\!]_{\mathcal{M},\rho}$; $[\![t_1.t_2]\!]_{\mathcal{M},\rho} = [\![t_1]\!]_{\mathcal{M},\rho}.[\![t_2]\!]_{\mathcal{M},\rho}$; and
- $[\![\int \beta]\!]_{\mathcal{M},\rho} = \sum_{s\in S} d^0(s)\mu_s^{\mathcal{M}}(\{\pi \in S^\omega : \pi(0) = s, L(\pi) \Vdash^{\mathsf{QLTL}} \beta\})$.

When the context is evident and no ambiguity arises, we shall drop the subscript $\mathcal{M}$. The measurability of the sets $\{\pi \in S^\omega : \pi(0) = s, L(\pi) \Vdash^{\mathsf{QLTL}} \beta\}$ is guaranteed due to a result by **Landweber** [71] proving that any ω-regular language is measureable in the Borel σ-algebra defined over its alphabet. In this case the set of models (infinite sequences of valuations) that satisfy β is ω-regular and therefore the set of paths is also measureable.

Since the denotation of terms of the form $\int \beta$ does not depend on the assignment ρ, we will drop it from the denotation in some statements. It will always be implied, in these cases, that the assertion is true for any assignment ρ. Moreover, the satisfaction of global formulae is given by:

- $\mathcal{M}, \rho \Vdash^{\mathsf{PQLTL}} (t_1 \leq t_2)$ iff $[\![t_1]\!]_{\mathcal{M},\rho} \leq [\![t_2]\!]_{\mathcal{M},\rho}$;
- $\mathcal{M}, \rho \Vdash^{\mathsf{PQLTL}} (\sim \delta)$ iff $\mathcal{M}, \rho \nVdash \delta$; and
- $\mathcal{M}, \rho \Vdash^{\mathsf{PQLTL}} (\delta_1 \supset \delta_2)$ iff $\mathcal{M}, \rho \Vdash \delta_2$ or $\mathcal{M}, \rho \nVdash \delta_1$.

Moreover, the notion of semantic entailment is introduced as usual: $\Gamma \models^{\mathsf{PQLTL}} \delta$ iff, for every model $\mathcal{M}$ and assignmentρ, $\mathcal{M}, \rho \Vdash^{\mathsf{PQLTL}} \delta$ whenever $\mathcal{M}, \rho \Vdash^{\mathsf{PQLTL}} \gamma$ for each $\gamma \in \Gamma$. Whenever it is clear from the context which logic we are interested in, we will drop the subscript.

Remark 4.2.1. Due to Proposition 1.1.4, we know that we can rewrite any QLTL formula in order to obtain a formula in existential prenex normal form. This fact motivates the natural fragment of PQLTL, PEQLTL, where

we directly assume that the measure terms are built using only formulae in existential prenex normal form. This syntactic fragment allows us to obtain much better complexity bounds in the presented algorithms; note however that since any QLTL formula is equivalent to a existential prenex normal formula, PEQLTL is as expressive as PQLTL.

Lemma 4.2.1. Let $\beta, \beta_1, ..., \beta_m \in \mathsf{QLTL}$. For any PDTS $\mathcal{M}$, we have that

$$[\![\int \beta]\!]_{\mathcal{M}} = [\![\int \bigvee_{i=0}^{2^m-1} (\beta \wedge (\bigwedge_{j=1}^{m} b_{ij}))]\!]_{\mathcal{M}} = \sum_{i=0}^{2^m-1} [\![\int (\beta \wedge (\bigwedge_{j=1}^{m} b_{ij}))]\!]_{\mathcal{M}} \quad (4.1)$$

where $b_{ij} = \begin{cases} \beta_j & \text{if the } j\text{-th bit of } i \text{ is 1,} \\ (\neg\beta_j) & \text{otherwise.} \end{cases}$

Proof. We will prove this lemma by induction on m.

If $m = 0$, then $[\![\int \beta]\!]_{\mathcal{M}} = [\![\int \bigvee_{i=0}^{1} (\beta \wedge (\bigwedge_{j=1}^{0} b_{ij}))]\!]_{\mathcal{M}} = [\![\int \beta \wedge \sim \bot]\!]_{\mathcal{M}} = [\![\int \beta]\!]_{\mathcal{M}}$.

Assume the lemma is true for m. Then:

$$\begin{aligned}
[\![\int \beta]\!]_{\mathcal{M}} &= [\![\int \bigvee_{i=0}^{2^m-1} (\beta \wedge (\bigwedge_{j=1}^{m} b_{ij}))]\!]_{\mathcal{M}} \\
&= [\![\int \bigvee_{i=0}^{2^m-1} (\beta \wedge (\bigwedge_{j=1}^{m} b_{ij}) \wedge (\beta_{m+1} \vee (\neg\beta_{m+1})))]\!]_{\mathcal{M}} \\
&= [\![\int \bigvee_{i=0}^{2^m-1} (\beta \wedge (\bigwedge_{j=1}^{m} b_{ij}) \wedge \beta_{m+1}) \vee \bigvee_{i=0}^{2^m-1} (\beta \wedge (\bigwedge_{j=1}^{m} b_{ij}) \wedge (\neg\beta_{m+1}))]\!]_{\mathcal{M}} \\
&= [\![\int \bigvee_{i=0}^{2^{m+1}-1} (\beta \wedge (\bigwedge_{j=1}^{m+1} b_{ij}))]\!]_{\mathcal{M}} \\
&= \sum_{i=0}^{2^{m+1}-1} [\![\int (\beta \wedge (\bigwedge_{j=1}^{m+1} b_{ij}))]\!]_{\mathcal{M}}.
\end{aligned}$$

where the last equality follows from the fact that the sets of paths (in $\mathcal{M}$) that satisfy $(\beta \wedge (\bigwedge_{j=1}^{m+1} b_{ij}))$ are disjoint, for any i, j, and as such we can use the additivity of the measure to obtain the desired result. □

Notice that the syntax and semantics of PQLTL strictly contain the logic pLTL, informally presented in [101, 104]. In addition, we allow quantitative reasoning, namely through the use of variables and comparisons

between terms. Readers familiar with probabilistic temporal logics may wonder about the lack of nesting of the probability operator like, for instance, in pCTL. It is the opinion of the authors that nesting of probabilities is unintuitive, and for the sake of simplicity, we have disregarded it in this work. Still, it must be noticed that this means that some assertions in PCTL cannot be expressed in PQLTL (and vice versa).

4.3 SAT algorithm for PQLTL

We now derive an algorithm for deciding the satisfiability problem for PQLTL. This algorithm, although computationally demanding for PQLTL formulae in general form, will be adapted to specific (but widely used) cases, resulting in comparatively efficient algorithms. We will develop two versions of the algorithm: one version will solve the classical SAT problem of determining if there is any model that satisfies the input formula; the other version will in fact provide a witness for the satisfiability. We will call these the *weak* and *strong* SAT problems, respectively.

Given a PQLTL global formula δ, let $gatm_\delta = \{a_1, ..., a_n\}$ be the subset of global atoms that occur in δ. Consider a countable set of propositional symbols (or local *atoms*) $\Xi = \{\xi_1, ...\xi_n, ...\}$ and an injective map λ that assigns a propositional symbol $\lambda(a_i) = \xi_i \in \Xi$ to each global atom in $gatm_\delta$; this function can be extended to map PQLTL formulae δ into propositional formulae λ_δ by simple structural induction. Let $V_{\lambda(gatm_\delta)}$ be the set of valuations over $\lambda(gatm_\delta)$, that we will identify with $\{0,1\}^n$ in the expected way. Clearly, we have that $\lambda(gatm_\delta)) = atm(\lambda_\delta)$, where $atm(\lambda_\delta)$ is the set of atoms in the classical propositional formula λ_δ.

We can use any classical SAT algorithm to check the satisfiability of λ_δ. If λ_δ is not satisfiable, then δ is also not satisfiable and we are done. Otherwise, let $mol(\lambda_\delta)$ (henceforth called the *molecules* of λ_δ) be the set of all $\Phi \subseteq atm(\lambda_\delta)$ such that the following propositional formula holds:

$$((\bigwedge_{\xi_i \in \Phi} \xi_i) \wedge (\bigwedge_{\xi_i \in atm(\lambda_\delta) \setminus \Phi} \neg \xi_i)) \Rightarrow \lambda_\delta. \tag{4.2}$$

Then, each formula λ_δ is equivalent to

$$\bigvee_{\Phi \in mol(\lambda_\delta)} ((\bigwedge_{\xi_i \in \Phi} \xi_i) \wedge (\bigwedge_{\xi_i \in atm(\lambda_\delta) \setminus \Phi} \neg \xi_i)). \tag{4.3}$$

Which is just a syntactic characterization of the disjunctive normal form.

To prove satisfiability of δ, in addition to λ_δ being satisfiable, we must have that

$$\bigcup_{\Phi \in mol(\lambda_\delta)} ((\bigcap_{\xi_i \in \Phi} \lambda^{-1}(\xi_i)) \cap (\bigcap_{\xi_i \in atm(\lambda_\delta) \setminus \Phi} \sim \lambda^{-1}(\xi_i))). \tag{4.4}$$

must also be satisfiable. For each molecule $\Phi_k \in mol(\lambda_\delta)$, we will denote the expression

$$((\bigcap_{\xi_i \in \Phi_k} \lambda^{-1}(\xi_i)) \cap (\bigcap_{\xi_i \in atm(\lambda_\delta) \setminus \Phi_k} \sim \lambda^{-1}(\xi_i))). \tag{4.5}$$

by mol_k, and abusively call it the k-th molecule of δ. It is also possible to consider mol_v, where $v \in V_{\lambda(gatm_\delta)}$:

$$((\bigcap_{\{\xi_i : v(\xi_i)=1\}} \lambda^{-1}(\xi_i)) \cap (\bigcap_{\{\xi_i : v(\xi_i)=0\}} \sim \lambda^{-1}(\xi_i))). \tag{4.6}$$

Lemma 4.3.1. The PQLTL formula δ is satisfiable iff there exists $v \in V_{\lambda(gatm_\delta)}$ such that $v(\lambda_\delta) = 1$ and mol_v is satisfiable.

Proof. ($\Rightarrow$) Suppose δ is satisfiable. Let $(\mathcal{M}, \rho)$ be a model of δ and recall $gatm_\delta = \{a_1, .., a_n\}$. Consider

$$mol = \bigwedge_{i=1}^{n} \alpha_i \qquad \text{where} \qquad \alpha_i = \begin{cases} a_i & \mathcal{M}, \rho \Vdash a_i, \\ (\sim a_i) & \text{otherwise.} \end{cases} \tag{4.7}$$

Then, obviously $\mathcal{M}, \rho \Vdash mol$. Furthermore, this mol is associated with $v \in V_{\lambda(gatm_\delta)}$ s.t. $v(a_i) = 1$ iff $\mathcal{M}, \rho \Vdash a_i$, which is a witness for λ_δ.

($\Leftarrow$) Suppose mol_v is satisfiable and $v(\lambda_\delta) = 1$. Let $(\mathcal{M}, \rho)$ be a model of mol_v. Then $\mathcal{M}, \rho \Vdash a_i$ iff $v(\lambda(a_i)) = 1$, by induction in the structure of δ, $\mathcal{M}, \rho \Vdash \delta$:

- if $\delta = a \in gatm(\delta)$, then $v(\lambda_\delta) = v(\lambda(a)) = 1$ iff $\mathcal{M}, \rho \Vdash a$, that is $\mathcal{M}, \rho \Vdash \delta$;
- if $\delta = \sim \delta_1$, then $v(\lambda_\delta) = v(\lambda(\sim \delta_1)) = v(\neg\lambda(\delta_1)) = 1 - v(\lambda(\delta_1)) = 1$ iff $v(\lambda(\delta_1)) = 0$ iff, by IH, $\mathcal{M}, \rho \nVdash \delta_1$ iff $\mathcal{M}, \rho \Vdash \sim \delta_1$, that is $\mathcal{M}, \rho \Vdash \delta$;
- if $\delta = \delta_1 \supset \delta_2$, then $v(\lambda_\delta) = v(\lambda(\delta_1 \supset \delta_2)) = v(\lambda(\delta_1) \Rightarrow \lambda(\delta_2)) = max[1 - v(\lambda(\delta_1)), \lambda(\delta_2)] = 1$ iff $v(\lambda(\delta_1)) = 0$ or $v(\lambda(\delta_1)) = 1$ iff, by IH, $\mathcal{M}, \rho \nVdash \delta_1$ or $\mathcal{M}, \rho \Vdash \delta_2$ iff $\mathcal{M}, \rho \Vdash \delta_1 \supset \delta_2$, that is $\mathcal{M}, \rho \Vdash \delta$.

□

We will now propose a SAT algorithm for a molecule mol_k. If, for all k, this algorithm returns no model, then δ is not satisfiable. If it does return a model for some k, then that model also satisfies δ.

Before fixing a mol_k, consider $\Theta(\delta)$, a set of QLTL formulae, defined as $\beta \in \Theta(\delta)$ iff $\int \beta$ is subterm of δ). Since $\Theta(mol_k) = \Theta(\delta)$ for all k, so we will denote this set just by Θ. For the remaining, we fix an enumeration of Θ, $\beta_1, ..., \beta_l$, and denote the set $\{0,1\}^l$ by V_Θ.

Fix now a mol_k. We will consider, for each $\beta_i \in \Theta$, the disjunction of its "molecules" in Θ:

$$\beta_i \equiv \bigvee_{\substack{\sigma \in V_\Theta : \\ \sigma(\beta_i) = 1}} (\beta_i \wedge (\bigwedge_{\substack{\sigma(\beta_j) = 1, \\ j \neq i}} \beta_j \quad \wedge \bigwedge_{\substack{\sigma(\beta_j) = 0, \\ j \neq i}} \neg\beta_j)). \tag{4.8}$$

Where $\sigma(\beta_i) = 1$ if the i-th bit of σ is 1, $\sigma(\beta_i) = 0$ otherwise. Let us now assign, to each $\sigma \in V_\Theta$, an algebraic real variable $x_{\boldsymbol{\sigma}} \notin Z$ indexed by a binary integer $\boldsymbol{\sigma}$ identified with σ in the obvious way. Intuitively, this variable will take on the value of the denotation of its corresponding molecule of formulae of Θ in the output PDTS $\mathcal{M}$:

$$x_{\boldsymbol{\sigma}} = x_{\boldsymbol{\sigma}_1,...,\boldsymbol{\sigma}_l} = [\![\int \bigwedge_{\boldsymbol{\sigma}_j=1} \beta_j \wedge \bigwedge_{\boldsymbol{\sigma}_j=0} \neg\beta_j]\!]_{\mathcal{M}}. \tag{4.9}$$

Where $\boldsymbol{\sigma}_i$ represents the i-th bit of $\boldsymbol{\sigma}$. Therefore, by Lemma 4.2.1,

$$[\![\int \beta_i]\!]_{\mathcal{M}} = [\![\int \bigvee_{\substack{\sigma \in V_\Theta : \\ \sigma(\beta_i) = 1}} (\beta_i \wedge (\bigwedge_{\substack{\sigma(\beta_j) = 1, \\ j \neq i}} \beta_j \quad \wedge \bigwedge_{\substack{\sigma(\beta_j) = 0, \\ j \neq i}} \neg\beta_j))]\!]_{\mathcal{M}} = \tag{4.10}$$

$$= \sum_{\substack{\sigma \in V_\Theta) : \\ \sigma(\beta_i) = 1}} [\![\int (\beta_i \wedge (\bigwedge_{\substack{\sigma(\beta_j) = 1, \\ j \neq i}} \beta_j \quad \wedge \bigwedge_{\substack{\sigma(\beta_j) = 0, \\ j \neq i}} \neg\beta_j))]\!]_{\mathcal{M}} = \sum_{\substack{\sigma \in V_\Theta : \\ \boldsymbol{\sigma}_i = 1}} x_{\boldsymbol{\sigma}}.$$

Each atom in mol_k can now be written as an inequality between polynomials in $x_{\boldsymbol{\sigma}}$ and Z variables, taking $\neg(t_1 \leq t_2)$ as $(t_1 > t_2)$ and the other obvious syntactic shortcuts. In addition, we must have that $x_{\boldsymbol{\sigma}} \geq 0$ and $\sum_{\boldsymbol{\sigma}} x_{\boldsymbol{\sigma}} = 1$. Therefore, any model satisfying mol_k must also satisfy the following system of inequations:

$$\begin{cases} \alpha_{1k} \\ \alpha_{2k} \\ \dots \\ \alpha_{nk} \\ \sum_{i=0}^{2^l} x_i = 1 \\ x_i \geq 0 \end{cases} \quad \text{where } \alpha_{ik} \text{ is the } i\text{-th literal of } \mu_k \ . \tag{4.11}$$

Before we solve the system, we must also include one additional restriction in non-feasible $x_{\boldsymbol{\sigma}}$, that is, $x_{\boldsymbol{\sigma}}$ whose associated QLTL molecule

$$\bigwedge_{\boldsymbol{\sigma}_j=1} \beta_j \wedge \bigwedge_{\boldsymbol{\sigma}_j=0} \neg\beta_j \tag{4.12}$$

is not satisfiable. For each such variable, we add "$x_{\boldsymbol{\sigma}} = 0$" to the system.

By using the result mentioned in Remark 2.1.2 for the existential fragment of the first-order theory of real ordered fields, we can check whether the system has solutions. If this system has at least one solution, then we can construct the witness for the SAT Algorithm for δ, that we describe next.

Consider the individual QLTL models $\mathcal{M}_{\boldsymbol{\sigma}}$ associated with each $x_{\boldsymbol{\sigma}} \neq 0$, regarding them as a Markov chains with transitions of probability 1. Consider the disjoint union of all these models, and starting distribution on the initial state of each model $\mathcal{M}_{\boldsymbol{\sigma}}$ with probability $p_{\boldsymbol{\sigma}} = x_{\boldsymbol{\sigma}}$; then $\mathcal{M}$ is a PDTS. Consider also the assignment ρ that maps each real algebraic variable in δ to its solution in the previous system. And so, $(\mathcal{M}, \rho)$ witness the satisfiability of δ.

The procedure is summarized in Algorithm 2.

Theorem 4.3.1. Algorithm 2 is correct, i.e., if Algorithm 2 returns a model, then that model witnesses the satisfiability of δ.

Proof. If Algorithm 2 returns a model, then each $\mathcal{M}_{\boldsymbol{\sigma}}$ satisfies $\phi_{\boldsymbol{\sigma}}$ corresponding *only* to that $\boldsymbol{\sigma}$, because $\phi_{\boldsymbol{\sigma}}$ contains at least one conjunct that is the negation of the respective conjunct in $\phi_{\boldsymbol{\sigma'}}, \boldsymbol{\sigma} \neq \boldsymbol{\sigma'}$. Then, on $\mathcal{M}_{\boldsymbol{\sigma'}}$ seen as a Markov chain with transitions of probability 1, the measure of the paths that satisfy $\phi_{\boldsymbol{\sigma}}$ is 1 if $\boldsymbol{\sigma'} = \boldsymbol{\sigma}$ (there is only one path and it satisfies $\phi_{\boldsymbol{\sigma}}$), and 0 otherwise (there is only one path and it *does not* satisfy $\phi_{\boldsymbol{\sigma}}$). Therefore, $[\![\int \phi_{\boldsymbol{\sigma}}]\!]_{\mathcal{M}} = \sum_{s_i \in S} d_0(s_i) \cdot \mu^{\mathcal{M}}_{s_i}(\{\pi : \pi \Vdash^{\mathsf{QLTL}} \int \phi_{\boldsymbol{\sigma}}, \pi(0) = s_i\}) = \sum_{\sigma' \in V} d_0(s_0^{\boldsymbol{\sigma'}}) \cdot \mu^{\mathcal{M}}_{s_0^{\boldsymbol{\sigma'}}}(\{\pi : \pi \Vdash^{\mathsf{QLTL}} \int \phi_{\boldsymbol{\sigma}}, \pi(0) = s_0^{\boldsymbol{\sigma'}}\}) = d_0(s_0^{\boldsymbol{\sigma}}) \cdot \mu^{\mathcal{M}}_{s_0^{\boldsymbol{\sigma}}}(\{\pi :$

Algorithm 2: StrongSATPQLTL(δ)

Input: PQLTL formula δ
Output: PQLTL model $(\mathcal{M} = (S, M, d^0, L), \rho)$ or *no model*

compute $\lambda_\delta, atm(\lambda_\delta)$;
foreach $v \in V_{\lambda(gatm_\delta)}$ such that $v(\lambda_\delta) = 1$ **do**
  compute mol_v, Θ;
  $$\kappa \longleftarrow \begin{cases} \alpha_{1v} \\ \alpha_{2v} \\ ... \\ \alpha_{nv} \\ \sum_{\{\sigma \in V_\Theta\}} x_\sigma = 1 \\ x_\sigma \geq 0 \end{cases} ;$$
  foreach $\int \beta_i \in \kappa$ **do**
    $\int \beta_i \longleftarrow \sum_{\{\sigma \in V_\Theta : \sigma(\beta_i)=1\}} x_\sigma$;
  end
  foreach x_σ **do**
    $\phi_\sigma \longleftarrow \bigwedge_{\sigma(i)=1} \beta_i \wedge \bigwedge_{\sigma(i)=0} (\neg \beta_i)$;
    $\mathcal{M}_\sigma \longleftarrow StrongQLTLSat(\phi_\sigma)$;
    if $\mathcal{M}_\sigma = no\ model$ **then**
      $\kappa \longleftarrow \kappa \cap x_\sigma = 0$;
    end
  end
  $s \longleftarrow \exists StrongRealSat(\kappa)$;
  if $s = no\ solution$ **then**
    goto next mol_v in line 2;
  end
  $\mathcal{M} = \bigcup_{\sigma \in V_\Theta} \mathcal{M}_\sigma$;
  $S = S_\mathcal{M}$; $d^0 = \{x_{\sigma_1}, ..., x_{\sigma_{2^l+1}}\}$ over $\{s_0^{\mathcal{M}_{\sigma_1}}, ..., s_0^{\mathcal{M}_{\sigma_{2^l+1}}}\}$ for $x_\sigma \neq 0$;
  $L = L_\mathcal{M}$; $M_{s_1,s_2} = 1$ iff $s_1 \rightarrow s_2$ in the respective $\mathcal{M}_\sigma$;
  return $m = (S, M, d_0, L)$ and $\rho(x) = \kappa(x)$ for $x \in Z$;
end
return *no model*;

$\pi \Vdash^{\mathsf{QLTL}} \int \phi_{\boldsymbol{\sigma}}, \pi(0) = s_0^{\boldsymbol{\sigma}}\}) + \sum_{\boldsymbol{\sigma'} \neq \boldsymbol{\sigma}} d_0(s_0^{\boldsymbol{\sigma'}}) \cdot \mu^{\mathcal{M}}_{s_0^{\boldsymbol{\sigma'}}}(\{\pi : \pi \Vdash^{\mathsf{QLTL}} \int \phi_{\boldsymbol{\sigma'}}, \pi(0) = s_0^{\boldsymbol{\sigma'}}\}) = d_0(s_0^{\boldsymbol{\sigma}}) \cdot 1 + \sum_{\boldsymbol{\sigma'} \neq \boldsymbol{\sigma}} d_0(s_0^{\boldsymbol{\sigma'}}) \cdot 0 = x_{\boldsymbol{\sigma}}$.

We also have that β_i is satisfied if $\bigvee_{\{\boldsymbol{\sigma} \in V_{p(\mu)} : \boldsymbol{\sigma}(\beta_i) = 1\}} \phi_{\boldsymbol{\sigma}}$ is satisfied since each of the paths that satisfy β_i satisfy also one (and only one) $\phi_{\boldsymbol{\sigma}}$ such that $\boldsymbol{\sigma}(\beta_i) = 1$. On the other hand, since each of these paths satisfy only one such $\phi_{\boldsymbol{\sigma}}$, we have $[\![\int \bigvee_{\{\boldsymbol{\sigma} \in V_\Theta : \boldsymbol{\sigma}(\beta_i) = 1\}} \phi_{\boldsymbol{\sigma}}]\!]_{\mathcal{M}} = \sum_{\{\boldsymbol{\sigma} \in V_\Theta : \boldsymbol{\sigma}(\beta_i) = 1\}} [\![\int \phi_{\boldsymbol{\sigma}}]\!]_{\mathcal{M}} = \sum_{\{\boldsymbol{\sigma} \in V_\Theta : \boldsymbol{\sigma}(\beta_i) = 1\}} x_{\boldsymbol{\sigma}}$.

Since these values and the values assigned to each $\rho(x)$ are exactly the solutions of the system, each inequality in some mol_v (the one for which output is produced) must hold for this model. But satisfaction of each inequality means satisfaction of mol_v, which means δ is satisfiable by Lemma 4.3.1. □

Lemma 4.3.2. If mol_v is satisfiable, then the **foreach** command at line 2 must exit in line 2.

Proof. All steps of the algorithm are computations of total functions, therefore, the cycle quits. If the cycle quits, it must exit either in line 2 or in line 2. Suppose that it exits in line 2. Then the system κ has no solution.

On the other hand, if mol_v is satisfiable, there must be a model $\mathcal{M}$ and assignment ρ such that $\mathcal{M}, \rho \Vdash \alpha_{iv}$, where α_{iv} is the i-th literal of mol_v. For this model, the denotation of each measure term must take on a value such that the polynomial equation defined by each α_{iv} is satisfied.

By Lemma 4.2.1,

$$[\![\int \beta]\!]_{\mathcal{M},\rho} = \sum_{i=0}^{2^m - 1} [\![\int (\beta \wedge (\bigwedge_{j=1}^{l} b_{ij}))]\!]_{\mathcal{M},\rho}$$

for the basic subformulae $\beta_1, ..., \beta_l$ of Θ.

By the same lemma,

$$1 = [\![\int \top]\!]_{\mathcal{M},\rho} = \sum_{i=0}^{2^m - 1} [\![\int (\top \wedge (\bigwedge_{j=1}^{l} b_{ij}))]\!]_{\mathcal{M},\rho} = [\![\int (\bigwedge_{j=1}^{l} b_{i,j})]\!]_{\mathcal{M},\rho}.$$

From the semantics of PQLTL, $[\![\int \gamma]\!]_{\mathcal{M}',\rho} \geq 0$ for any basic (QLTL) formula γ, model $(\mathcal{M}', \rho)$.

Finally, if any QLTL formula γ is not satisfied by some path, then $[\![\int \gamma]\!]_{\mathcal{M}',\rho} = 0$ regardless of the model $(\mathcal{M}', \rho)$.

Consider now the values for algebraic variables in Z assigned by ρ and $x_{\boldsymbol{\sigma}} = [\![\bigwedge_{j=1}^{l} b_{ij}]\!]_{\mathcal{M},\rho}$ where $b_{ij} = \beta_j$ if the i-th bit of $\boldsymbol{\sigma}$ is 1, $b_{ij} = (\neg \beta_j)$

otherwise. It is clear from the previous points that these $x_{\boldsymbol{\sigma}}$ witness the consistency of the system κ, which is a contradiction.

Therefore the cycle must exit at line 2. □

Theorem 4.3.2. *Algorithm 2 is adequate, i.e., if Algorithm 2 returns* no model*, then formula δ is not satisfiable.*

Proof. Suppose δ is satisfiable. Then, by Lemma 4.3.1, there is mol_v satisfiable, s.t. $v \in V_{\lambda(gatm_\delta)}$ and $v(\lambda_\delta) = 1$. Algorithm 2 will enter the **foreach** at line 2 at least once with such mol_v. By Lemma 4.3.2, Algorithm 2 must exit at line 2 and therefore, cannot return *no model*. □

Theorem 4.3.3. Algorithm 2 decides the strong satisfiability problem for PQLTL in [n+2]-EXPSPACE, where n is the alternation depth of the quantifiers in the input.

Proof. By Theorem 4.3.1 and Theorem 4.3.2, 2 clearly decides the satisfiability problem for PQLTL.

Storage of $atm(\lambda_\delta)$ in line 2 requires $\mathcal{O}(|\delta|)$ space.

After each iteration of the cycle in line 2, if the algorithm has not finished, we can discard all computations done inside the cycle before starting over, so the number of iterations does not influence the space requirements of the algorithm.

Computation of mol_v and Θ can be done in $\mathcal{O}(|\delta|)$. Storing system κ, however, requires storing a variable for each $\sigma \in V_\Theta$, which takes $\mathcal{O}(2^{|\Theta|}) = \mathcal{O}(2^{|\delta|})$.

There is a polynomial number of β_i, so the substitution in line 2 does not increase space constraints.

The cycle of line 2 runs $\mathcal{O}(2^{|\delta|})$ times, each time either storing $\mathfrak{M}_\sigma$ or adding "$x_\sigma = 0$" to the system. ϕ can be computed in $O(|\delta|)$ and adding "$x_\sigma = 0$" takes constant space. However, *StrongQLTLSat* is in [n+1]-EXPSPACE (Theorem 1.1.5), and, we actually need the [n+2]-EXPSPACE algorithm since the negation of a formula with n alternations of quantifiers is a formula with $n+1$ alternations, and we are also considering $\neg\beta$ formulae. Therefore, each $\mathfrak{M}_\sigma$ takes [n+2]-EXPSPACE space in $|\delta|$. There are $\mathcal{O}(2^{|\delta|})$ variables, each requiring saving at most one $\mathcal{M}_\sigma$ this means that this cycle uses space of the order $\mathcal{O}(2^{|\delta|})\mathcal{O}((\text{n+2})\text{-exp}\{p(|\delta|)\}) = \mathcal{O}((\text{n+2})\text{-exp}\{p(|\delta|)\})$. Notice the size of the system remains $\mathcal{O}(2^{|\delta|})$.

In Remark 2.1.2), it is mentioned that the strong algorithm for the satisfaction in the real algebraic number theory is in EXPSPACE. Since we

actually need a solution for the system, this means line 2 takes $O(2^{2^{|\delta|}})$ space.

Line 2 adds no complexity, as it is merely a union of previously stored objects and line 2 just collects one subset of already existent structures (x_σ's and s_0's).

Therefore, the space complexity of the algorithm is

$$\mathcal{O}(|\delta|) + \mathcal{O}(2^{2^{|\delta|}}) + \mathcal{O}((\mathsf{n{+}2})\text{-}\mathsf{exp}\{p(|\delta|)\}) = \mathcal{O}((\mathsf{n{+}2})\text{-}\mathsf{exp}\{p(|\delta|)\}),$$

which is [n+2]-EXPSPACE. □

Corollary 4.3.1. *Algorithm 2 decides the strong satisfiability problem for PEQLTL in 2-EXPSPACE.*

Algorithm 2 can easily be adapted to solve the *weak* version of the SAT problem. The impact of relaxing the requirements is twofold: in line 2, we need only to consider the weak version of the SAT algorithm for QLTL, which is n-EXPSPACE (in our case, since we consider terms of the form $(\neg\beta)$, we introduce an extra alternation, leaving us in [n+1]-EXPSPACE). Furthermore, we do not have to save the witnesses, only an indicator of the satisfiability of each ϕ_σ. Therefore, the cycle of line 2 reduces to [n+1]-EXPSPACE complexity. The other consequence of considering the weak SAT is that in line 2, where we need only to check the *consistency* of system κ. By Remark 2.1.2, this procedure can be done in PSPACE over the size of the system, which is EXPSPACE. Therefore, line 2 needs only EXPSPACE resources. This is of particular importance for the PEQLTL weak SAT problem, since for $n \geq 1$, the QLTL SAT already dominates the 2-EXPSPACE complexity of solving system κ, but for $n = 0$ (PEQLTL), the complexity of the whole algorithm would be dominated by line 2. We refrain from explicitly writing this algorithm for the weak SAT problem since it is very similar to Algorithm 2. Instead, we present the following modified algorithm:

Theorem 4.3.4. Algorithm 3 decides the weak satisfiability problem for PQLTL in [n+1]-EXPSPACE, where n is the alternation depth of the quantifiers in the input.

Corollary 4.3.2. *Algorithm 3 decides the weak satisfiability problem for PEQLTL in EXPSPACE.*

Algorithm 3: WeakSATPQLTL(δ)

Input: PQLTL formula δ
Output: δ *satisfiable* or *no model*

Consider Algorithm 2 and:
replace line 2 with $y \longleftarrow WeakQLTLSat(\phi_\sigma)$;
replace line 2 with **if** $y = no\ model$;
replace line 2 with $s \longleftarrow \exists WeakRealSat(\kappa)$;
delete lines 2, 2 and 2;
replace line 2 with **return** δ *satisfiable*.

4.4 Restriction to non-quantified linear inequalities - pLTL$^+$

We now consider the case of non-quantified LTL. The SAT algorithm for a single LTL formula in a probabilistic setting is essentially the same as the SAT for the non-probabilistic version. However, when we consider quantitative reasoning as introduced in the syntax of PQLTL, the procedure becomes much less obvious, since different measure terms can share propositional symbols. Theorem 4.3.3 proves the decidability of the SAT problem for PQLTL, which is strictly more complicated than this problem, but the algorithm presented hinges on the fact that an exponential number of variables (in $|\delta|$) is used to represent the probability of each conceivable configuration of QLTL statements or their negations in the original formula δ. This representation can be much simplified. In this section, we show that considering non-quantified LTL formulae and a simple syntactic restriction, we can adapt Algorithm 2 to only consider a number of these variables linear on the size of the original formula, assuming all other to be 0. This will allow a significant reduction on the space requirements of the algorithm.

4.4.1 Syntactic restriction

A critical requirement for the new algorithm is that we must have a system of linear inequalities instead of a polynomial one. This can be done by weakening PQLTL, removing (besides the quantification, obviously) multiplication at the term level and leaving only sum as an algebraic connective. We denote this weaker logic by pLTL$^+$. Its syntax is given below, were $p \in \Lambda$ and $c \in \mathbb{Q}$:

$$\begin{array}{rl} \beta := p \mid (\neg\beta) \mid (\beta \Rightarrow \beta) \mid \mathsf{X}\beta \mid \beta\mathsf{U}\beta & \text{basic formulae} \\ t := z \mid c \mid \int\beta \mid (t+t) \mid (c.t) & \text{probabilistic terms} \\ \delta := (t \leq t) \mid (\sim\delta) \mid (\delta \supset \delta) & \text{global formulae} \end{array}$$

The semantics for pLTL^+ is the restriction of the semantics of PQLTL obtained by removing the denotation of multiplication. Notice that pLTL^+ is still very expressive and that most of the natural language assertions concerning probabilities (of traces) are expressible within it. For instance pLTL logic introduced in [101, 104] is contained in pLTL^+, since we are able to compare probabilites of formulae against 0 or 1. The most interesting feature of pLTL^+ is that satisfiability can be checked within PSPACE, as we show next.

4.4.2 SAT algorithm for pLTL^+

We now present the (weak) SAT algorithm for pLTL^+. It is an adaptation of Algorithm 3 that removes the need to consider all $x_{\boldsymbol{\sigma}}$ variables at once, thus reducing the space complexity. Notice that the strong version of the SAT problem for this problem is in EXPSPACE, since the strong SAT for LTL reduces to it and therefore, in this case, the previous algorithms can be used without increasing the required complexity of the procedure. Algorithm 4 describes the procedure.

Theorem 4.4.1. If Algorithm4 returns δ *satisfiable*, then δ is indeed satisfiable.

Proof. The proof of correction for this algorithm mimics the proof of correction of Algorithm 2 with the obvious changes in the domain of the sums and removing the construction of the witnesses. □

Naturally, there is the concern that by considering only subsets of $x_{\boldsymbol{\sigma}}$ of size at most $|\delta|+1$, we might leave out models where the formula would be satisfied. We claim this is not the case by showing that the existence of such models implies the existence of at least one suitably sized ($|\{x_{\boldsymbol{\sigma}} \neq 0\}| \leq |\delta|+1$) model that is found by Algorithm 4.

Theorem 4.4.2. Algorithm 4 is adequate, i.e., if Algorithm 4 returns *no model*, then formula δ is not satisfiable.

Algorithm 4: Sat $\mathsf{pLTL}^+(\delta)$

Input: pLTL^+ formula δ
Output: δ *satisfiable* or *no model*

compute $\lambda_\delta, atm(\lambda_\delta)$;
foreach $v \in V_{\lambda(gatm_\delta)}$ such that $v(\lambda_\delta) = 1$ **do**
  compute mol_v, Θ;
  $\kappa \longleftarrow \begin{cases} \alpha_{1v} \\ \alpha_{2v} \\ ... \\ \alpha_{nv} \end{cases}$;
  foreach $V \subseteq V_\Theta$ s. t. $0 < |V| \leq |\delta| + 1$ **do**
    foreach $\int \beta_i \in \kappa$ **do**
      $\int \beta_i \longleftarrow \sum_{\{\sigma \in V : \sigma(p_i)=1\}} x_\sigma$;
    end
    $\kappa \longleftarrow \kappa \cap \begin{cases} \sum_{\sigma \in V} x_\sigma = 1 \\ x_\sigma \geq 0 \text{ for } \sigma \in V \end{cases}$;
    $s \longleftarrow LinearSolve(\kappa)$;
    if $s =$ *no solution* **then**
      break;
    end
    foreach $x_\sigma \neq 0$ **do**
      $\phi_\sigma \longleftarrow \bigwedge_{\sigma(i)=1} \beta_i \wedge \bigwedge_{\sigma(i)=0} (\neg \beta_i)$;
      $y \longleftarrow WeakLTLSat(\phi_\sigma)$;
      if $y =$ *no model* **then**
        goto next V in line 4;
      end
    end
    return δ *satisfiable*;
  end
end
return *no model*;

Proof. Suppose δ is a pLTL^+ satisfiable formula; then Algorithm 4 applied to δ returns a PDTS $\mathcal{M}$ and assignment ρ such that $\mathcal{M}, \rho \Vdash \delta$. Consider now the system stored in memory when Algorithm 4 is at line 2, just before exiting. Remove the 2^l conditions $x_{\boldsymbol{\sigma}} \geq 0$, all conditions of the form $x_{\boldsymbol{\sigma}} = 0$ and all the corresponding $x_{\boldsymbol{\sigma}}$. This transformed system has $|gatm(\delta)| + 1$ inequations and these inequations are linear, since δ is a pLTL^+ formula. Furthermore, the system has a nonnegative solution (the set of values of $x_{\boldsymbol{\sigma}}$ that would be outputted). Then, from linear programming [25], there is a solution for the system with at most $|gatm(\delta)| + 1$ variables taking values different from 0. Consider this solution and construct $\mathcal{M}'$ from $\mathcal{M}$ reassigning probabilities according to the new values of $x_{\boldsymbol{\sigma}}$ and discarding unneeded $\mathcal{M}_{\boldsymbol{\sigma}}$. It is clear that $\mathcal{M}', \rho \Vdash \delta$, since the denotations of terms yield the same values.

Furthermore, this particular set of $x_{\boldsymbol{\sigma}}$ (different form 0) has size at most $|gatm(\delta)| + 1$, so it will eventually be considered in the cycle in line 4 of Algorithm 4. At least in this iteration of the cycle, the algorithm must exit at line 4 and therefore, cannot return *no model*. □

Theorem 4.4.3. Algorithm 4 decides the weak satisfiability problem for pLTL^+ in PSPACE.

Proof. By Theorem 4.4.1 and Theorem 4.4.2, Algorithm 4 clearly decides the satisfiability problem for pLTL^+.

Storage of $atm(\lambda_\delta)$ in line 2 requires $\mathcal{O}(|\delta|)$ space.

After each iteration of the cycle in line 2, if the algorithm has not finished, we can discard all computations done inside the cycle before starting over, so the number of iterations does not influence the space requirements of the algorithm.

Computation of mol_v and Θ can be done in $\mathcal{O}(|\delta|)$. Storing system κ, requires storing a variable for each of $|atms(\delta)| + 1$ $\sigma \in V_\Theta$, which also takes $\mathcal{O}(|\delta|)$.

There is a polynomial number of β_i, so the substitution in line 4 leaves the algorithm still in $\mathcal{O}(p_1(|\delta|))$, for some polynomial p_1.

Line 4 requires an additional $\mathcal{O}(|\delta|)$ space. Notice the size of the system remains $\mathcal{O}(p_2(|\delta|))$, for some polynomial p_2.

It is well known from linear algebra that the algorithm for the satisfaction of systems of *linear* inequations is in P, which means that at line 4, the algorithm is still in $\mathcal{O}(p_3(|\delta|))$ space, for some polynomial p_3. Notice that considering only linear systems avoided once again an increase in the complexity of the algorithm.

The cycle of line 4 runs $\mathcal{O}(|\delta|)$ times, but it does not need to save any information at each iterations, so the space complexity does not increase with each iteration. Moreover, ϕ_σ can be computed in $\mathcal{O}(|\delta|)$, and $WeakLTLSat$ is in PSPACEby Theorem 1.1.1. These means that this cycle uses space of the order $p_4(\mathcal{O}(p_5(|\delta|))) = \mathcal{O}(p_6(|\delta|))$, for polynomials p_4, p_5 and p_6.

Therefore, the space complexity of the algorithm is $\mathcal{O}(|\delta|) + \mathcal{O}(p_3(|\delta|)) + p_4(\mathcal{O}(p_5(|\delta|)))$ which is PSPACE. □

Theorem 4.4.4. The satisfiability problem for pLTL^+ is PSPACE-complete.

Proof. Algorithm 4 shows the satisfiability problem for pLTL^+ can be solved in PSPACE. LTLSat is PSPACE complete [95]. Given a LTL formula β, we can obviously build the formula $(\int \beta > 0)$, from β in constant time. If there is a model for β, at least that model seen as a Markov chain satisfies $(\int \beta > 0)$ and an algorithm for the pLTL^+ SAT returns a model. If β is not satisfiable, then no path satisfies β and, in particular, all probability measures over paths that satisfy β in any model (the empty set) yield 0 and no model can be returned by a pLTL^+ SAT. □

4.5 Complete Hilbert calculus

We now turn our attention towards using the previous algorithms to obtain a complete Hilbert calculus for PQLTL and its reducts. The proof technique closely follows those in [24, 38, 77].

We present the axiomatization in Table 4.1. We consider a Hilbert system - recursive set of axioms and finitary rules. We recall the axiom schema **ROF** is decidable thanks to Tarski's result on the decidability of real ordered fields and for the decidability of QLTL, one can reference [45]. Thus, the axioms in Table 4.1 constitute a recursive set.

Theorem 4.5.1. The calculus presented on Table 4.1 is sound, that is $\vdash_{\mathsf{PQLTL}} \delta$ implies $\models_{\mathsf{PQLTL}} \delta$.

Proof. Proof of correction is straightforward and will not be detailed; Axioms **GTaut** and **Prob** follow trivially from the definition of the semantics, the correctness of **ROF** comes from Tarski's result; **FAdd** and **Mon** are a consequence, respectively of the finite additivity and monotonicity of probability measures. The **MP** rule follows from the definition of the semantics of $\supset$. □

Axioms

[**GTaut**]	$\vdash_{\mathsf{PQLTL}}$	δ	for each PQLTL instantiation δ of a tautological propositional formula;
[**Prob**]	$\vdash_{\mathsf{PQLTL}}$	$(\int \varphi = 1)$	for each QLTL valid formula φ;
[**ROF**]	$\vdash_{\mathsf{PQLTL}}$	$(t_1 \leq t_2)$	for each instantiation of a valid analytical inequality;
[**FAdd**]	$\vdash_{\mathsf{PQLTL}}$	$((\int(\neg(\beta_1 \wedge \beta_2)) = 1) \supset (\int(\beta_1 \vee \beta_2) = \int \beta_1 + \int \beta_2))$;	
[**Mon**]	$\vdash_{\mathsf{PQLTL}}$	$((\int(\beta_1 \Rightarrow \beta_2) = 1) \supset (\int \beta_1 \leq \int \beta_2))$;	

Inference rules

[**MP**] $\delta_1, (\delta_1 \supset \delta_2) \vdash_{\mathsf{PQLTL}} \delta_2$.

Table 4.1: HC_{PQLTL} : complete calculus for PQLTL.

The proof of completeness will follow by the usual contrapositive approach: If $\nvdash^{\mathsf{PQLTL}} \delta$ then $\not\models^{\mathsf{PQLTL}} \delta$. Then, there is a PDTS $\mathcal{M}$ and an assignment ρ such that $\mathcal{M}, \rho \not\Vdash^{\mathsf{PQLTL}} \delta$. A formula δ is said *PQLTL-consistent* if $\nvdash^{\mathsf{PQLTL}} (\sim \delta)$. We must first show that the consistency of a global formula is propagated to the consistency of at least one of its molecules.

Lemma 4.5.1. Let δ be a PQLTL-consistent formula. Then there is a molecule $\mathbf{\Phi}_k \in mol(\delta)$ such that mol_k is consistent.

Proof. Suppose, by contradiction, that for each $\mathbf{\Phi}_k \in mol(\delta)$, $(\sim mol_k)$ is a theorem, then, by tautological reasoning (**GTaut**),

$$\bigcap_{\mathbf{\Phi}_k \in mol(\delta)} (\sim mol_k) = \bigcap_{\mathbf{\Phi} \in mol(\delta)} (\sim ((\bigcap_{\varphi \in \Phi} \varphi) \cap (\bigcap_{\varphi \in atm(\delta) \setminus \Phi} (\sim \varphi)))). \quad (4.13)$$

is also a theorem. Therefore,

$$\sim (\bigcup_{\mathbf{\Phi} \in mol(\delta)} ((\bigcap_{\varphi \in \Phi} \varphi) \cap (\bigcap_{\varphi \in atm(\delta) \setminus \Phi} (\sim \varphi)))). \quad (4.14)$$

is also a theorem. And so $\sim \delta$ is also a theorem and δ is not PQLTL-consistent, which is a contradiction. □

Theorem 4.5.2. The calculus presented on Table 4.1 is complete, that is $\models^{\mathsf{PQLTL}} \delta$ implies $\vdash^{\mathsf{PQLTL}} \delta$.

Proof. We will prove that every PQLTL-consistent formula has a model. This will suffice, since $\nvdash^{\mathsf{PQLTL}} \delta$ implies $\nvdash^{\mathsf{PQLTL}} (\sim (\sim \delta))$, that is, $(\sim \delta)$ is PQLTL-consistent and so $(\sim \delta)$ is satisfiable which means $\not\models^{\mathsf{PQLTL}} \delta$.

Suppose then that γ is a PQLTL-consistent formula. By Lemma 4.5.1 there is a molecule of γ, $mol_\gamma = (\bigcap_{i\in I}(t \leq t')_i) \cap (\bigcap_{j\in J}(t \leq t')_j)$ that is PQLTL-consistent.

Assume, by contradiction, that the SAT Algorithm 2 returns *no model* for μ_γ. If the SAT algorithm returns *no model* for mol_γ it has to be for one of the following two reasons: (i) it can not find a v at line 2; (ii) for all viable v the SatReal algorithm returns no model at line 2. We will now show that for both cases we can contradict the consistency of μ_γ.

In case (i), λ_{mol_γ} (a propositional formula) is not satisfied by any valuation, i.e. $(\neg\lambda_{mol_\gamma})$ is a valid formula and, by completeness of the propositional calculus, a theorem. Therefore, by **GTaut**, $\vdash^{\mathsf{PQLTL}} (\sim mol_\gamma)$.

In case (ii), using **Prob**, **FAdd** and **Mon**, and considering Θ the set of subformulae of mol_γ as in Subsection 4.3 we can rewrite each $\int \beta_j \in mol_\gamma$ as

$$\sum_{\{\sigma\in\{0,1\}^{|\Theta|}:\sigma_j=1\}} \int (\beta_j \wedge (\bigwedge_{\sigma_i=1} \beta_i \wedge \bigwedge_{\sigma_i=0} (\neg\beta_i))). \tag{4.15}$$

We will refer to this rewritten form as μ^*_γ. The same axioms also allow us to derive the theorems

$$\sum_{\sigma\in\{0,1\}^{|\Theta|}} \int (\bigwedge_{\sigma_i=1} \beta_i \wedge \bigwedge_{\sigma_i=0} (\neg\beta_i)) = 1. \tag{4.16}$$

and

$$\int (\bigwedge_{\sigma_i=1} \beta_i \wedge \bigwedge_{\sigma_i=0} (\neg\beta_i)) \geq 0. \tag{4.17}$$

for all $\sigma \in \{0,1\}^{|\Theta|}$. Finally, we can derive which of these conjuncts are impossible with the use of **Prob**. We will call the set of all σ s.t. $\bigwedge_{\sigma_i=1} \beta_i \wedge \bigwedge_{\sigma_i=0}(\neg\beta_i)$ is impossible by **I**.

By **GTaut** we can conjunct these theorems to mol^*_γ :

$$\begin{array}{l} \mu_\gamma \equiv \\ \quad \mu^*_\gamma \ \wedge \\ \quad\quad \bigwedge_{\sigma\in\mathbf{I}} \quad (\int(\bigwedge_{\sigma_i=1} \beta_i \wedge \bigwedge_{\sigma_i=0}(\neg\beta_i)) = 0) \quad \wedge \\ \quad\quad\quad (\sum_{\sigma\in\{0,1\}^{|\Theta|}} \int(\bigwedge_{\sigma_i=1} \beta_i \wedge \bigwedge_{\sigma_i=0}(\neg\beta_i)) = 1) \quad \wedge \\ \quad\quad\quad\quad \bigwedge_{\sigma\in\{0,1\}^{|\Theta|}} \quad (\int(\bigwedge_{\sigma_i=1} \beta_i \wedge \bigwedge_{\sigma_i=0}(\neg\beta_i)) \geq 0). \end{array} \tag{4.18}$$

The system generated by $\kappa x_\sigma^{(\bigwedge_{\sigma_i=1} \beta_i \wedge \bigwedge_{\sigma_i=0}(\neg\beta_i))}$, which is just the right hand side of the expression in (4.18) with each conjunct of the form $(\bigwedge_{\sigma_i=1} \beta_i \wedge \bigwedge_{\sigma_i=0}(\neg\beta_i))$ substituted by a fresh variable $x_\sigma$is exactly the system κ in line

2 of Algorithm 2 , which we are assuming has no solution. Therefore, by completeness of the axiomatization of the real ordered fields, it must be possible to derive $(\sim \kappa)$, and, using **ROF**, derive its instantiation $(\sim mol_\gamma)$, which completes the proof. □

Corollary 4.5.1. *The calculus presented Table 4.1 with axioms* **GTaut** *and* **Prob** *substituted by*

[**GTaut2**]	$\vdash$	δ	*for each* PEQLTL *instantiation* δ *of a tautological propositional formula;*
[**Prob2**]	$\vdash$	$(\int \varphi = 1)$	*for each* EQLTL *valid formula* φ*;*

is a complete axiomatization of the PEQLTL *fragment of* PQLTL*.*

Corollary 4.5.2. *The calculus presented Table 4.1 with axioms* **GTaut**, **Prob** *and* **ROF** *substituted by*

[**GTaut3**]	$\vdash$	δ	*for each* pLTL *instantiation* δ *of a tautological propositional formula;*
[**Prob3**]	$\vdash$	$(\int \varphi = 1)$	*for each* LTL *valid formula* φ*;*
[**LROF**]	$\vdash$	$(t_1 \leq t_2)$	*for each instantiation of a valid* linear *analytical inequality;*

is a complete axiomatization of the pLTL^+ *fragment of* PQLTL*.*

4.5.1 Example: Training for the PONG™ world championship

We now present a very basic toy example, just to illustrate the use of the calculus. The setting is as follows. PONG™ is a primitive computer game, akin to tennis, where players control a pad on each side of a screen and bounce back and forth a "ball". The world champion of PONG™ is training for the next series, but also has to watch over his young nephew. He decides to play the game with him, but the youngster has a non-negligible chance of not being able to return the "ball" successfully. Therefore, the world champion sets a "wall" in the nephew's side of the field which automatically returns the "ball" should the nephew fail to do it. Figure 4.1 summarizes the setting.

Assume that the world champion never fails to return the ball; we will use PQLTL to show that on all even transitions, the ball is returned by a human player. Notice that the "intuitive" approach of using LTL to state $(\mathsf{G}(hum \Rightarrow \mathsf{XX}hum))$ fails in this model, since the nephew would be required to keep returning the ball after his first success.

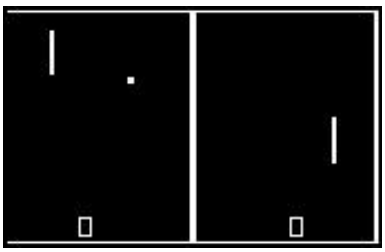


Figure 4.1: A PONG$^{\mathrm{TM}}$game between the world champion and his nephew.

In order to model this situation, we will take as hypothesis the following formulae:

a) $\int \mathsf{G}((s_1 \Rightarrow hum) \wedge (s_2 \Rightarrow hum) \wedge (s_3 \Rightarrow \neg hum)) = 1$;

b) $\int \mathsf{G}(s_1 \dot{\vee} s_2 \dot{\vee} s_3) = 1$;

c) $\int \mathsf{G}(s_1 \Rightarrow \mathsf{X}(s_2 \vee s_3) \wedge (s_2 \Rightarrow \mathsf{X}s_1) \wedge (s_3 \Rightarrow \mathsf{X}s_1)) = 1$;

d) $\int s_1 = 1$.

Where $\dot{\vee}$ denotes the *exclusive or* connective. Let Γ be the set of formulae *a*), *b*), *c*), *d*). The Markov chain of Figure4.2 is one of many that satisfies Γ and we will use it as a guideline for following the example. In this case, s_1 is a state that represents the world champion, s_2 the nephew and s_3 the wall. The propositional symbol *hum* indicates if the player is human.

Condition *a*) fixes which states pertain to human players; *b*) states that the ball can only be played by one player at a time and ensures that the players are different entities; *c*) expresses the possible transitions of the ball; *d*) just states that the world champion has the service (starts the game).

We are trying to derive the expression:

$$\int (\exists p.(p \wedge (\mathsf{X}\neg p) \wedge (\mathsf{G}(p \leftrightarrow \mathsf{XX}p)) \wedge (\mathsf{G}(p \Rightarrow hum)))) = 1. \tag{4.19}$$

The following lemma will be extensively used in the following derivations. It essentially allows propositional reasoning to be used *inside* measure terms of probability 1.

Lemma 4.5.2. Let $\Delta \vdash (\int(\beta_1 \Rightarrow \beta_2) = 1)$ and $\Delta \vdash (\int \beta_1 = 1)$. Then $\Delta \vdash (\int \beta_2 = 1)$.

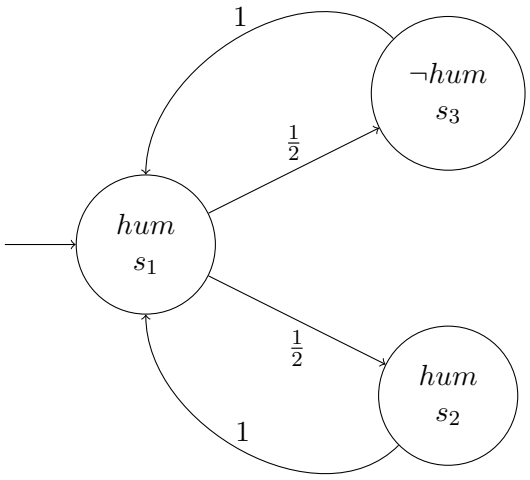


Figure 4.2: A model satisfying Γ, assuming the nephew has $\frac{1}{2}$ probability of failing to return the ball

Proof. By **Mon**, $\Delta \vdash ((\int (\beta_1 \Rightarrow \beta_2) = 1) \supset (\int \beta_1 \leq \int \beta_2))$, therefore, by **MP**, $\Delta \vdash (\int \beta_1 \leq \int \beta_2)$. Now, by **ROF** and **Prob** we get $\Delta \vdash (1 \leq \int \beta_2) \cap (\int \beta_2 \leq 1)$ and once again by **ROF**, $\Delta \vdash (\int \beta_2 = 1)$. □

From Γ, we have directly

$$\Gamma \vdash \int s_1 = 1; \tag{4.20}$$

$$\Gamma \vdash \int (\mathsf{G}(s_1 \Rightarrow hum)) = 1. \tag{4.21}$$

From b), we derive, by tautological reasoning permitted by Lemma 4.5.2, $\int \mathsf{G}((s_2 \vee s_3) \leftrightarrow (\neg s_1)) = 1$, and more concretely $\int ((s_2 \vee s_3) \leftrightarrow (\neg s_1)) = 1$. From c) we derive $\int \mathsf{G}(s_1 \Rightarrow \mathsf{X}(s_2 \vee s_3)) = 1$. From these assertions, d), and Lemma 4.5.2 again, we get

$$\Gamma \vdash \int (\mathsf{X}(\neg s_1)) = 1. \tag{4.22}$$

$\int \mathsf{G}(s_1 \Rightarrow \mathsf{X}(s_2 \vee s_3)) = 1$ allows us to get $\int \mathsf{G}((s_1 \Rightarrow \mathsf{X}s_2) \vee (s_1 \Rightarrow \mathsf{X}s_3)) = 1$. With this assertion and c), we derive

$$\Gamma \vdash \int \mathsf{G}(s_1 \Rightarrow \mathsf{XX}s_1) = 1. \tag{4.23}$$

Finally, from $\int \mathsf{G}(s_2 \Rightarrow \mathsf{X}s_1) = 1$ and $\int \mathsf{G}(s_1 \Rightarrow \mathsf{X}(s_2 \vee s_3)) = 1$ (both from c)) we derive $\int \mathsf{G}(s_2 \Rightarrow \mathsf{XX}(s_2 \vee s_3)) = 1$ and, in a similar way, $\int \mathsf{G}(s_3 \Rightarrow \mathsf{XX}(s_2 \vee s_3)) = 1$. Both these assertions together yield $\int \mathsf{G}((s_2 \vee s_3) \Rightarrow \mathsf{XX}(s_2 \vee s_3)) = 1$. Joining $\int((s_2 \vee s_3) \leftrightarrow (\neg s_1)) = 1$, which we have already justified, we get $\int \mathsf{G}((\neg s_1) \Rightarrow \mathsf{XX}(\neg s_1)) = 1$, which is classically equivalent to

$$\Gamma \vdash \int \mathsf{G}(\mathsf{XX}s_1 \Rightarrow s_1) = 1. \tag{4.24}$$

Once again by classical reasoning permitted by Lemma 4.5.2, we can join (4.20), (4.21), (4.22), (4.23), (4.24) and get

$$\Gamma \vdash \int (s_1 \wedge (\mathsf{X}\neg s_1) \wedge (\mathsf{G}(s_1 \leftrightarrow \mathsf{XX}s_1)) \wedge (\mathsf{G}(s_1 \Rightarrow hum))) = 1. \tag{4.25}$$

From the contrapositive version of Axiom $\mathsf{QX2}$ from QLTL calculus in [45], we have

$$\begin{aligned}\Gamma \vdash \int((s_1 \wedge (\mathsf{X}\neg s_1) \wedge (\mathsf{G}(s_1 \leftrightarrow \mathsf{XX}s_1)) \wedge (\mathsf{G}(s_1 \Rightarrow hum))) \Rightarrow \\ (\exists s_1.(s_1 \wedge (\mathsf{X}\neg s_1) \wedge (\mathsf{G}(s_1 \leftrightarrow \mathsf{XX}s_1)) \wedge (\mathsf{G}(s_1 \Rightarrow hum)))) = 1.\end{aligned} \tag{4.26}$$

Therefore, by **MP**, we have

$$\int (\exists s_1.(s_1 \wedge (\mathsf{X}\neg s_1) \wedge (\mathsf{G}(s_1 \leftrightarrow \mathsf{XX}s_1)) \wedge (\mathsf{G}(s_1 \Rightarrow hum)))) = 1 \tag{4.27}$$

as we wanted.

4.6 Model-checking Algorithm

Originally this section appeared in the Appendix of the published paper as it is a straightforward generalization of the model-checking algorithm for pCTL^*. We chose to present it here for completeness sake.

The model-checking problem for PQLTL consists in deciding whether a certain PQLTL formula δ, is satisfied by a given PDTS $\mathcal{M} = (S, M, d^0, L)$ for some non-specified assignment ρ. There are two main conceptual steps on the model-checking algorithm; the first step is to compute the denotation of each measure term. In order to do so, we will use an automata-theoretical algorithm for probabilistic deterministic transition systems proposed in [101], which can be seen, for instance, in [27]. We will only adapt it to use the already latent possibilities in the referenced algorithm to enclose the more expressive QLTL.

After this step, we only need to compute whether there exists an assignment ρ that satisfies the set of polynomial inequalities obtained from δ; this can be accomplished through the decidability result about ordered real closed fields.

Algorithm 5: MeasureTermCalc($\beta, \mathcal{M}$)

Input: QLTL formula β, PDTS $\mathcal{M} = (S, M, d^0, L)$
Output: $[\![\int \beta]\!]_{\mathcal{M}}$

compute DRA $R_\beta = (Q, 2^\Lambda, \tau, s^0, \alpha)$ s.t. $L^\omega(R_\beta) = L(\beta)$;

compute $\mathcal{M} \times R_\beta = (S \times Q, M', d^{0'}, L)$:

$$M'_{(s,q),(s',q')} = \begin{cases} M_{s,s'}, & \text{if } (q, L(s), q') \in \tau \\ 0, & \text{otherwise.} \end{cases};$$

$d^{0'} = (d^0, s^0)$; $L'(s,q) = L(s)$;

compute $\alpha' = \{(S \times L, S \times U) : (L, U) \in \alpha\}$;

compute $A = \bigcup\{E \subseteq S \times Q : E \text{ is ergodic}, E \text{ satisfies } \alpha'\}$;

foreach $(s,q) \in A$ **do**
  label (s,q) with acc;
end

compute

$$P^\infty(s,q) = \begin{cases} 1, \text{ if } (s,q) \text{ is labeled by } acc; \\ 0, \begin{array}{l}\text{if there are only null measure paths} \\ \text{connecting (s,q) to a state labeled by } acc; \end{array} \\ \sum_{(s',q') \in S \times Q} M'_{(s,q),(s',q')} P^\infty(s',q'), \text{ otherwise.} \end{cases}$$

return $\sum_{(s',q_0)} P^\infty(s', s^0) d(s')$

We will now describe the first step of the model-checking algorithm. The algorithm receives as input a measure term of the form $\int \beta$ and a PDTS $\mathcal{M} = (S, M, d^0, L)$, where β is a Σ_k^{QLTL} formula; it returns as output an algebraic real number, corresponding to $[\![\int \beta]\!]_{\mathcal{M}}$. This automata-theoretic approach uses the equivalence between QLTL and deterministic Rabin automata (DRA) to build a product PDTS containing the information about the QLTL formula and the PDTS given as inputs. Afterwards, as in [27], it is only a matter of computing the ergodic sets of the product PDTS, and

check which ergodic sets satisfy the accepting condition. In fact, we are just reducing the computation of $\int \beta$, for some $\beta \in \mathsf{QLTL}$ on the input PDTS to the computation of the probability of the formula $\int \mathsf{F}acc$ in a labeled product PDTS obtained from the input and the Rabin automaton of the input formula. This reduction already appears in [27].

Finally, we need to compute the probability in each state of satisfying the accepting condition. Although the definition in Line 5 is recursive, it can be solved using a system of linear equations.

Therefore, the model-checking procedure for PQLTL is fully described in Algorithm6.

Algorithm 6: PQLTLModelChecker$(\delta, \mathcal{M})$

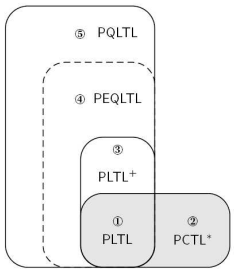
Input: PQLTL formula δ, PDTS $\mathcal{M} = (S, M, d^0, L)$
Output: *satisfied* or *not satisfied*
foreach measure term $\int \beta$ in δ **do**
  compute $c_\beta = \text{MeasureTermCalc}(\beta, \mathcal{M})$;
  replace $\int \beta \leftarrow\!\shortmid c_\beta$ in δ;
end

return $\exists$*WeakRealSat(*δ*);*

The space complexity of the model-checking algorithm depends mainly on the complexity of the algorithm that translates PQLTL formulae into deterministic Rabin automata. If we assume that our PQLTL formulae are in Σ_k^{QLTL}, the size of the deterministic Rabin automata generated is $k+2$ exponential on the size of the inner LTL formulae. The remaining instructions of Algorithm 5 can be carried out in space polynomial in the size of the system. Finally, the last instructions in Algorithm 6 will only use polynomial space, since we do not need to produce an assignment ρ. Thus, assuming that the initial formula is in EQLTL we obtain precisely the same space and time complexity as the algorithms referenced in [27].

Proposition 4.6.1. *Algorithm 6 is correct.*

Proof. The proof follows immediately from the proof of the correction of the model-checking algorithm referenced in [27]. □



not introduced in this work

① pLTL - "With probability of 99%, the process eventually reaching the critical state implies that a flag will always eventually be raised."

② pCTL*- "With probability of 98%, the process eventually reaching the critical state implies that a flag will always eventually be raised with probability 99%."

③ pLTL$^+$ - "The probability of reaching `Undecided` is at least 10% less than reaching either `Accept` or `Reject`."

④ PQLTL/ ⑤ PEQLTL- "The probability that process A refrains from reading from the common channel in all even steps and that process B refrains from doing so in all odd steps is at least 99.9%."
"The probability of reaching `Accept` is inversely proportional to that of reaching `Undecided`."

Logic	Strong SAT	Weak SAT	MC
PQLTL	[n+2]-EXPSPACE	[n+1]-EXPSPACE	[n+2]-EXPSPACE
PEQLTL	2-EXPSPACE	EXPSPACE	2-EXPSPACE
pLTL$^+$	EXPSPACE	PSPACE	PSPACE
pLTL	EXPSPACE	PSPACE	PSPACE
pCTL*	unknown	unknown	PSPACE

Table 4.2: Relations between pLTL, pCTL*, pLTL$^+$ and PQLTL.

4.7 Conclusions and future work

Herein we have proposed a new probabilistic logic to reason about semialgebraic constrains of probabilities of sets of path of a Markov chains specified by a QLTL formula. Moreover, we presented a model-checking and SAT [n+2]-EXPSPACE algorithm for the logic and a weakly complete Hilbert calculus. We have considered relevant subfragments with a more efficient SAT algorithm, namely pLTL$^+$ which extends pLTL. In Table 4.2 we summarize the results obtained versus the known results concerning pLTL [101] and pCTL [27] logics. We note that SAT algorithms presented also work for models that incorporate both non-deterministic and probabilistic transitions, such as those in [15, 27, 101], since Markov chains are particular cases of these models.

While our main goal in this work was the study of the satisfiability problem and the presentation of an Hilbert calculus for probabilistic logics closely related with pCTL*, due to the importance of pCTL in the field of Model Checking, we need to make some comparisons between the goals achieved herein and some of the known results regarding the satisfiability problem in probabilistic logics.

Our work develops a probabilistic logic from an extension of LTL without allowing nesting of probability operators, and as such, while it still does not solve the satisfiability for pCTL*, it offers an useful framework for adding *a posteriori* nesting of the probability operator. We note that the best known result regarding pCTL is the decidability of qualitative pCTL in [20]. While in [83], **Ognjanovic** proposed a probabilistic logic and presented a calculus for it. Although the calculus uses infinitary inference schemes, and as such it is much less applicable, he also proved that the class of measurable models is PSPACE-hard and in NEXPTIME, which may hint a possible connection with pLTL$^+$.

For future work we intend to enrich PQLTL with global temporal reasoning, and eventually, for the sake of exhaustiveness, nesting of the probability operator. We also intend to explore the calculus in less academic examples. It would also be interesting to prove lower complexity bounds for the logics presented, however, since we introduced probabilities and real numbers, this problem becomes quite hard to tackle, and we leave it for future work.

Concluding Remarks

As it was already mentioned in the beginning of the chapter, we could adapt PQLTL in order to reason with Markov decision processes. We would probably need to consider interval algebraic operations since terms of the form $\int \beta$ could no longer be mapped to a single probability. Instead, as we have already discussed in Section 3.1, each scheduler assigns a probability to a temporal property; one way of coping with this fact would be to state that $\int \beta$ now corresponds with an interval such that its extrema are given by the infimum and supremum over all schedulers.

Regarding the satisfiability problem for pCTL*, we feel that while unnested pCTL* or PQLTL formulae are generally enough for model-checking purposes, the real problem in the decidability of pCTL* comes from the nested applications of the probability operator. This statement can be easily seen to be true by analyzing the output of the satisfiability algorithm for PQLTL: all the models are in fact "deterministic" in a sense, since they can be described

weighted parallel composition of deterministic QLTL models; in order to obtain more complex Markov chains we need to consider nested applications of the probability operator.

Finally, we would also like to point out that the probabilization of QLTL does not strictly follow Definition 2.1.8. Most of the differences are minor but there exists a major point that has to be discussed; nevertheless for completeness sake we discuss them all:

- the syntax of the probabilization requires only as constants 0 and 1; as it was discussed in Remark 2.1.5, this does not change in any form the expressibility of the logic;

- the syntax does not have a $\Box$ operator to reason globally. While this does in fact restrict the expressibility of PQLTL, the adaptation of the results presented herein to cope with the added $\Box$ operator is straightforward;

- the major difference between the probabilization operator described in 2.1.8 and PQLTL emerges in the semantics chosen for PQLTL. Recall that the probabilization of a logic defines directly the semantics for a probabilistic formula over a probability space constructed from models of the base logic, which in this case are infinite sequences of valuations. In our case, our models are in fact the probability spaces induced by Markov chains, so we are dealing with a strict subset of the set of all probabilistic models over sequences of valuations. Another option would be to consider the Kripke semantics for QLTL; in that case a probabilistic model would be a distribution over Kripke frames, which is a concept considerably distinct from the common probabilistic concepts like Markov chains or Markov decision processes.

Chapter 5

Propositional Quantification in EPPL

The introduction of propositional quantifiers to a logic system is a topic not often approached when studying the properties of a logic, even though in some cases it allows for a more complete understanding of the logic at hand.

The most common examples of propositional quantified logics are Quantified Propositional Logic QPL and Quantified Linear Temporal Logic QLTL; these two logics illustrate what may happen when we enrich the base logic with propositional quantifiers. In the case of a quantified propositional formula, it is easy to see that the quantification can be eliminated almost directly. In the case of QLTL, the propositional quantifiers cannot be eliminated and point us out the "fault" existent in Linear Temporal Logic, namely the fact that there are ω-regular languages that can not be defined using LTL.

In either case, propositional quantifiers have an extremely significant impact as they allow many metalogic assertions to be internalized; for instance the statement ϕ is a satisfiable formula can now be written as $\exists\phi$, which becomes a valid formula; even semantic entailment $\Gamma \models \phi$ can be internalized by $\forall(\bigwedge_{\gamma\in\Gamma}\gamma) \Rightarrow \phi$, if Γ is finite. Moreover, given a formula ϕ, we can analyze its robustness by analyzing its satisfiability when it is prepended by different alternations of quantifiers.

Besides obtaining a better understanding of exogenous probabilistic propositional logic, we wish to clarify what happens when one considers propositional quantification in EPLTL. If EPPL does admit elimination of quantifiers, in quantified EPLTL any increase in expressibility (which is guaranteed since QLTL is more expressive than LTL) will be due to the temporalization

operator. Otherwise, there will be two sources to account for in the increase of expressibility.

Exogenous Probabilistic Propositional Logic was already introduced in Section 2.2 as well as the main results that will be needed in this Chapter.

5.1 Introduction

Propositional quantification has been studied thoroughly for many kinds of logics [4, 22, 96]; the simplest example of them all is the introduction of propositional quantifiers in classical propositional logic (PL), providing the most natural example of a PSPACE-complete problem [93, 98]. In quantified propositional logic (QPL), one is able to quantify over the propositional variables, gaining the possibility to state whether a certain formula is independent of some propositional symbol appearing there. In fact, using quantified propositional logic, one is able to internalize the propositional logic notions of *validity*, *satisfiability* and even *semantic entailment*. However, one should note that the introduction of quantifiers does not alter the expressibility of PL, as we can naïvely replace $\forall p(\phi(p, \mathbf{r}))$ by $\phi(\top, \mathbf{r}) \wedge \phi(\bot, \mathbf{r})$.

This work was also inspired by another extremely fruitful example: the extension of linear temporal logic (LTL) by the addition of quantifiers [96, 33]. While LTL can be easily translated to a first-order theory of linear orders with a successor function, its extension (QLTL) can only be formalized as a second-order theory. This increase in expressibility is in fact quite important, allowing the specification of many additional temporal properties describable in natural language [106].

In this work we want to understand whether the quantified extension of exogenous probabilistic propositional logic (EPPL) admits elimination of quantifiers. Propositional quantifier-free EPPL was introduced in [78], inspired in earlier probabilistic logics from **Halpern** [38, 50], as a stepping stone towards a logic for reasoning about quantum states [77] that has been used for model-checking quantum systems [47]. These exogenous logics provide two levels of reasoning where the structures of the inner level logic are used as building blocks for the structures of the outer level logic. In EPPL, the inner structures are propositional valuations and, at the outer level, the structures are probability distributions over valuations. The term exogenous follows **Kozen**'s terminology [66], expressing that the logic has proper terms to reason about probabilities, and that the probabilities are not hidden in the syntactic constructors. Temporal extensions of EPPL were proposed in [9], like EPLTL, and the decidability of its SAT problem can be applied

to planning under uncertainty, which is the focus of Chapter 6; moreover, this temporal extension of EPPL also opened up a novel approach in the verification of properties in Markov chains, which is discussed in Chapter 7.

Our main goal in this work is to use EPPL as a probabilistic alternative to classical propositional logic in order to obtain probabilistic logics able to perform temporal, modal, or dynamical reasoning over probabilities. As already mentioned, propositional quantification in temporal logics provides a vast increase in expressibility, however in order to fully study the quantified temporal extensions of EPPL, one must understand first if the introduction of quantifiers in EPPL can be eliminated, as in the case of PL.

If in fact EPPL admits elimination of propositional quantification, we then can expect that any gain on expressiveness when considering quantified temporal probabilistic formulae can be pinpointed to the temporalization; however, assuming that quantifiers can not be eliminated from quantified EPPL (QEPPL), there will be two sources for this fact, as both the temporalization and the probabilization of propositional logic do not have elimination of quantifiers.

There is however an additional difficulty before we are able to investigate the quantification of EPPL; while in first-order the semantics of quantifiers is well established, in the case of propositional-based logics the case is not as straightforward. For instance, the FOL formula $\forall x \phi(x, \mathbf{r})$ is satisfied on some interpretation $\mathcal{I}$ and assignment ρ, if and only if any other assignment ρ' *varying only in the variable* x satisfies $\phi(x, \mathbf{r})$. However, when we consider propositional logics, the semantic meaning of a propositional quantifier is not obvious, since the notion of p-equivalence between models is not as straightforward.

Our contributions

In this Chapter we show that EPPL endowed with propositional quantification admits elimination of propositional quantification.

1. We define an appropriate p-equivalence relation between EPPL structures and we discuss its meaning.

2. We show that QEPPL admits an adapted version of the lemma of absent variables.

    - The lemma of absent variables allow us to focus our efforts on finite structures.

- This lemma is particularly difficult to prove in QEPPL, as we are required to build a standard probability space over a countable infinite number of propositional symbols obeying an infinite number of restrictions imposed by the p-equivalence, using as a building block a finite probability space.

3. We show that QEPPL allows quantifier elimination, that is, for each quantifier-free formula $\psi \in \mathsf{EPPL}$ there is a quantifier-free formula $\varphi \in \mathsf{EPPL}$ such that

$$\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\{p,p_1,\ldots,p_n\})} \exists p.\psi(p, p_1, \ldots, p_n) \text{ iff}$$
$$\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\{p,p_1,\ldots,p_n\})} \varphi(p_1, \ldots, p_n).$$

The paper is structured as follows: we focus on building and discussing the notion of p-equivalence in Section 5.2. In Section 5.3, we present the extension of EPPL with propositional quantification. Finally, in Section 5.4 we show the desired result.

5.2 Propositional p-Equivalence relation between EPPL models

In this section we will discuss the appropriate p-equivalence relation between EPPL models. This will directly give meaning to an universal/existential propositional quantified EPPL formula. In most propositional-based logics, the notion of p-equivalence between models is easily found and easy to accept. However, to our knowledge there has not been any study of p-equivalence between probabilistic propositional-based models.

The relevant notion of p-equivalence between models in most logics can be obtained by noting that two models M_1 and M_2 are p-equivalent if and only if any formula ϕ not containing the propositional symbol p is satisfiable in M_1 if and only if it is satisfiable in M_2, that is, any formula ϕ not dependent on p cannot distinguish between M_1 and M_2. This concept of course relies on the degree of expressiveness of the logic at hand.

Example 5.2.1. Let v_1, v_2 be propositional valuations over Λ. Let $q \neq p \in \Lambda$ be any propositional symbol. Then, in order for v_1 to be p-equivalent to v_2, $v_1 \Vdash q$ iff $v_2 \Vdash q$ has to hold. Trivially, that implies $v_1(q) = v_2(q)$ for any propositional symbol different from p, which is the usual notion of p-equivalence relation in PL.

Let us now consider two infinite sequences of valuations $v = v_1 v_2 \ldots$, $u = u_1 u_2 \ldots$, models of LTL. Let $q \neq p \in \Lambda$; in order for v to be p-equivalent to u, $v \Vdash^{\mathsf{LTL}} q$ iff $u \Vdash^{\mathsf{LTL}} q$ has to hold; therefore $v(0)(q) = u(0)(q)$, for all propositional symbols $q \neq p$. Moreover, $v \Vdash^{\mathsf{LTL}} \mathsf{X}^i q$ iff $u \Vdash^{\mathsf{LTL}} \mathsf{X}^i q$, which is equivalent to $v(i)(q) = u(i)(q)$ for all natural numbers i and all propositional symbols $q \neq p$. This definition is the standard definition of p-equivalence between LTL models.

We shall now construct the definition of p-equivalence accordingly. Recall that the syntax and semantics for EPPL were given in Definition 2.2.1 and Definition 2.2.4 respectively. Let $(\mathcal{P}_i, \rho_i)$ with $\mathcal{P}_i = (V_i, \sigma(\{0,1\}^\Lambda), \nu_i)$, for $i \in \{1,2\}$ be EPPL models. Then we can obtain the necessary and sufficient conditions for the p-equivalence to hold by considering the following constraints:

- Consider formulae $\Diamond q$, for $q \neq p$. Since $\mathcal{P}_1, \rho_1 \Vdash^{\mathsf{EPPL}} \Diamond q$ iff $\mathcal{P}_2, \rho_2 \Vdash^{\mathsf{EPPL}} \Diamond q$ has to hold, we obtain that $B_{q \mapsto 1} \cap V_1 \neq \emptyset$ iff $B_{q \mapsto 1} \cap V_2 \neq \emptyset$. By considering all propositional symbols q different from p we obtain that a necessary condition for the two models to be p-equivalent is that $B_v \cap V_1 \neq \emptyset$ iff $B_v \cap V_2 \neq \emptyset$, for any valuation v with $\#dom(v) < +\infty, p \notin dom(v)$.

- Formulae like $x = r$ allows us to conclude that $\rho_1(x) = \rho_2(x)$, for any real variable x, since the interpretation of the constants r are fixed.

- More interestingly, formulae like $\int \beta = r$ enforce that $\nu_1(B_v \cap V_1) = \nu_1(B_v) = \nu_2(B_v) = \nu_2(B_v \cap V_2)$ has to hold, again for any valuation v with $\#dom(v) < +\infty, p \notin dom(v)$.

We propose the following definition of p-equivalence:

Definition 5.2.1. Let $(\mathcal{P}_i, \rho_i)$ with $\mathcal{P}_i = (V_i, \sigma(\{0,1\}^\Lambda), \nu_i)$, for $i \in \{1,2\}$ be EPPL models, and ρ_1, ρ_2 assignments. Let $p \in \Lambda$. Then, $(\mathcal{P}_1, \rho_1)$ is said to be p-equivalent to $(\mathcal{P}_2, \rho_2)$, represented as $(\mathcal{P}_1, \rho_1) \sim_p (\mathcal{P}_2, \rho_2)$, if for any $\Pi \in \mathcal{P}_{fin}(\Lambda - \{p\})$ and for any $v \in \{0,1\}^\Pi$, the following conditions are fulfilled:

- $B_v \cap V_1 \neq \emptyset$ iff $B_v \cap V_2 \neq \emptyset$,

- $\nu_1(B_v \cap V_1) = \nu_2(B_v \cap V_2)$,

- $\rho_1 = \rho_2$.

We note that $\sim_p$ is in fact an equivalence relation. Moreover, we will represent the p-equivalence relation simply as $\mathcal{P}_1 \sim_p \mathcal{P}_2$ since the requirement relative to the assignments force them to be the same.

Remark 5.2.1. We note that while the definition of p-equivalence seems to rely heavily on the existence of the constants $r \in \mathbb{R}_{\mathsf{Alg}}$, we could easily be lead to the same conclusion by considering the existence of only 0 and 1; a real algebraic number r is represented by a quantifier-free first-order formula ϕ_r in the signature of the ordered real closed fields, therefore any two EPPL models will have to agree in the truth value of ϕ_r, which will be enough to obtain the desired p-equivalence.

Example 5.2.2. Consider probability spaces $\mathcal{P}_i = (V_i, \mathcal{P}(\{p, q_1, q_2\}), \nu_i)$, with $i \in \{1, \ldots, 4\}$ over the propositional symbols $\{p, q_1, q_2\}$. The probability spaces are represented in Figure 5.1; for instance, the valuation v such that $p \mapsto 0, q_1 \mapsto 0, q_2 \mapsto 1$, with measure $\nu_1(\{v\}) = \frac{1}{4}$ belonging to V_1, is represented by $(0, 0, 1) \mapsto \frac{1}{4}$.

In this example $\mathcal{P}_2 \sim_p \mathcal{P}_3$; we note that each set of valuations agreeing on propositional symbols different from p may gain or lose elements, all that matters is that the probability remains assigned equally. For instance the set $B_{q_1 \mapsto 1, q_2 \mapsto 1} \cap V_3$ has more elements than in $\mathcal{P}_2$, but the measure of the set remains equal.

Neither $\mathcal{P}_1$ nor $\mathcal{P}_4$ are equivalent to $\mathcal{P}_2$. In the case of $\mathcal{P}_1$, that is due to a discrepancy on the assignment of measure to the cylinders; in the case of $\mathcal{P}_4$, the problem stems from the fact that the set $B_{q_1 \mapsto 1, q_2 \mapsto 1} \cap V_4$ is not empty while in all other probability spaces the respective set is indeed empty, even if their probability measures are compatible.

There are two noteworthy models for each equivalence class generated by the $\sim_p$ relation; given a probability model $\mathcal{P}$ it should be possible to obtain an equivalent one where any valuation asserts the truthfulness/falseness of p. We represent the model obtained from $\mathcal{P}$ where p is always true/false by $\mathcal{P}^p_\top / \mathcal{P}^p_\bot$.

Lemma 5.2.1. *Let $\mathcal{P} = (V, \sigma(\{0,1\}^\Lambda), \nu)$ be a EPPL model over Λ. Consider $\mathcal{P}^p_\top = (V', \sigma(\{0,1\}^\Lambda), \nu')$ obeying the following restrictions:*

- *$V' = \{v \in \{0,1\}^\Lambda : v(p) = 1, \exists w \in V$ s.t. $w(q) = v(q)$ for all $q \neq p \in \Lambda\}$;*
- *if $p \notin dom(v)$, then $\nu'(B_v \cap V') = \nu(B_v \cap V)$ and $\nu'(B_{v, p \mapsto 1} \cap V') = \nu(B_v \cap V)$.*

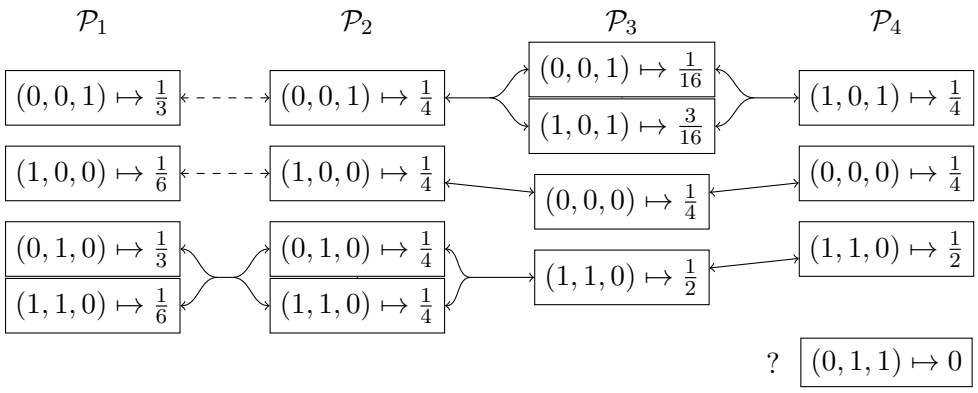


Figure 5.1: Four Probability spaces such that only $\mathcal{P}_2 \sim_p \mathcal{P}_3$. Note that $\mathcal{P}_1$ fails to be p-equivalent to $\mathcal{P}_2$ due to the restrictions imposed on the measure of the cylinders; the probability space $\mathcal{P}_4$ is not p-equivalent to $\mathcal{P}_2$ due to the restriction on the cylinders, even though it satisfies the measure restrictions.

Then $\mathcal{P}^p_\top \in \mathfrak{M}(\mathsf{EPPL}(\Lambda))$ is uniquely defined and $\mathcal{P}^p_\top \sim_p \mathcal{P}$.

Proof. We need to show that $\mathcal{P}^p_\top$ is in fact a EPPL model. The proof that $\mathcal{P}^p_\top \sim_p \mathcal{P}$ holds follows directly from the definition.

First we need to show that V' is measurable in $\sigma(\{0,1\}^\Lambda)$. Let $v(p) = 1$ and $dom(v) = \{p\}$. Then, $V = C \cup D$, where $C = B_v \cap V$ and $D = B_{\overline{v}^p} \cap V$; now we note that both C and D can be written, assuming p is the first propositional symbol, as $C = \{p \mapsto 1\} \times C'$ and $D = \{p \mapsto 0\} \times D'$. Since $\sigma(\{0,1\}^\Lambda)$ is a Borel σ-algebra both C' and D' are measureable over $\sigma(\{0,1\}^{\Lambda - \{p\}})$. Therefore V' is obtained simply as $(\{p \mapsto 1\} \times C') \cup (\{p \mapsto 1\} \times D')$.

Secondly, we need to check that ν' as defined is indeed a pre-measure. We note that the restrictions assign a pre-measure to all cylinders; moreover the pre-measure is in fact σ-additive, as it is clearly additive and any cylinder described as an union of a family of cylinders can be written instead an union of a **finite** family of cylinders, as the topology is compact. As such by Caratheodory's extension theorem the pre-measure can be extended uniquely to a full probability measure over $\sigma(\{0,1\}^\Lambda)$; that allows us to conclude that $\mathcal{P}^p_\top$ is in fact a EPPL model and it is uniquely defined. □

One can further extend the Lemma above by considering a model $\mathcal{P}^p_q$ which is $\sim_p$ to $\mathcal{P}$ that mimics the behavior in p of q.

Lemma 5.2.2. *Let $\mathcal{P} = (V, \sigma(\{0,1\}^\Lambda), \nu)$ be a EPPL model over Λ. Consider $\mathcal{P}_q^p = (V', \sigma(\{0,1\}^\Lambda), \nu')$ obeying the following restrictions:*

- $V' = \{v \in \{0,1\}^\Lambda : \exists w \in V \text{ s.t. } w(r) = v(r) \text{ for all } r \neq p \text{ and } v(p) = w(q)\}$.
- $$\nu'(B_v \cap V') = \begin{cases} \nu(B_v \cap V), \text{ if } p \notin dom(v) \\ \nu'(B_{v,q\mapsto 0} \cap V') + \nu'(B_{v,q\mapsto 1} \cap V'), \text{ if } q \notin dom(v) \text{ and } p \in dom(v) \\ \nu(B_{v',q\mapsto 0} \cap V), \text{ if } v = v', q \mapsto 0 \text{ and } v(p) = 0 \\ 0, \text{ if } v = v', q \mapsto 0 \text{ and } v(p) = 1 \\ \nu(B_{v',q\mapsto 1} \cap V), \text{ if } v = v', q \mapsto 1 \text{ and } v(p) = 1 \\ 0, \text{ if } v = v', q \mapsto 1 \text{ and } v(p) = 0. \end{cases}$$

Then $\mathcal{P}_q^p \in \mathfrak{M}(\mathsf{EPPL}(\Lambda))$ is uniquely defined and $\mathcal{P}_q^p \sim_p \mathcal{P}$.

Proof. Again, we need to show that V' is measureable in $\sigma(\{0,1\}^\Lambda)$. The proof is similar to Lemma 5.2.1. Now $V = A \cup B \cup C \cup D$, with $A = B_{p\mapsto 0,q\mapsto 0} \times A'$, $B = B_{p\mapsto 1,q\mapsto 0} \times B'$, $C = B_{p\mapsto 0,q\mapsto 1} \times C'$ and $D = B_{p\mapsto 1,q\mapsto 1} \times D'$. Again A', B', C', D' are all measureable in the Borel σ-algebra over $\{0,1\}^{\Lambda - \{p,q\}}$ and $V' = (\{p \mapsto 0\} \times \{q \mapsto 0\} \times A') \cup (\{p \mapsto 0\} \times \{q \mapsto 0\} \times B') \cup (\{p \mapsto 1\} \times \{q \mapsto 1\} \times C') \cup (\{p \mapsto 1\} \times \{q \mapsto 1\} \times D')$.

It can be shown that the restrictions imposed do in fact define a pre-measure over the all cylinders, and as such the measure is uniquely defined. In addition the measure requirements in the $\sim_p$ equivalence relation are obtained directly from the first condition. □

Lemma 5.2.3. *Let $\mathcal{P} = (V, \sigma(\{0,1\}^\Lambda), \nu)$, $\mathcal{Q} = (W, \sigma(\{0,1\}^\Lambda), \mu)$. Furthermore, let $\mathcal{P}_q^p = (V', \sigma(\{0,1\}^\Lambda), \nu')$, $\mathcal{Q} = (W', \sigma(\{0,1\}^\Lambda), \mu')$. Let z be a propositional symbol in Λ different from p and q.*

If $\mathcal{P} \sim_z \mathcal{Q}$ then $\mathcal{P}_q^p \sim_z \mathcal{Q}_q^p$.

Proof. Let v be a valuation such that $\#dom(v) < +\infty$ and $z \notin dom(v)$.

We need to show that $B_v \cap V' \neq \emptyset$ iff $B_v \cap W' \neq \emptyset$. By symmetry we just need to prove one side of the equivalence. Suppose that $B_v \cap V' \neq \emptyset$. Suppose that $p \neq dom(v)$. Then,

$$\begin{aligned} &B_v \cap V' \neq \emptyset \text{ then, by } \sim_p \\ &B_v \cap V \neq \emptyset \text{ then, by } \sim_z \\ &B_v \cap W \neq \emptyset \text{ then, by } \sim_p \\ &B_v \cap W' \neq \emptyset \end{aligned}$$

Now, suppose that $p, q \in dom(v)$, that is $v = v', p \mapsto i, q \mapsto j$, with $i, j \in \{0, 1\}$; note that if $i \neq j$, then $B_{v', p \mapsto i, q \mapsto j} \cap V' = B_{v' p \mapsto i, q \mapsto j} = \emptyset$, by construction of V' and W'. Let $B_{v', p \mapsto i, q \mapsto i} \cap V' \neq \emptyset$. Then,

$$
\begin{aligned}
&B_{v', p \mapsto i, q \mapsto i} \cap V' \neq \emptyset \text{ then, by construction of } V' \\
&B_{v', q \mapsto i} \cap V' \neq \emptyset \text{ then, by } \sim_p \\
&B_{v', q \mapsto i} \cap V \neq \emptyset \text{ then, by } \sim_z \\
&B_{v', q \mapsto i} \cap W \neq \emptyset \text{ then, by } \sim_p \\
&B_{v', q \mapsto i} \cap W' \neq \emptyset \text{ then, by construction of } W' \\
&B_{v', q \mapsto i, p \mapsto i} \cap W' \neq \emptyset.
\end{aligned}
$$

In the case that $q \notin dom(v)$, $B_{v', p \mapsto i} \cap V' \neq \emptyset$ iff $B_{v', p \mapsto i, q \mapsto i} \cap V' \neq \emptyset$, so the above proof suffices.

Regarding the measure conditions, suppose that $p \neq dom(v)$. Then,

$$
\begin{aligned}
&\nu'(B_v \cap V') = \text{ by } \sim_p \\
&\nu(B_v \cap V) = \text{ by } \sim_z \\
&\mu(B_v \cap W) = \text{ by } \sim_p \\
&\mu'(B_v \cap W').
\end{aligned}
$$

Again if $p, q, \in dom(v)$, $\nu'(B_{v', p \mapsto i, q \mapsto i} \cap V') = \mu'(B_{v', q \mapsto i, p \mapsto i} \cap W' = 0$ if $i \neq j$. Otherwise,

$$
\begin{aligned}
&\nu'(B_{v', p \mapsto i, q \mapsto i} \cap V') = \text{ then, by construction of } V' \\
&\nu'(B_{v', q \mapsto i} \cap V') = \text{ then, by } \sim_p \\
&\nu(B_{v', q \mapsto i} \cap V)) = \text{ then, by } \sim_z \\
&\mu(B_{v', q \mapsto i} \cap W) = \text{ then, by } \sim_p \\
&\mu'(B_{v', q \mapsto i} \cap W') = \text{ then, by construction of } W' \\
&\mu(B_{v', q \mapsto i, p \mapsto i} \cap W').
\end{aligned}
$$

Finally, if $q \notin dom(v)$, the above prove also suffices because $\mu'(B_{v', p \mapsto i} \cap V') = \mu'(B_{v', p \mapsto i, q \mapsto i} \cap V')$. □

5.3 Quantified Exogenous Probabilistic Propositional Logic

Having a notion of p-equivalence between EPPL models at hand, we are able to easily enrich the syntax and semantics of EPPL by introducing propositional quantifiers. We will denote this extension by QEPPL. As already

discussed in the Introduction, our aim is to understand fully the temporal extensions of EPPL. In order to do so, we have to clarify whether QEPPL admits quantifier elimination. Even if QEPPL is in fact as expressive as EPPL, some properties should be easier to express using propositional quantification.

After presenting the syntax and semantics of QEPPL, we will discuss the intuitive meaning and usefulness of propositional quantification. Afterwards, we will prove the intermediary but necessary results for paving the way for the next section.

A QEPPL formula is built from the same constructs as in EPPL, with the added quantification over propositional symbols Λ. As usual, we introduce the abbreviation $\forall p.\delta \equiv (\sim\exists p.(\sim\delta))$. Furthermore $fpvar(\delta)$ represents the set of free propositional symbols, that is, propositional symbols not in the scope of any quantifier.

Definition 5.3.1. The *well formed formulae* in $\mathsf{QEPPL}(\Lambda)$ are given below in Backus-Naur notation:

$$\begin{aligned}
\beta &:= p \quad | \quad (\beta \wedge \beta) \quad | \quad (\neg\beta) \quad | \quad \top \\
t &:= x \quad | \quad (t+t) \quad | \quad (tt) \quad | \quad r \quad | \quad \int \beta \\
\delta &:= \underbrace{\Box\beta}_{\text{global atoms}} \quad | \quad \sim\delta \quad | \quad \exists p.\delta \quad | \quad \delta \cap \delta \quad | \quad \underbrace{t \leq t}_{\text{comparison atoms}}
\end{aligned}$$

where $p \in \Lambda$, $x \in Z$, $\top$ is constant true (*verum*) and $r \in \mathbb{R}_{\mathsf{Alg}}$ is an algebraic real number.

Definition 5.3.2. The satisfaction relation between a EPPL model $\mathcal{P} = (V, \sigma(\{0,1\}^\Lambda), \nu)$ over Λ, an assignment $\rho : Z \mapsto \mathbb{R}_{\mathsf{Alg}}$, and a formula $\delta \in \mathsf{QEPPL}(\Lambda)$ is defined inductively as follows:

- $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\Lambda)} \Box\beta$ iff for all $v \in V, v \Vdash \beta$,
- $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\Lambda)} (t_1 \leq t_2)$ iff $[\![t_1]\!]_{(\mathcal{P},\rho)} \leq [\![t_2]\!]_{(\mathcal{P},\rho)}$,
- $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\Lambda)} (\sim\delta)$ iff $\mathcal{P}, \rho \nVdash^{\mathsf{QEPPL}(\Lambda)} \delta$,
- $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\Lambda)} (\delta_1 \cap \delta_2)$ iff $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\Lambda)} \delta_1$ and $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\Lambda)} \delta_2$.
- $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\Lambda)} (\exists p.\delta)$ iff there exists $\mathcal{P}' \sim_p \mathcal{P}$ s.t. $\mathcal{P}', \rho \Vdash^{\mathsf{QEPPL}(\Lambda)} \delta$.

The notion of entailment $\models^{\mathsf{QEPPL}(\Lambda)}$ is defined akin to the $\mathsf{EPPL}(\Lambda)$ entailment relation. As in EPPL, we will drop the suffix (Λ) when it is clear from the context which entailment/satisfaction relation we are dealing with.

Example 5.3.1 (Some Quantified formulae)**.** We present some simple formulae in order to gain some intuition about the effects of the propositional quantifier. The formulae we are going to present do not include algebraic variables and as such we will not be concerned with assignments ρ. Moreover our set of propositional symbols is $\{p, q\}$, making our models quite small.

Consider the formula $\delta_1 \equiv \exists p. \int p \vee q = \frac{2}{3}$. We wish to characterize the class of EPPL models that satisfy δ_1; a model $\mathcal{P}$ satisfies δ_1 iff there exists some p-equivalent model $\mathcal{P}'$ such that $\mathcal{P}' \Vdash \int p \vee q = \frac{2}{3}$. The models that satisfy this formula are represented in Figure 5.2; we are working with full models, that is, the sample space is always $\{0,1\}^{\{p,q\}}$; we could extend the examples for incomplete sets, but the example would essentially be the same.

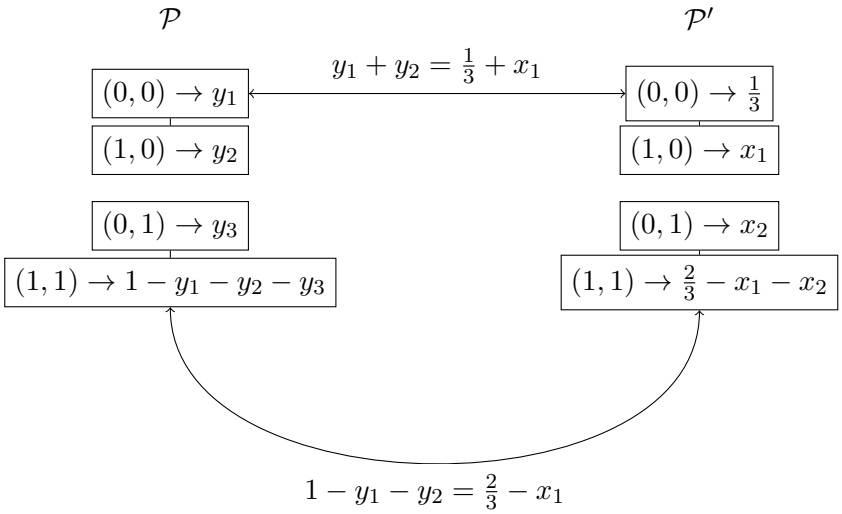


Figure 5.2: Schematization representing the models that satisfy $\int p \vee q = \frac{2}{3}$, and models that are p-equivalent. Note that there is an implicit restriction that all values are between 0 and 1.

We note that as long as the cylinder $B_{q \mapsto 0}$ has measure ν greater or equal than $\frac{1}{3}$, there will be always a p-equivalent model such that $\mu(B_{p \mapsto 0, q \mapsto 0})$ is precisely $\frac{1}{3}$; therefore the only restriction imposed upon $\mathcal{P}$ is that it must satisfy $\int \neg q \geq \frac{1}{3}$. So, $\models^{\mathsf{QEPPL}} \delta_1 \Leftrightarrow \int \neg q \geq \frac{1}{3}$.

Now consider $\delta_2 \equiv \forall p. \int p \vee q \geq \frac{2}{3}$. In this case, the class of models that satisfy δ_2 can be represented by $\int q \geq \frac{2}{3}$. In order for $\int p \vee q \geq \frac{2}{3}$ to hold, one must have $\mu(B_{p\mapsto 0, q\mapsto 0}) \leq \frac{1}{3}$). Therefore, as long as $\int q$ is sufficiently large, ($\geq \frac{2}{3}$), all p-equivalent models do in fact verify $\int q \geq \frac{2}{3}$. So, $\models^{\mathsf{QEPPL}} \delta_2 \Leftrightarrow \int q \geq \frac{2}{3}$.

Remark 5.3.1. A trivial consequence of the $\sim_p$ equivalence relation is that if $\mathcal{P}_1$ and $\mathcal{P}_2$ are p-equivalent, then $[\![t]\!]_{\mathcal{P}_1,\rho} = [\![t]\!]_{\mathcal{P}_2,\rho}$, if p is not a propositional symbol in p. The proof directly follows using structural induction on the term t and the definition of $\sim_p$ equivalence.

We will begin our study of QEPPL by proving some important lemmas that will be useful later in order to prove our main result.

Lemma 5.3.1 (Substitution Lemma)**.** *Let $\phi, \psi \in \mathit{QEPPL}(\Lambda)$ be such that $\models^{\mathit{QEPPL}(\Lambda)} \psi \Leftrightarrow \phi$. Denote $\lceil \delta \rceil^{\phi}_{\psi}$ the substitution of all the occurrences in $\delta \in \mathit{QEPPL}(\Lambda)$ of the subformula ϕ by the subformula ψ. Then $\models^{\mathit{QEPPL}(\Lambda)} \delta \Leftrightarrow \lceil \delta \rceil^{\phi}_{\psi}$.*

Proof. The proof follows from structural induction on δ. The basis cases are trivial, as either ϕ is δ and therefore the equivalence follows, or δ cannot have ϕ as a subformula.

Suppose that $\delta \equiv \delta_1 \cap \delta_2$ and ϕ is different from δ; in this case, ϕ might be a subformula of δ_1 and δ_2. By the induction hypothesis, we obtain that $\models^{\mathsf{QEPPL}} \delta_i \Leftrightarrow \lceil \delta_i \rceil^{\phi}_{\psi}$ for $i \in \{1, 2\}$. Therefore, $\models^{\mathsf{QEPPL}} \delta \Leftrightarrow \lceil \delta_1 \cap \delta_2 \rceil^{\phi}_{\psi}$. The negation case is similar.

Suppose that $\delta \equiv \exists p. \delta_1$, and ϕ is not δ. Then ϕ may only be a subformula of δ_1 so, by the induction hypothesis we obtain $\models^{\mathsf{QEPPL}} \delta_1 \Leftrightarrow \lceil \delta_1 \rceil^{\phi}_{\psi}$. But then for any model $\mathcal{P}$ and any assignment ρ:

$$
\begin{aligned}
&\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \exists p. \delta_1 \text{ iff} \\
&\mathcal{P}', \rho \Vdash^{\mathsf{QEPPL}} \delta_1 \text{ for some } \mathcal{P}' \sim_p \mathcal{P} \text{ iff} \\
&\mathcal{P}', \rho \Vdash^{\mathsf{QEPPL}} \lceil \delta_1 \rceil^{\phi}_{\psi} \text{ for some } \mathcal{P}' \sim_p \mathcal{P} \text{ iff} \\
&\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \exists p. \lceil \delta_1 \rceil^{\phi}_{\psi}
\end{aligned}
$$

□

Lemma 5.3.2. *Assume that $\models^{\mathit{QEPPL}(\Lambda)} (\exists p. \gamma(p, \mathbf{r})) \Leftrightarrow \delta(p, \mathbf{r})$, where $\gamma, \delta \in \mathit{EPPL}(\Lambda)$.*

Then there exists $\delta'(\mathbf{r}) \in \mathit{EPPL}(\Lambda)$ such that $\models^{\mathit{QEPPL}(\Lambda)} (\exists p. \gamma(p, \mathbf{r})) \Leftrightarrow \delta'(\mathbf{r})$.

Proof. The formula $\delta'(\mathbf{r})$ is defined to be $\lceil \delta(p,\mathbf{r}) \rceil_{\top}^{p}$, that is the substitution of all occurrences of the propositional symbol p by *verum* ($\top$). We start by showing that $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \lceil \delta(p,\mathbf{r}) \rceil_{\top}^{p}$ if and only if $\mathcal{P}_{\top}^{p}, \rho \Vdash^{\mathsf{QEPPL}} \delta(p,\mathbf{r})$.

This proof follows from structural induction. The step cases are trivial, as δ is quantifier-free, and we only need to prove the statement for global atoms and comparison terms. Let $\mathcal{P} = (V, \sigma(\{0,1\}^{\Lambda}, \nu)$ and $\mathcal{P}_{\top}^{p} = (V', \sigma(\{0,1\}^{\Lambda}, \nu)$, as in Lemma 5.2.1.

For the case of global atoms, let $\delta \equiv \Box\beta$. Then $\mathcal{P}_{\top}^{p}, \rho \Vdash^{\mathsf{EPPL}} \Box\beta$ iff $\forall v \in V', v \Vdash \beta$ iff $\forall v \in V', v \Vdash \lceil\beta\rceil_{\top}^{p}$ iff $\forall v \in V, v \Vdash \lceil\beta\rceil_{\top}^{p}$ iff $\mathcal{P}, \rho \Vdash^{\mathsf{EPPL}} \lceil\Box\beta\rceil_{\top}^{p}$.

In the case of comparison terms, we show that $[\![\lceil t \rceil_{\top}^{p}]\!]_{\mathcal{P},\rho} = [\![t]\!]_{\mathcal{P}_{\top}^{p},\rho}$. This statement is again proved by structural induction on the probabilistic term t. The only non-trivial case is $t \equiv \int \beta$. Let $W = \{0,1\}^{\{p\}\cup\{\mathbf{r}\}}$, for the purposes of this proof.

$$
\begin{aligned}
[\![\textstyle\int \lceil \beta(p,\mathbf{r}) \rceil_{\top}^{p}]\!]_{(\mathcal{P},\rho)} &= \sum_{\substack{v \Vdash \lceil \beta(p,\mathbf{r}) \rceil_{\top}^{p} \\ v \in W}} \nu(B_v \cap V) = \\
\sum_{\substack{v \in W, v(p)=1 \\ v \Vdash \lceil \beta(p,\mathbf{r}) \rceil_{\top}^{p}}} \nu(B_v \cap V) &+ \sum_{\substack{v \in W, v(p)=0 \\ v \Vdash \lceil \beta(p,\mathbf{r}) \rceil_{\top}^{p}}} \nu(B_v \cap V) = \\
\sum_{\substack{v \in W, v(p)=1 \\ v \Vdash \lceil \beta(p,\mathbf{r}) \rceil_{\top}^{p}}} \nu(B_v \cap V) &+ \sum_{\substack{v \in W, v(p)=1 \\ v \Vdash \lceil \beta(p,\mathbf{r}) \rceil_{\top}^{p}}} \nu(B_{\overline{v}^{p}} \cap V) = \\
\sum_{\substack{v \in W, v(p)=1 \\ v \Vdash \lceil \beta(p,\mathbf{r}) \rceil_{\top}^{p}}} (\nu(B_v \cap V) + \nu(B_{\overline{v}^{p} \cap V})) &= \sum_{\substack{v \in W, v(p)=1 \\ v \Vdash \lceil \beta(p,\mathbf{r}) \rceil_{\top}^{p}}} \nu'(B_v \cap V') = \\
\sum_{\substack{v \in W, v(p)=1 \\ v \Vdash \beta(p,\mathbf{r})}} \nu'(B_v \cap V') &= \sum_{\substack{v \in W \\ v \Vdash \beta(p,\mathbf{r})}} \nu'(B_v \cap V') = [\![\textstyle\int \beta(p,\mathbf{r})]\!]_{(\mathcal{P},\rho)_{\top}^{p}}.
\end{aligned}
$$

Assume $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \exists p.\gamma(p,\mathbf{r})$:

$$
\begin{aligned}
&\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \exists p.\gamma(p,\mathbf{r}) \text{ then,} \\
&\mathcal{P}', \rho \Vdash^{\mathsf{QEPPL}} \exists p.\gamma(p,\mathbf{r}), \text{ for all } \mathcal{P}' \sim_{p} \mathcal{P} \text{ then,} \\
&\mathcal{P}', \rho \Vdash^{\mathsf{QEPPL}} \delta(p,\mathbf{r}), \text{ for all } \mathcal{P}' \sim_{p} \mathcal{P} \text{ then,} \\
&\mathcal{P}_{\top}^{p}, \rho \Vdash^{\mathsf{QEPPL}} \delta(p,\mathbf{r}) \text{ then,} \\
&\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \lceil \delta(p,\mathbf{r}) \rceil_{\top}^{p}.
\end{aligned}
$$

For the converse implication, assume $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \lceil \delta(p, \mathbf{r}) \rceil^p_\top$:

$$\begin{aligned}
&\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \lceil \delta(p, \mathbf{r}) \rceil^p_\top \text{ then,}\\
&\mathcal{P}^p_\top, \rho \Vdash^{\mathsf{QEPPL}} \delta(p, \mathbf{r}) \text{ then,}\\
&\mathcal{P}^p_\top, \rho \Vdash^{\mathsf{QEPPL}} \exists p.\gamma(p, \mathbf{r}) \text{ then,}\\
&\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \exists p.\gamma(p, \mathbf{r}).
\end{aligned}$$

□

These two results are expected, and serve as a support for the quantifier elimination procedure. We will now show that propositional quantification obey the same axioms as first-order quantifiers. These valid formulae allow us to reason with propositional quantification as usual in first-order logic.

Lemma 5.3.3. *Let $\delta, \delta_1, \delta_2 \in \mathsf{QEPPL}(\Lambda)$.*

- *If $\models^{\mathsf{QEPPL}} \delta$ then $\models \forall p.\delta$.*
- *$\models^{\mathsf{QEPPL}} \forall p.\delta \supset \lceil \delta \rceil^p_q$, where q is a free variable in δ.*

Proof. If any $\mathcal{P} \in \mathfrak{M}(\mathsf{EPPL}(\Lambda))$ is such that $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \delta$, then any $\mathcal{P}' \sim_p$ equivalent to $\mathcal{P}$ is such that $\mathcal{P}', \rho \Vdash \delta$ and as such $\mathcal{P} \Vdash \forall p.\delta$.

The proof of $\models^{\mathsf{QEPPL}} \forall p.\delta \supset \lceil \delta \rceil^p_q$ relies heavily on showing that $\mathcal{P}^p_q \Vdash \delta$ iff $\mathcal{P} \Vdash \lceil \delta \rceil^p_q$. The basis steps are similar to the proof of Lemma 5.3.2. The structural induction steps are also straightforward with the notable exception of propositional quantification connective.

Suppose that $\delta \equiv \exists p.\delta_1$. Then $\mathcal{P}^p_q \Vdash \exists p.\delta_1$ iff $\mathcal{P}, \rho \Vdash \exists p.\delta_1$, as $\mathcal{P} \sim_p \sqrt{}^p_q$. But $\exists p.\delta_1$ is precisely $\lceil \exists p.\delta_1 \rceil^p_q$. The case when $\delta \equiv \exists q.\delta_1$ is forbidden, as q is not free in δ. Let z be another propositional symbol.

We need to prove that $\mathcal{P}^p_q, \rho \Vdash \exists z.\delta_1$ iff $\mathcal{P}, \rho \Vdash \exists z.\lceil \delta_1 \rceil^p_q$. This follows from Lemma 5.2.3. Note that $\mathcal{P}, \rho \Vdash \exists z.\lceil \delta_1 \rceil^p_q$ holds iff there exists $\mathcal{Q} \sim_z$ equivalent to $\mathcal{P}$ such that $\mathcal{Q}, \rho \Vdash \lceil \delta_1 \rceil^p_q$. By the induction hypothesis we obtain that $\mathcal{Q}^p_q, \rho \Vdash \delta_1$. Finally, by Lemma 5.2.3, $\mathcal{Q}^p_q$ is $\sim_z$ equivalent to $\mathcal{P}^p_q$ and as such $\mathcal{P}^p_q, \rho \Vdash \exists z.\delta_1$. □

We just remark that Hilbert's Axiom $\forall p.(\delta_1 \supset \delta_2) \supset (\delta_\supset(\forall p.\delta_2))$, when p is not a free variable of δ_1 is also a valid formula, however its proof is highly technical, and we will not need the result or any insight from its proof.

One important property that remains to be checked is whether the truth value of a formula depends in some way of propositional symbols not directly

used in that formula. Since in general the set of propositional symbols is infinite, if the truth value in some model over all the propositional symbols depended on unmentioned propositions, any attempt to prove the existence of quantifier elimination would be unfruitful.

We note that in fact it would be quite simple to prove that the truth value of δ only depends on the behavior of the EPPL model relative to $pvar(\delta)$, if we had stated that Λ was finite right from the start; we did not wish to make such a restriction as we believed the general result was provable, albeit harder, and therefore these technical lemmata are required.

This following lemma states an extremely important property of reducts that will be used in proving the crucial theorem that allows reasoning over finite probability models; a reduct over Π should capture all the information relevant to decide the truth value of any formula only written using propositional symbols from Π. This lemma will allow the extension of Lemma 2.2.1 to QEPPL formula.

Lemma 5.3.4. *Let $\delta \in$ QEPPL(Λ), and $\mathcal{P}$ be a EPPL model over Λ. Then the following sets of reduced models are equal:*

$$A = \{\mathcal{Q}|_{pvar(\delta)} : \mathcal{Q} \sim_p \mathcal{P}, \mathcal{Q} \in \mathfrak{M}(\textsf{EPPL}(\Lambda))\},$$

$$B = \{\mathcal{Q} \in \mathfrak{M}(\textsf{EPPL}(pvar(\delta))) : \mathcal{Q} \sim_p \mathcal{P}|_{pvar(\delta)}\}.$$

Proof. Let $\mathcal{P} = (V, \sigma(\{0,1\})^\Lambda, \nu)$, and its reduct $\mathcal{P}|_{pvar(\delta)} = (V', \mathcal{P}(\{0,1\}^\Pi), \nu')$; let $\mathcal{Q} = (W, \sigma(\{0,1\}^\Lambda, \mu)$ and its reduct $\mathcal{Q}|_{pvar(\delta)} = (W', \mathcal{P}(\{0,1\}^\Pi), \mu')$.

The inclusion $A \subseteq B$ is proved directly from the definition of reduct (Definition 2.2.5) and the definition of p-equivalence (Definition 5.2.1). Assume that $\mathcal{P} \sim_p \mathcal{Q}$. We need to show that $\mathcal{Q}|_{pvar(\delta)} \sim_p \mathcal{P}|_{pvar(\delta)}$.

The first condition in Definition 5.2.1 requires

$$\forall_{\Pi \subseteq pvar(\delta) - \{p\}} \forall_{v \in \{0,1\}^\Pi} \; B_v \cap V_1 \neq \emptyset \text{ iff } B_v \cap V_2 \neq \emptyset$$

. Let v be a valuation defined over $pvar(\delta) - \{p\}$; then since $\mathcal{P} \sim_p \mathcal{Q}$, we know that $B_v \cap P \neq \emptyset$ iff $B_v \cap Q \neq \emptyset$. By restricting the domain in both the cylinder and in the sample space, we obtain the desired result.

Since the model $\mathcal{Q}|_{pvar(\delta)}$ we are dealing with has probability with finite support, we just need to check that for any valuation $v \in \{0,1\}^{pvar(\delta)-p}$ the following equation holds:

$$\mu'(B_v \cap W') = \nu'(B_v \cap V|_{pvar(\delta)}).$$

By some direct application of definitions, we obtain that $A \subseteq B$:

$$\begin{aligned}
&\mu'(B_v \cap W') = && \text{since } \mu'(W') = 1\\
&= \mu'(B_v) = && \text{by Definition 2.2.5}\\
&= \mu(B_v) = && \text{since } \mu(W) = 1\\
&= \mu(B_v \cap W) = && \text{by definition of } \sim_p\\
&= \nu(B_v \cap V) = && \text{since } \nu(V) = 1\\
&= \nu(B_v) = && \text{by Definition 2.2.5}\\
&= \nu'(B_v) = && \text{since } \nu'(V') = 1\\
&= \nu'(B_v \cap V').
\end{aligned}$$

The proof of inclusion $A \supseteq B$ is more involved. Let $\mathcal{Q}' \in B$. Then, $\mathcal{Q}' \in \mathfrak{M}(\mathsf{EPPL}(pvar(\delta)))$ and $\mathcal{Q}' \sim_p \mathcal{P}|_{pvar(\delta)}$. We need to show that there exists a model $\mathcal{Q} \in \mathfrak{M}(\mathsf{EPPL}(\Lambda))$, such that $\mathcal{Q}'$ is in fact the reduct of $\mathcal{Q}$, and that the model $\mathcal{Q}$ is p-equivalent to $\mathcal{P}$. It should be noted however that there are many possible models $\mathcal{Q}$ obeying these two conditions.

This proof will be divided in two sections. First, we will construct $\mathcal{Q} = (W, \sigma(\{0,1\}^\Lambda), \nu) \in \mathfrak{M}(\mathsf{EPPL}(\Lambda))$; In order to do so, assume that $\mathcal{Q}' = (W', \sigma(\{0,1\}^\Lambda), \nu') \in \mathfrak{M}(\mathsf{EPPL}(pvar(\delta)))$ such that $\mathcal{Q}' \sim_p \mathcal{P}|_{pvar(\delta)}$, that is $\mathcal{Q}' \in B$, and let $\mathcal{P} = (V, \sigma(\{0,1\})^\Lambda, \mu)$.

Phase 1:

The sample space is defined to be $W = (V \cup \overline{V}^p) \cap \{w \in \{0,1\}^\Lambda : w|_{pvar(\delta)} \in W'\}$.

We then need to assign measure to the cylinders B_v, for any valuation $v \in \{0,1\}^{\Pi_j}$ and $\Pi_j \in \mathcal{P}_{fin}(\Lambda)$. Once we assign measure to all the cylinders, by Caratheodory's extension theorem, the probability space will be completely defined.

However, if $v \in \{0,1\}^{\Pi_j}$, where $q \notin \Pi_j$, $B_v = B_{v,q\mapsto 0} \cup B_{v,q\mapsto 1}$, so, in order to guarantee that we only assign measure to a certain cylinder once, we choose an ordering of the propositional symbols $i : \mathbb{N} \mapsto \Lambda$, and define only B_v for any valuation $v \in \{0,1\}^{\Pi_k}$, where $\Pi_0 = \{p\}$ and $\Pi_{k+1} = \Pi_k \cup \{i(k+1)\}$; we assume moreover, that $i(k) \in pvar(\delta)$, for $k \leq \#pvar(\delta)$. The measure will be recursively assigned.

Basis: For any valuation $v \in \{0,1\}^{\Pi_k}$ provided that $k \leq \#pvar(\delta)$, $\nu(B_v) = \nu'(B_v)$. Note that we obtain $\nu(W) = 1$.

Recursive Step: Assume that $\nu(B_v)$ is defined for any valuation $v \in \{0,1\}^{\Pi_k}$. Suppose that $i(k+1) = q$, and let v' be the restriction of $v \in \{0,1\}^{\Pi_k}$ to all propositional symbols in Π_k except p. Then consider the

following system of inequalities:

$$\nu(B_{p\mapsto 0,v'}) = \nu(B_{p\mapsto 0,v',q\mapsto 0}) + \nu(B_{p\mapsto 0,v',q\mapsto 1}), \quad (1)$$
$$\nu(B_{p\mapsto 1,v'}) = \nu(B_{p\mapsto 1,v',q\mapsto 0}) + \nu(B_{p\mapsto 1,v',q\mapsto 1}), \quad (2)$$
$$\mu(B_{v',q\mapsto 0}) = \nu(B_{p\mapsto 0,v',q\mapsto 0}) + \nu(B_{p\mapsto 1,v',q\mapsto 0}), \quad (3)$$
$$\mu(B_{v',q\mapsto 1}) = \nu(B_{p\mapsto 0,v',q\mapsto 1}) + \nu(B_{p\mapsto 1,v',q\mapsto 1}), \quad (4)$$
$$\nu(B_{p\mapsto i,v',q\mapsto j)}) \geq 0, i, j \in \{0,1\}.$$

Notice that the left-hand side of all equations is completely defined. In the case of the first two equations, both valuations belong to $\{0,1\}^{\Pi_k}$; the left-hand side of the last two equations involves the known model $\mathcal{P}$. Moreover, if we did not wish to preserve the $\sim_p$ equivalence between $\mathcal{P}$ and $\mathcal{Q}$, then we would only need the first two equations, which express the additivity of the measure for disjoint cylinders.

This system of inequalities always has at least one solution, in fact in non-degenerate cases it has a denumerable number of solutions since we are imposing that the measure of all cylinders should be an algebraic number. So in fact, $\mathcal{Q}$ can be constructed in multiple ways, by choosing in each inductive step one of the solutions of the inequalities.

Phase 2:

We now have to prove that $\mathcal{Q} = (W, \sigma(\{0,1\}^\Lambda), \mu)$ is a probability space. We start by noting that W is measureable in $\sigma(\{0,1\}^\Lambda)$. Recall that the sample set $W = (V \cup \overline{V}^p) \cap \{w \in \{0,1\}^\Lambda : w|_{pvar(\delta)} \in W'\}$; $(V \cup \overline{V}^p)$ is measureable and $\{w \in \{0,1\}^\Lambda : w|_{pvar(\delta)} \in W'\}$ is the union of a finite number of cylinders and as such W is measureable. Therefore, we only need to show that $\nu : \mathcal{B} \mapsto [0;1]$ is a pre-measure function, for some adequate choices in the inductive step.

The measure ν of any empty cylinders $B_v \cap W$ can easily seen to be 0. Furthermore, if the coarser cylinders are not empty, then the finer cylinders will also not be empty, due to the construction of W. Therefore, if a coarser cylinder is empty, then it has to have null measure, which allows us to conclude that the finer cylinders must also have null measure.

We just need to prove the σ-additivity of the function ν. However, note that additivity is directly obtained from the equations (1) and (2). Suppose now that:

$$B_v = \bigcup_{i \in \mathbb{N}_0} B_{v_i}.$$

Since the underlying topology is compact we know there is a subsequence $\{B_{v_{i_k}}\}_{k=1}^n$ such that $B_v = \bigcup_{k=1}^n B_{v_{i_k}}$. Therefore, additivity is enough to prove σ-additivity.

Afterwards we just need to check whether:

- $(W, \sigma(\{0,1\}^\Lambda), \nu)|_{pvar(\delta)} = \mathcal{Q}'$,
- $(W, \sigma(\{0,1\}^\Lambda), \nu) \sim_p \mathcal{P}$.

Both these two facts derive directly from the construction. The first fact is proven trivially using the basis step of the construction. Moreover, the second fact can be proved using the guarantee that the choices in the inductive step obey the equations (3) and (4), for any valuation $B_{v'}$, where $v' : \Pi \subset \Lambda$, with $p \notin \Pi$, and $\#\Pi < +\infty$. □

We now extend Lemma 2.2.1 to cope with propositional quantification.

Lemma 5.3.5 (Lemma of absent variables)**.** *Let $\delta \in \mathsf{QEPPL}(\Lambda)$ and $\mathcal{P} = (V, \sigma(\{0,1\}^\Lambda), \nu) \in \mathfrak{M}(\mathsf{EPPL}(\Lambda))$, and let $\rho : Z \mapsto \mathbb{R}_{\mathsf{Alg}}$. Then,*

$$\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\Lambda)} \delta \text{ iff } \mathcal{P}|_{pvar(\delta)}, \rho \Vdash^{\mathsf{QEPPL}(pvar(\delta))} \delta.$$

Proof. Let $\mathcal{P}|_{pvar(\delta)} = (V', \mathcal{P}(\{0,1\}^{pvar(\delta)}), \mu)$; the proof follows by structural induction in δ.

- δ is $(\Box\beta)$: then, $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\Lambda)} \Box\beta$ iff for all $v \in V$, $v \Vdash \beta$, iff for all $v \in V'$, $v \Vdash \beta$ iff $\mathcal{P}|_{pvar(\delta)}, \rho \Vdash^{\mathsf{QEPPL}(pvar(\delta))} \Box\beta$. In the second equivalence we use the fact that omitted variables are irrelevant in classical propositional logic.
- δ is $(t_1 \leq t_2)$: we show that $[\![t_1]\!]_{\mathcal{P},\rho} = [\![t_1]\!]_{\mathcal{P}|_{pvar(\delta),\rho}}$. The non-obvious case is when t_1 is $\int \beta$. Note that

$$\begin{aligned} nu(\{v \in V : v \Vdash \beta\}) &= \sum_{v \in \{0,1\}^{pvar(\delta)}, v \Vdash \beta} \nu(B_v \cap V) \\ &= \sum_{v \in \{0,1\}^{pvar(\delta)}, v \Vdash \beta} \nu'(B_v \cap V|_{pvar(\delta)}), \end{aligned}$$

  where the last equality is due to the definition of $\mathcal{P}|_{pvar(\delta)}$.

We now proceed to the induction step; both the $\cap$ and $\sim$ connectives are trivial applications of the induction hypothesis. However, in order to obtain the induction step for the $\exists p.$ connective, we require the use of Lemma 5.3.4.

- ($\Rightarrow$) Assume that $\mathcal{P} \Vdash^{\mathsf{QEPPL}(\Lambda)} \exists p.\delta_1$. Then there exists $\mathcal{P}'$ such that $\mathcal{P}' \sim_p \mathcal{P}$ and $\mathcal{P}' \Vdash^{\mathsf{QEPPL}(\Lambda)} \delta_1$. But then, by induction hypothesis, $\mathcal{P}'|_{pvar(\delta)} \Vdash^{\mathsf{QEPPL}(pvar(\delta))} \delta_1$. Now, $\mathcal{P}'|_{pvar(\delta)} \in A$, so $\mathcal{P}'|_{pvar(\delta)} \in B$, by Lemma 5.3.4 which immediately allows us to conclude that $\mathcal{P}|_{pvar(\delta)} \Vdash^{\mathsf{QEPPL}(pvar(\delta))} \exists p.\delta_1$.

- ($\Leftarrow$) Assume that $\mathcal{P}|_{pvar(\delta)} \Vdash^{\mathsf{QEPPL}(pvar(\delta)))} \exists p.\delta_1$. Then there exists $\mathcal{Q}' \in \mathfrak{M}(\mathsf{EPPL}(pvar(\delta)))$, $\mathcal{Q}' \sim_p \mathcal{P}|_{pvar(\delta)}$ such that $\mathcal{Q}' \Vdash^{\mathsf{QEPPL}(pvar(\delta))} \delta_1$. Then $\mathcal{Q}' \in B$, and as such $\mathcal{Q}' \in A$ by Lemma 5.3.4. Then there exists $\mathcal{Q} \in \mathfrak{M}(\mathsf{EPPL}(\Lambda))$ such that $\mathcal{Q}|_{pvar(\delta)} = \mathcal{Q}'$ and $\mathcal{Q} \sim_p \mathcal{P}$. So finally by using induction hypothesis we obtain $\mathcal{P} \Vdash^{\mathsf{QEPPL}(\Lambda)} \exists p.\delta_1$.

□

This theorem carries over for semantic entailment that is, if $\models^{\mathsf{QEPPL}(pvar(\delta))} \delta$ then $\models^{\mathsf{QEPPL}(\Pi)} \delta$ for any $\Pi \supseteq pvar(\gamma)$.

5.4 Elimination of quantifiers in QEPPL

The technique used here to prove quantifier elimination is essentially based on showing that propositional quantification can be reduced to appropriate quantification over the real numbers, using the underlying real closed field theory. Indeed, what we will show is that searching for a $\sim_p$ equivalent model satisfying a EPPL formula γ reduces to finding some algebraic numbers that assign measure to appropriate cylinders. The idea is to extend Lemma 2.2.2 in order to cope with the propositional quantifiers. Afterwards, we can use quantifier elimination over the real closed fields to obtain an equivalent fomula without quantifiers. However, we will still require an additional procedure in order to translate the formula to an equivalent EPPL formula.

First, we start by showing that both global and probabilistic reasoning of EPPL can be done using only algebraic equalities/inequalities by enriching the assignment ρ and transforming the EPPL formula accordingly. Recall that an algebraic formula is a first-order formula constructed using the signature of ordered fields. Thanks to **Tarski**'s result on quantifier elimination, it is known that any formula constructed with this signature can be decided in the theory of real closed fields.

Lemma 5.4.1. *Let $\gamma \equiv \exists p.\gamma_1$, where γ_1 is a EPPL formula and let $\mathcal{P} = (V, \mathcal{P}(\{0,1\}^{pvar(\gamma)}), \nu)$ be a EPPL$(pvar(\gamma))$ model. Furthermore, let $\rho : Z \mapsto$*

$\mathbb{R}_{\mathsf{Alg}}$. *Then, there exists an elementary algebraic formula* δ^γ *in prenex normal formula with only existential quantifiers and an assignment of variables* $\rho^\gamma_{\mathcal{P}}$ *extending* ρ*, such that:*

$$\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \exists p.\gamma_1 \text{ iff } \rho^\gamma_{\mathcal{P}} \Vdash^{\mathsf{EAlg}} \delta^\gamma.$$

Proof. We have to provide a slightly more convoluted translation for γ than the one given in Lemma 2.2.2. First, we assume that γ_1 is a conjunction of global logic formulae $\Diamond\beta$ and $\sim\Diamond\beta$, and comparison terms $t_1 \leq t_2$ and $t_1 > t_2$. This assumption can be done without loss of generality since $\exists p.(\delta_1 \cup \delta_2) \Leftrightarrow (\exists p.\delta_1) \cup (\exists p.\delta_2)$. Like in Lemma 2.2.2, we also assume that all the classical propositional formulae β are given in disjunctive normal form where each conjunct has $\#(pvar(\gamma))$ elements, such that the expression for β is the disjunction of formulae representing the valuations over $pvar(\gamma)$ that satisfy β. Furthemore, recall the translation $(.)^-$ defined in Lemma 2.2.2.

The translation operator is represented as $(.)^*$ and the translation of γ is defined as: $(\gamma)^* \equiv \exists x'_{v_i} \ldots \exists x'_{\nu(v_i)} (\gamma_1)^- \wedge \gamma_2 \wedge \gamma_3$, where in the translation of γ_1 the fresh variables introduced are $x'_{v_i}, \ldots, x'_{\nu(v_i)}$. The auxiliary formulae γ_2 and γ_3 are defined as follows:

γ_2 – For all valuations v over $pvar(\gamma)$, γ_2 is the conjunction of the following equations:

- $[(x_v = 1) \vee (x_{\overline{v}^p} = 1)] \Leftrightarrow [(x'_v = 1) \vee (x'_{\overline{v}^p} = 1)]$,
- $x_v x_{\nu(v)} + x_{\overline{v}^p} x_{\nu(\overline{v}^p)} = x'_v x'_{\nu(v)} + x'_{\overline{v}^p} x'_{\nu(\overline{v}^p)}$.

γ_3 – For all valuations over $pvar(\gamma)$, γ_3 is the conjunction of the following equations together with $\sum_v x'_{\nu(v)} = 1$:

- $x'_v = 1 \vee x'_v = 0$,
- $0 \leq x'_{\nu(v)}$,
- $x'_v = 0 \Rightarrow x'_{\nu(v)} = 0$.

Intuitively, γ_2 expresses the equivalence relation between two EPPL models; the introduced variables x' represent a model satisfying γ_1 and so the introduced variables x represent a model that is $\sim_p$-equivalent to the model satisfying γ_1. The assignment $\rho^\gamma_{\mathcal{P}}$ is built as in the proof of Lemma 2.2.2.

Now let $\delta^\gamma = (\gamma)^*$. We start by proving that if $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \exists p.\gamma_1$ then $\rho^\gamma_{\mathcal{P}} \Vdash^{\mathsf{EAlg}} \delta^\gamma$.

If $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \exists p.\gamma_1$, there exists a EPPL model $\mathcal{P}' \in \mathfrak{M}(\mathsf{EPPL}(pvar(\gamma)))$, p-equivalent to $\mathcal{P}$ such that $\mathcal{P}', \rho \Vdash^{\mathsf{EPPL}} \gamma_1$. Let $\mathcal{P}' = (V', \mathcal{P}(\{0,1\}^{pvar(\gamma)}), \nu')$.

By Lemma 2.2.2, we know that there exists an elementary algebraic formula $\delta^{\gamma_1} \equiv (\gamma_1)^-$ without quantifiers and an assignment $\rho^{\gamma_1}_{\mathcal{P}'}$, representing $\mathcal{P}'$ and ρ, such that $\rho^{\gamma_1}_{\mathcal{P}'} \Vdash^{\mathsf{EAlg}} (\gamma_1)^-$.

Next, we consider an assignment $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'}$ over $x_{v_i}, \ldots, x_{\nu(v_i)}$ and $x'_{v_i}, \ldots, x'_{\nu(v_i)}$ that captures the information about the two probability structures, namely: (i) $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'}(x_v) = 1$ if the valuation $v \in V$ in V being 0 otherwise; (ii) $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'}(x_{\nu(v)})$ is the measure of $\nu(\{v\})$ in V; (iii) mutatis mutandis for x'_v and $x'_{\nu(v)}$, and, finally, (iv) $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'}(x) = \rho(x) = \rho^{\gamma_1}_{\mathcal{V}'}(x)$.

Now, we show that if $\mathcal{P}$ is p-equivalent to $\mathcal{P}'$ and $\mathcal{P}', \rho \Vdash^{\mathsf{EPPL}} \gamma_1$ then $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'} \Vdash^{\mathsf{EAlg}} (\gamma_1)^- \wedge \gamma_2 \wedge \gamma_3$:

1. By Lemma 2.2.2, $\rho^{\gamma_1}_{\mathcal{P}'} \Vdash^{\mathsf{EAlg}} (\gamma_1)^-$ and so, $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'} \Vdash^{\mathsf{EAlg}} (\gamma_1)^-$;

2. By construction of $\rho^{\gamma_1}_{\mathcal{P}'}$, we have that $\sum_v x'_{\nu(v)} = 1$, $x'_v = 1 \vee x'_v = 0$ and $0 \leq x'_{\nu(v)}$ are satisfied by $\rho^{\gamma_1}_{\mathcal{P}'}$, since $\rho^{\gamma_1}_{\mathcal{P}'}$ represents a probability space. Given that, γ_3 is the conjunction of these inequations, we have that $\rho^{\gamma_1}_{\mathcal{P}'} \Vdash^{\mathsf{EAlg}} \gamma_3$ and so, $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'} \Vdash^{\mathsf{EAlg}} \gamma_3$;

3. Consider a valuation w over $pvar(\gamma) \setminus \{p\}$; assume that $B_w \cap V$ is not empty and so $B_w \cap V'$ is also not empty. This relation is precisely expressed by the equation $[(x_v = 1) \vee (x_{\overline{v}^p} = 1)] \Leftrightarrow [(x'_v = 1) \vee (x'_{\overline{v}^p} = 1)]$. Furthermore, $\nu(B_w \cap V)) = \nu((B_v \cup B_{\overline{v}^p}) \cap V) = \nu'((B_v \cup B_{\overline{v}^p}) \cap V')$, by $\sim_p$ equivalence, which can also be translated to $x'_v x'_{\nu(v)} + x'_{\overline{v}^p} x'_{\nu(\overline{v}^p)} = x_v x_{\nu(v)} + x_{\overline{v}^p} x_{\nu(\overline{v}^p)}$. We can then conclude that $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'}$ has to satisfy γ_2, because of the $\sim_p$ equivalence between $\mathcal{P}$ and $\mathcal{P}'$.

We summarize the proof until now as follows: if $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(pvar(\gamma))} \exists p.\gamma_1$, then there exists $\mathcal{P}', \rho \Vdash^{\mathsf{EPPL}(pvar(\gamma))} \gamma_1$, with $\mathcal{P}', \rho \sim_p$ equivalent to $\mathcal{P}, \rho$. So, $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'} \Vdash^{\mathsf{EAlg}} (\gamma_1)^- \wedge \gamma_2 \wedge \gamma_3$; then, the restriction of $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'}$ to the variables of ρ and the unmarked added variables x_v and $x_{\nu(v)}$ is precisely $\rho^{\gamma}_{\mathcal{P}}$, and so $\rho^{\gamma}_{\mathcal{P}} \Vdash^{\mathsf{EAlg}} \delta^{\gamma}$.

It remains to show the converse. Assume $\rho^{\gamma}_{\mathcal{P}} \Vdash^{\mathsf{EAlg}} \delta^{\gamma}$, we need to prove that $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}} \exists p.\gamma_1$ or, equivalently, that there exists a EPPL model $\mathcal{P}' \in \mathfrak{M}(\mathsf{EPPL}(pvar(\gamma)))$ p-equivalent to $\mathcal{P}$ such that $(\mathcal{P}', \rho) \Vdash^{\mathsf{EPPL}} \gamma_1$.

If $\rho^{\gamma}_{\mathcal{P}} \Vdash^{\mathsf{EAlg}} \delta^{\gamma}$, then there exists an extension $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'}$ over the real variables initially quantified $(\{x'_v\}, \{x'_{\nu(v)}\})$ such that $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'} \Vdash^{\mathsf{EAlg}} (\gamma_1)^- \wedge \gamma_2 \wedge \gamma_3$. Restrict $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'}$ to the assignment ρ and the marked variables x'_v and $x'_{\nu(v)}$ and name it $\rho^{\gamma}_{\mathcal{P}'}$. We can use this assignment to construct a model and an assignment $\mathcal{P}', \rho$ in the following way:

- the sample space is given by $V' = \{v \in \{0,1\}^{pvar(\gamma)}, \rho^{\gamma}_{\mathcal{P}'}(x_v) = 1\}$;

- the measure function $\nu' : \sigma(\{0,1\}^{\Lambda}) \mapsto [0,1]$ is constructed by assigning measure to the singletons, as in $\nu'(\{v\}) = \rho^{\gamma}_{\mathcal{P}'}(x_{\nu(v)})$;
- $\rho(x) = \rho^{\gamma}_{\mathcal{P}'}(x)$.

First, notice that in fact $\mathcal{P}' \in \mathfrak{M}(\mathsf{EPPL}(pvar(\gamma)))$. This is easily done by using the fact that $\rho^{\gamma}_{\mathcal{P}'} \Vdash^{\mathsf{EAlg}} \gamma_3$. Moreover, notice that $(\mathcal{P}', \rho) \Vdash^{\mathsf{EPPL}} \gamma_1$ by Lemma 2.2.2 and using the fact that $\rho^{\gamma}_{\mathcal{P}'} \Vdash^{\mathsf{EAlg}} (\gamma_1)^*$. Finally, we need to show that $\mathcal{P}' \sim_p \mathcal{P}$. This follows directly from the fact that $\rho^{\gamma}_{\mathcal{P},\mathcal{P}'} \Vdash^{\mathsf{EAlg}} \gamma_2$.

Then, we conclude that in fact there exists a EPPL model $\mathcal{P}' \sim_p \mathcal{P}$ such that $\mathcal{P}', \rho \Vdash^{\mathsf{EPPL}(pvar(\gamma))} \gamma_1$. □

Remark 5.4.1. If we had chosen to prove this theorem directly for infinite models, the construction of $\mathcal{P}, \rho$ using the assignment $\rho^{\gamma}_{\mathcal{P}'}$ would have to contain the reasoning present in Lemma 5.3.5, as the assignment could only contain information about a finite number of cylinders, which we would then have to expand in order to obtain a model over all propositional symbols in Λ.

We know that due to the elimination of quantifiers in real closed fields, there exists a formula equivalent to δ^{γ} without any quantifiers. Unfortunately, the structure of this formula may be quite complex and as such, before being able to translate it back to a EPPL formula, we need some additional modifications. Those modifications will be described in the following lemma; we are then able to separate variables corresponding to global reasoning from those corresponding to probabilistic terms.

Lemma 5.4.2. *Consider $\gamma \equiv \exists p.\gamma_1$, where $\gamma_1 \in \mathsf{EPPL}(pvar(\gamma))$. Then δ^{γ}, as defined in Lemma 5.4.1, is equivalent to τ^{γ} given below:*

$$\tau^{\gamma} \equiv \bigvee_{V \subseteq pvar(\gamma)} \phi_V \wedge \psi(\mathbf{x}, [\mathbf{x_v}]^V, \mathbf{x}_{\nu(\mathbf{v})})$$

where:

- $\phi_V = (\bigwedge_{v \in V} x_v = 1) \wedge (\bigwedge_{v \notin V} x_v = 0)$*;*
- $\psi(\mathbf{x}, \mathbf{x_v}, \mathbf{x}_{\nu(\mathbf{v})})$ *is a first-order formula over the signature of ordered real closed fields without quantifiers, with $\mathbf{x}, \mathbf{x_v}, \mathbf{x}_{\nu(\mathbf{v})}$ vectors of algebraic variables, equivalent to δ^{γ};*
- $[\mathbf{x_v}]^V$ *is the replacement of each variable x_v by 1 if $v \in V$, 0 otherwise.*

Proof. Using the elimination of quantifiers in Real Closed Fields, we know that $\delta^\gamma \Leftrightarrow \psi(\mathbf{x}, \mathbf{x_v}, \mathbf{x}_{\nu(\mathbf{v})})$. Since γ_3 is a conjunct in δ^γ, it enforces that x_v should be either 1 or 0. The equivalence follows easily using this fact. $\square$

We are now able to prove the existence of quantifier elimination in QEPPL by the traditional method of first eliminating a single existential quantifier.

Proposition 5.4.1. *Consider* $\gamma \equiv \exists p.\gamma_1$, *where* $\gamma_1 \in \mathit{EPPL}(pvar(\gamma))$. *Then, there exists* $\alpha \in \mathit{EPPL}(pvar(\gamma))$ *such that*

$$(\mathcal{P}, \rho) \Vdash^{\mathit{QEPPL}(pvar(\gamma))} \gamma \text{ iff } (\mathcal{P}, \rho) \Vdash^{\mathit{EPPL}(pvar(\gamma))} \alpha.$$

Proof. Using Lemmas 5.4.1, 5.4.2, we know that $(\mathcal{P}, \rho) \Vdash^{\mathsf{QEPPL}(pvar(\gamma))} \gamma$ iff $\rho_V^\gamma \Vdash^{\mathsf{EAlg}} \tau_\gamma$. So, if one sees τ^γ as a EPPL formula, we immediately know that $(\mathcal{P}, \rho_V^\gamma) \Vdash^{\mathsf{QEPPL}(pvar(\gamma))} \tau^\gamma$. We then need to translate back the extra variables x_{v_i} and $x_{\nu(v_i)}$, eliminating the need for the extra assignments. The translation we will apply to τ^γ is represented as $(.)^o$. As seen above in Lemma 5.4.2,

$$\tau_\gamma \equiv \bigvee_{V \subseteq pvar(\gamma)} \phi_V \wedge \psi(\mathbf{x}, [\mathbf{x_v}]^V, \mathbf{x}_{\nu(\mathbf{v})}).$$

The translation $(.)^o$ is applied as follows to the ϕ_V conjunct:

- $(x_v = 1)^o \equiv \Diamond \beta_v$,
- $(x_v = 0)^o \equiv \sim \Diamond \beta_v$,
- $(\gamma_1 \wedge \gamma_2)^o \equiv (\gamma_1)^o \cap (\gamma_2)^o$.

Regarding the $\psi(\mathbf{x}, [\mathbf{x_v}]^V, \mathbf{x}_{\nu(\mathbf{v})})$ conjunct, the translation is defined as follows:

- $(x)^o \equiv x$,
- $(x_{\nu(v)})^o \equiv \int \beta_v$,
- $(r)^o \equiv r$,
- $(\gamma_1 + \gamma_2)^o \equiv (\gamma_1)^o + (\gamma_2)^o$,
- $(\gamma_1 \gamma_2)^o \equiv (\gamma_1)^o (\gamma_2)^o$,
- $(\gamma_1 \leq \gamma_2)^o \equiv (\gamma_1)^o \leq (\gamma_2)^o$,
- $(\gamma_1 > \gamma_2)^o \equiv (\gamma_1)^o > (\gamma_2)^o$,

- $(\neg\gamma)^o \equiv \sim(\gamma)^o$,
- $(\gamma_1 \wedge \gamma_2)^o \equiv (\gamma_1)^o \cap (\gamma_2)^o$,
- $(\gamma_1 \vee \gamma_2)^o \equiv (\gamma_1)^o \cup (\gamma_2)^o$.

It is easy to see that $(\tau_\gamma)^o \in \mathsf{EPPL}(pvar(\gamma))$. □

Theorem 5.4.1. *Consider $\delta \in \mathsf{QEPPL}(\Lambda)$. Then, there exists a formula $\alpha \in \mathsf{EPPL}(fpvar(\delta))$ such that $\models^{\mathsf{QEPPL}(\Lambda)} \delta \Leftrightarrow \alpha$.*

Proof. This proof is done by induction on the construction of δ. The basis cases are straightforward:

- δ is $\Diamond\beta$: set $\alpha = \Diamond\beta$. Clearly $\models^{\mathsf{QEPPL}(\Lambda)} \delta \Leftrightarrow \alpha$.
- δ is $(t_1 \leq t_2)$: set $\alpha = (t_1 \leq t_2)$, then $\models^{\mathsf{QEPPL}(\Lambda)} \delta \Leftrightarrow \alpha$.

The induction step is also straightforward, requiring the application of Lemmas 5.3.1, 5.3.2, 5.3.5, and Proposition 5.4.

- δ is $\delta_1 \cap \delta_2$: then by the induction hypothesis we know that there exists $\alpha_1, \alpha_2 \in \mathsf{EPPL}(fpvar(\delta))$ such that $\models^{\mathsf{QEPPL}(\Lambda)} \delta_i \Leftrightarrow \alpha_i$. By substitution of equivalents from Lemma 5.3.1 we obtain $\models^{\mathsf{QEPPL}(\Lambda)} \delta \Leftrightarrow (\alpha_1 \cap \alpha_2)$.
- δ is $\sim \delta_1$: again, by the induction hypothesis we know that there exists $\alpha_1 \in \mathsf{EPPL}(fpvar(\delta))$ such that $\models^{\mathsf{QEPPL}(\Lambda)} \delta_1 \Leftrightarrow \alpha_1$. Set $\alpha = \sim \alpha_1$. We then obtain $\models^{\mathsf{QEPPL}(\Lambda)} \delta \Leftrightarrow \alpha$.
- δ is $\exists p.\delta_1$: by the induction hypothesis we know that there exists $\alpha_1 \in \mathsf{EPPL}(fpvar(\delta))$ such that $\models^{\mathsf{QEPPL}(\Lambda)} \delta_1 \Leftrightarrow \alpha_1$. By substitution of equivalents, $\models^{\mathsf{QEPPL}(\Lambda)} \exists p.\delta_1 \Leftrightarrow \exists p.\alpha_1$. Let $\mathcal{P} \in \mathfrak{M}(\mathsf{EPPL}(\Lambda))$ such that $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\Lambda)} \exists p.\delta_1$. Then:

  $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\Lambda)} \exists p.\alpha_1$ iff, by Lemma 5.3.5
  $\mathcal{P}|_{pvar(\delta)}, \rho \Vdash^{\mathsf{QEPPL}(pvar(\delta))} \exists p.\alpha_1$ iff, by Proposition 5.4
  $\mathcal{P}|_{pvar(\delta)}, \rho \Vdash^{\mathsf{QEPPL}(pvar(\delta))} \alpha'$ iff, by Lemma 5.3.2
  $\mathcal{P}, \rho \Vdash^{\mathsf{QEPPL}(\Lambda)} \alpha'$

□

5.5 Conclusions and future work

We have shown that propositional quantification in EPPL can be eliminated and so, there is no added expressiveness. To this end, we constructed the most natural p-equivalent relation between models, we showed that the logic is able to express this equivalence relation. While the result itself carries theoretical interest, we expect to apply it beyond temporal probabilistic extensions, but also to logics for quantum reasoning. Moreover, the natural definition of p-equivalence between quantum models specially is not straightforward. We know that while the introduction of quantifiers in EPPL can be removed, we are certain that the quantified version of temporal EPPL will not have elimination of quantifiers, due to the difference between LTL and QLTL. We would also like to obtain (tighter) bounds for the quantifier elimination algorithm in both QEPPL and the global fragment of EPPL.

Finally, we would wish to exemplify the capabilities inherent to propositional quantification in the analysis of circuits. Let $f : \{0,1\}^{\Pi} \mapsto \{0,1\}^{\Sigma}$ with $\pi \cap \Sigma = \emptyset$, described by a certain propositional formula ϕ_f over $\Pi \cup \Sigma$. Suppose that we wish to build a circuit that approximates f up to a certain threshold ϵ, using as gates $\wedge, \vee, \neg$. However these gates are not completely reliable, as they have a failure rate δ. Moreover, we will not assume that the failure of each gate is independent: for instance, if r is the output bit of a negation of q, which is itself a negation of p, we will not assume that $\int(p \leftrightarrow r) = 1 - \delta^2$. Let C be some circuit that we propose as an approximation of f, represented as a propositional formula ϕ_C over $\Pi \cup \Sigma \cup AuxVar$, where $AuxVar$ represent the auxiliary variables required to model the inner gates.

The property representing the fact C approximates the function f can be written in EPPL as:

$$\int \phi_C \leftrightarrow \phi_f \geq 1 - \epsilon$$

If one writes some formulae describing the failure rate of each gate, then we wish to establish that $\Gamma \models \int \phi_C \leftrightarrow \phi_f \geq 1 - \epsilon$.

Suppose now that an adversary is able to control two certain inner gates of the circuit C with output propositional symbols p_1 and p_2, however there exists an inner gate q_1 that is adjustable by the user of the circuit, in the presence of unwarranted access. Then, propositional quantification comes to our aid when modeling this scenario:

$$\forall p_1, p_2 . \exists q_1 . \int \phi_C \leftrightarrow \phi_f \geq 1 - \epsilon$$

The fact that both the circuit and the function support this type of adaptability is captured by $\Gamma \models \forall p_1, p_2 \exists q_1 \int \phi_C \leftrightarrow \phi_f \geq 1 - \epsilon$. It is in this sense that propositional quantification allows us to analyze the robustness of a system. Suppose that we would be able to prove $\Gamma \models \exists q_1 \forall p_1 \forall p_2 \int \phi_C \leftrightarrow \phi_f \geq 1 - \epsilon$. Then the circuit would even be more reliable as in fact the settings for the inner gate q_1 could be adjusted a priori without knowing what happened in gates p_1 and p_2.

We would also like to point out the logic presented in [80] is closely related to EPPL and demonstrates how these exogenous logics can be adapted to reasoning about security protocols.

Chapter 6

Temporalizations of EPPL

This chapter concerns a conference paper developed by myself, Pedro Baltazar, Paulo Mateus and Rajagopal Nagarajan and presented in the FLAIRS Conference in St. Pete's Beach in 2013, studies and proposes an application for temporalizations of Exogenous Probabilistic Propositional Logic. Instead of considering a temporalization of EPPL akin to the one proposed by **Finger et al.** [40], we consider a temporalization based on the μ-Calculus, named μ-EPPL. Given that the μ-Calculus extends CTL*, a satisfiability procedure for μ-EPPL automatically gives us procedures for EPCTL and EPLTL, which are the restrictions of μ-EPPL to the temporal operators of CTL and LTL, respectively.

This work was already partially presented in **Baltazar** PhD thesis [8], namely the satisfiability procedures for EPLTL and EPCTL, and so we focus on adapting the result to cope with the μ-calculus. Since the paper was presented in the Florida Artificial Intelligence Research Society conference, it has a strong focus on the application of the satisfiability procedures to planning problems in Artificial Intelligence. Most of the logics used in planning in AI are based on propositional logic, for both state and temporal properties. Our hope would be to present a specification language more expressive than propositional logic that can capitalize on the optimized algorithms developed by the Temporal Logic community in order to solve more complex planning problems using realistic resources.

While on the original paper we devoted the preliminaries to the presentation of both EPPL and μ-TL , we have already introduced them in this work and as such, we will simply reference the definitions and results needed.

6.1 Introduction

One of the most fruitful techniques to solve the planning problem in AI has been by reduction to the satisfiability problem over classical propositional logic [59, 91]. Classical propositional logic does not address the problem of uncertainty and by considering a bounded time horizon one is able to reason and plan efficiently by using propositional symbols representing time. However, the main reason for the efficiency of this type of planning is the advanced SAT algorithms already in existence [91].

In this work, we use a logic that not only allows one to reason about quantities but also about probabilities, named Exogenous Probabilistic Propositional Logic (EPPL) [76, 78]. The term *exogenous* was coined by **Kozen** [67] to express the fact that the probabilities had an explicit syntax and were not hidden in the propositional symbols or connectives. Here we introduce a temporalization of this logic (μ-calculus extension of EPPL). Although quite expensive computationally, it has a relevant sublogic, Exogenous Probabilistic Linear Time Logic (EPLTL), which, as we show, has a PSPACE SAT algorithm. With these temporal logics at our disposal, and with the efficient algorithms already known to solve the Satisfiability problem of LTL [104], we are able to specify properties in a way relatively close to natural language. In this way we can produce plans for desired behaviors of agents, taking into account uncertainty. We also present Hilbert calculi for both logics, allowing us to derive that any satisfying plan has certain additional properties.

Temporal logics have a component to reason about states, called the state logic, and temporal modalities to reason about the evolution of states. The probabilistic state logic considered, EPPL, is akin to the logic proposed in [38] with some slight modifications. For the temporal modalites we use the most powerful decidable language, the μ-calculus; and the result is a fixpoint logic whose SAT algorithm allows us to output plans. Other approaches to reasoning about time and probability exist. However, they either have an undecidable SAT problem, such as the probabilistic situation calculus [75], or their SAT problem is not yet solved (for instance pCTL [27, 20]).

There have been three different approaches to planning under uncertainty: full probabilistic [62], contingent probabilistic [18] and conformant probabilistic [19]. The difference between these approaches is the observation model considered. In full probabilistic planning, we assume that the observations are total (but random); in contingent probabilistic, the random observations are partial; and finally, in conformant probabilistic planning, no observations exist (although the underlying behavior is random). The SAT

method proposed can be adopted to all the cases, although the specification has to be significantly different for each case. Many algorithms dedicated to each type of probabilistic planning exist, and are in general more restrictive than using a EPLTL specification. However, these are in general much more efficient, in most cases, because intelligent heuristics are used.

This paper is structured as follows. First, we begin by motivating our intended applications by a simple example. Then, we present the syntax, semantics and axiomatization of the state logic EPPL. After that, we introduce the fixpoint extension of exogenous probabilistic logic, μ-EPPL. Then, we address completeness for μ-EPPL and, finally, we revisit the example, formalizing the constraints given in the first section.

The central contribution of this paper is the introduction of a new logic μ-EPPL for planning under uncertainty, which enables us to reason with time explicitly. In particular, we are able to use the fact that μ-EPPL has a sublogic for which the SAT algorithm relies on the SAT problem of LTL, which has been thoroughly studied. The associated complexity results and algorithms are also completely new.

6.2 Motivating Example

Assume that a robot *Healer 2.0* has been built and it comes equipped with advanced medical equipment, previously tested in its earlier iteration. This time, however, it is capable of locomotion. Its intended use is in deserted locations, where it can be deployed far from its target, and then rescue its intended target. Unfortunately, the advanced medical equipment occupies a large amount of space, and *Healer 2.0* cannot transport much fuel. The cost of the deployment of the supplies is limited by a threshold; moreover this cost grows quadratically with the distance from the base.

The developers are interested in making a plan containing the robot movements on the grid (requires temporal reasoning), and on how to distribute the supplies along the grid (requires quantitative reasoning).

6.3 A Probabilistic State Logic

Exogenous Probabilistic Propositional Logic was already introduced in Chapter 2, Section 2.2. Both the syntax and semantics used in this Chapter follow Definitions 2.2.1, 2.2.4; respectively. EPPL formulae are used to describe

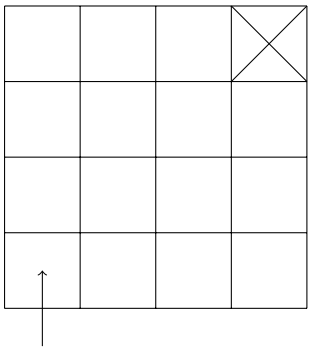


Figure 6.1: A grid illustrating the possible movements of *Healer 2.0*, with entry point and goal marked

properties of states, as PL formulae are used to describe properties of state in classical temporal logic.

6.4 μ-calculus extension of Exogenous Probabilistic Logic

We now introduce a μ-calculus extension of EPPL by introducing the fixpoint operators [65]. We also provide a sound and (weakly-) complete proof system by enriching the μ-TL proof system [105] with the axioms of EPPL. We will present full μ-EPPL, providing a complete Hilbert calculus and a SAT algorithm.

This temporal extension can be seen as the most general one, subsuming most of the common temporal logics like LTL or CTL. In LTL, time evolves deterministically, as a sequence of instants. In CTL, time branches and behaves like a tree of instants.

6.4.1 Propositional μ-calculus

The propositional μ-calculus was already introduced in Section 1.1.2. However, we make a small adaptation to the semantics of μ-calculus. Recall that the semantics for the μ-calculus is usually defined for multi-modal Kripke structures. In our case, as we are using μ-TL as a temporal logic, we simply consider that the set of actions is a singleton, and as such a multi-modal Kripke structure collapses to a classical Kripke structure.

We recall that μ-TL enjoys a sound and weakly complete axiomatization [105], and the proof system $HC^{\mu\text{-TL}}$ is given in Table 1.1.

6.4.2 Syntax

The syntax of μ-EPPL is formed by enriching the μ-TL by taking as propositional symbols the global atoms of EPPL. The syntax is detailed below:

$$\beta := \alpha \quad | \quad (\neg\beta) \quad | \quad (\beta \Rightarrow \beta)$$

$$t := z \quad | \quad r \quad | \quad \int \beta \quad | \quad (t + t) \quad | \quad (t.t)$$

$$\varphi := (\Box\beta) \quad | \quad (t \leq t) \quad | \quad \xi \quad | \quad (\sim \varphi) \quad | \quad (\delta \supset \varphi) \quad | \quad (\Diamond\varphi) \quad | \quad (\mu\xi.\varphi)$$

where $\alpha \in \Lambda$, $z \in Z$, $\xi \in \Xi$, and ξ appears under an even number of negations in $\mu\xi.\varphi$.

The basic formulae and probabilistic terms have the same intuitive meaning as in EPPL. Similarly, the global formulae with fixpoint operators have the same meaning as in μ-TL. Recall that $\Box\beta$ is an assertion concerning a single probabilistic space, describing the possible propositional valuations in the sample space, while $\Diamond\varphi$ is already a temporal assertion.

6.4.3 Semantics

In order to provide semantics for the logic, we introduce a very simple notion of Kripke structure over EPPL models. An μ-EPPL structure, or a EPPL-Kripke structure, consists of a tuple $m = (S, R, L)$ where S is a non-empty set of states, $R \subseteq S \times S$ is a total relation, and L is a map that assigns an EPPL model (including an assignment $\rho : Z \mapsto \mathbb{R}_{\mathsf{Alg}}$ to the algebraic variables) to each state in S. Our models are closely related to the models of probability and knowledge in [37]. It is easy to see that we are considering a Kripke-structure adaptation of Definition 3.3.2.

The μ-EPPL semantics mimics the semantics of the μ-TL. In a Kripke structure, L maps each state to a set of propositional symbols (or equivalently to a valuation), that is a propositional model. For our EPPL-Kripke structures, L maps each state to a EPPL model; EPPL-Kripke structures allow to model probabilistic programs, and protocols.

We now present the formal semantics of μ-EPPL. For this purpose, we denote by $[\delta]^M_V$ the set of states of M that satisfy δ given valuation $V : \Xi \to 2^S$. The set $[\delta]^m_V$ is defined inductively on the structure of δ as follows:

- $[(\Box\beta)]^m_V = \{s \in S : L(s) \Vdash^{\mathsf{EPPL}} (\Box\beta)\}$;

- $[t_1 \leq t_2]^m_V = \{s \in S : L(s) \Vdash_{\mathsf{EPPL}} (t_1 \leq t_2)\}$;

- $[\xi]^m_V = V(\alpha)$;

- $[(\sim \varphi)]^m_V = S \setminus [\varphi]^m_V$;

- $[(\varphi_1 \supset \varphi_2)]^m_V = [\sim \varphi_1]^m_V \cup [\varphi_2]^m_V$;

- $[(\Diamond\varphi)]^m_V = \{s \in S :$ exists $s' \in S$, such that $(s, s') \in R$, and $s' \in [\varphi]^m_V\}$;

- $[\mu\xi.\varphi]^m_V = \bigcap\{S' \subseteq S : [\varphi]^m_{V[\xi \leftarrow S']} \subseteq S'\}$;

  where $V[\xi \leftarrow S']$, as before, denotes the valuation V' that may just differ from V by $V'(\xi) = S'$.

Given a closed formula φ, if $s \in [\varphi]^m_V$ for some valuation V then we write $m, s \Vdash^{\mu\text{-}\mathsf{EPPL}} \varphi$.

6.4.4 Sound and Weakly Complete Hilbert calculus

In this section we provide a complete calculus for μ-EPPL. Completeness is obtained by using the completeness of the μ-TL and EPPL. We give the complete axiomatization $HC^{\mu\text{-}\mathsf{EPPL}}$ of μ-EPPL in Table 6.1 and in the rest of the section we focus on the proof of completeness.

In order to obtain completeness, we need to translate a μ-EPPL formula into a formula of μ-TL. Consider a bijective function $(.)^b$ that translates EPPL atomic formulae $(t_1 \leq t_2)$ and $(\Box\beta)$ to a propositional constant in Γ. Using the function $(.)^b$ we can translate μ-EPPL formulae to μ-TL formulae by preserving connectives.

As an example, consider the formula $\varphi := ((\Box\beta) \supset (\Diamond(\int \beta \leq 0)))$. We have $\varphi^b := (\tau_1 \Rightarrow (\Diamond\tau_2))$.

Observe that with the bijection $(\_)^b$ we can also translate a EPPL-Kripke structure to a Kripke structure. Indeed, a EPPL-Kripke structure $m =$

[**Axioms**]

[**Ax1**]	all EPPL tautologies
[**Ax2**]	all instantiations of $HC^{\mu\text{-TL}}$ tautologies with μ-EPPL formulae

[**Rules of Inference**]

[**R1**]	$\varphi_1, (\varphi_1 \supset \varphi_2) \vdash^{\mu\text{-EPPL}} \varphi_2$
[**R2**]	$(\varphi_1[\xi \leftarrow \varphi_2] \supset \varphi_2) \vdash^{\mu\text{-EPPL}} (\mu\xi.(\varphi_1 \supset \varphi_2))$

Table 6.1: $\mathbf{HC}^{\mu\text{-EPPL}}$: complete calculus for μ-EPPL

(S, R, L) can be transformed in the Kripke structures $m^b = (S, R, L^b)$, such that

$$L^b(s) = \{\delta^b \in \Gamma : \text{ for all atomic EPPL formulae } \delta \text{ such that } L(s) \Vdash^{\text{EPPL}} \delta\}.$$

The next lemma shows that the translation $(\_)^b$ preserves the meaning of formulae.

Lemma 6.4.1. *Let m be a EPPL-Kripke structure. Then, $[\varphi^b]_V^{m^b} = [\varphi]_V^m$.*

Proof. We carry out the proof by structural induction.

Base Case: Let φ be an atomic EPPL formula. By definition

$$[\varphi^b]_V^{m^b} = \{s \in S : \varphi^b \in L^b(s)\} = \{s \in S : L(s) \Vdash^{\text{EPPL}} \varphi\} = [\varphi]_V^m.$$

Induction Step:

1. For negation and implication it is straightforward.
2. φ is $(\Diamond\phi)$. Then, we have $\varphi^b = (\Diamond\phi^b)$, and using the induction hypotheses $[\phi^b]_V^{m^b} = [\phi]_V^m$ we get
$$[\varphi^b]_V^{m^b} = \{s \in S : \text{ exists } s' \in S, (s, s') \in R, \text{ and } s' \in [\phi^b]_V^{m^b}\} = [\varphi]_V^m.$$
3. φ is $\mu\xi.\phi$. Then, $\varphi^b = \mu\xi.\phi^b$, and by hypotheses $[\phi^b]_V^{M^b} = [\phi]_V^m$. Hence,
$$[\mu\xi.\phi^b]_V^{m^b} = \bigcap\{S' \subseteq S : [\phi^b]_{V[\xi \leftarrow S']}^{m^b} \subseteq S'\} = [\varphi]_V^m.$$

□

Using this lemma we are able to prove the *conservativeness* of μ-EPPL semantics relative to the μ-TL.

Corollary 6.4.1. $m, s \Vdash^{\mu\text{-EPPL}} \varphi$ iff $m^b, s \Vdash^{\mu\text{-TL}} \varphi^b$.

Corollary 6.4.2. If $\models^{\mu\text{-TL}} \varphi$ then $\models^{\mu\text{-EPPL}} \varphi$.

Regarding the EPPL conservativeness we establish the following result.

Lemma 6.4.2. *For any EPPL formula δ, we have*

$$\models^{EPPL} \delta \quad \textit{iff} \quad \models^{\mu\text{-}EPPL} \delta.$$

Proof. ($\rightarrow$) Suppose $\models^{\text{EPPL}} \delta$. Given a μ-EPPL model $m = (S, R, L)$ and for any state $s \in S$, $m, s \Vdash^{\mu\text{-EPPL}} \delta$ iff $L(s) \Vdash^{\text{EPPL}} \delta$. Since δ is a valid EPPL formula, it is also a valid μ-EPPL formula.

($\leftarrow$) Given a EPPL model m, and assignment ρ, we can consider the μ-EPPL model $m = (\{s\}, \{(s, s)\}, L(s) = (m, \rho))$. By the hypotheses $m, s \Vdash^{\mu\text{-EPPL}} \delta$, and so $L(s) \Vdash^{\text{EPPL}} \delta$. Hence δ is a valid EPPL formula. □

We are now able to prove the soundness of the axiomatic system $HC^{\mu\text{-EPPL}}$.

Proposition 6.4.1. *The axiomatization $HC^{\mu\text{-}EPPL}$ is sound.*

Proof. As usual, we proceed by proving that all axioms and rules are valid. From Lemma 6.4.2 and completeness of EPPL given by Theorem 2.2.1 we get that the Axiom $[Ax1]$ is valid. The validity of Axiom $[Ax2]$ is obtained by using Lemma 6.4.2 and the completeness of the $\mu-$calculus. Validity of rules $[R1]$ and $[R2]$ can be proved using Lemma 6.4.1. □

The axiomatic system $HC^{\mu\text{-EPPL}}$ extends the theoremhood of both HC^{EPPL} and $HC^{\mu\text{-TL}}$ systems.

Lemma 6.4.3. *Let φ be a μ-EPPL formula and δ a EPPL formula.*

(i) If $\vdash^{\mu\text{-}TL} \varphi^b$ then $\vdash^{\mu\text{-}EPPL} \varphi$.

(ii) Moreover, $\vdash^{EPPL} \delta$ iff $\vdash^{\mu\text{-}EPPL} \delta$.

Proof. (i) The proof proceeds by induction on the length of the derivation of φ, and by using the fact that all axioms and rules of $HC^{\mu\text{-TL}}$ are also in $HC^{\mu\text{-EPPL}}$.

(ii) ($\rightarrow$) As in the proof of (i), since HC^{EPPL} and $HC^{\mu\text{-EPPL}}$ have the same EPPL tautologies, and EPPL inference rules.

($\leftarrow$) Let $\vdash^{\mu\text{-}\mathsf{EPPL}} \delta$. By soundness we have $\Vdash^{\mu\text{-}\mathsf{EPPL}} \delta$. Given a EPPL model $\mathcal{P}$ and assignment ρ, consider the EPPL-Kripke structure $m = (\{s\}, \{(s,s)\})$ with $L(s) = (\mathcal{P}, \rho)$. Then $m \Vdash^{\mu\text{-}\mathsf{EPPL}} \delta$ iff $\mathcal{P}, \rho \Vdash^{\mathsf{EPPL}} \delta$. Hence, by EPPL completeness we get $\vdash^{\mathsf{EPPL}} \delta$. □

Given a μ-EPPL formula φ, let $at(\varphi) = \{\delta_1, \ldots, \delta_n\}$ be the set of EPPL atomic formulae. For each k-vector $u \in \{0,1\}^k$, consider the EPPL formula

$$\delta_u = \delta'_1 \cap \ldots \cap \delta'_k, \text{ where} \qquad \delta'_i = \begin{cases} \delta_i & \text{if } u(i) = 1 \\ (\sim \delta_i) & \text{otherwise.} \end{cases}$$

Let $C \subseteq \{0,1\}^k$ be the set of k-vectors u such that δ_u has a model, *i.e.* is consistent, and consider the EPPL formula $\delta_\varphi = \cup_{u \in C} \delta_u$. We use the abbreviation $AG\varphi := (\nu\xi.(\varphi \cap (\Box\xi)))$.

Lemma 6.4.4. *If $\vdash^{\mu\text{-}EPPL} \varphi$ then $\vdash^{\mu\text{-}EPPL} AG\varphi$.*

Proof. We have the derivation:

1)	$\vdash^{\mu\text{-}\mathsf{EPPL}} \varphi$	hypothesis
2)	$\vdash^{\mu\text{-}\mathsf{EPPL}} (\varphi \supset \bot) \supset \bot)$	tautology
3)	$\vdash^{\mu\text{-}\mathsf{EPPL}} (\varphi \supset \Diamond\bot) \supset \bot)$	Axiom $A4$
4)	$\vdash^{\mu\text{-}\mathsf{EPPL}} \mu\xi.(\varphi \supset \Diamond\xi) \supset \bot)$	applying rule $R2$ to 3)
5)	$\vdash^{\mu\text{-}\mathsf{EPPL}} \top \supset \sim (\mu\xi.(\varphi \supset \Diamond\xi))$	propositional tautology
6)	$\vdash^{\mu\text{-}\mathsf{EPPL}} \sim (\mu\xi.(\sim \varphi \cup (\sim \Box \sim \xi)))$	by modus ponens, and tautologies
7)	$\vdash^{\mu\text{-}\mathsf{EPPL}} \sim (\mu\xi. \sim (\varphi \cap (\Box \sim \xi))) = AG\varphi$	tautology for $\cup$

□

Lemma 6.4.5. *Let φ be a μ-EPPL formula and $m = (S, R, L)$ a Kripke structure.*

(i) $[AG\varphi^b]^m_V = [\varphi^b]^m_V \cap [\Box AG\varphi^b]^m_V$, for all valuations V;

(ii) $s \in [AG\varphi^b]^m_V$ and $(s, s') \in R$ then $s' \in [AG\varphi^b]^m_V$.

Proof. We get (i) from the equivalence

$$(\nu\xi.(\varphi \cap (\Box\xi))) = (\varphi \cap (\Box(\nu\xi.(\varphi \cap (\Box\xi))))).$$

Claim (ii) is obtained by using (i), i.e. the fact that $[AG\varphi^b]^K_V \subseteq [\Box AG\varphi^b]^K_V$. □

Lemma 6.4.6. *Let $K = (S, R, L)$ be a Kripke structure and $s \in S$. If $m' = (S', R', L')$ is a Kripke structure such that $S' \subseteq S$ contains all successors of s (where R' and L' are the restriction to S' of their counterparts in m), then for every formula φ we have*

$$s \in [\varphi^b]^{m'}_V \text{ iff } s \in [\varphi^b]^m_V.$$

Proof. The proof is by structural induction on φ. Let V be a valuation.

Base Case: If φ is an EPPL formula then $L(s) = L'(s)$, so $s \in [\varphi^b]^{m'}_V$ iff $s \in [\varphi^b]^m_V$.

Induction Step: The case of negation and implication are straightforward.

– if φ is $(\Diamond\phi)$ then $s \in [\Diamond\phi]^{m'}_V$ iff there is $s' \in S'$ with $(s, s') \in R$ and $s' \in [\phi]^{m'}_V$ iff $s \in [\Diamond\phi]^m_V$, and using the fact that S' contains all the successors of s.

– if φ is $(\mu\xi.\phi)$ is straightforward, using the induction hypotheses that $s \in [\varphi^b]^{m'}_V$ iff $s \in [\varphi^b]^m_V$. □

Proposition 6.4.2. *Let φ be a valid μ-EPPL formula. Then there is a EPPL formula δ_φ such that $\vdash^{\mu\text{-EPPL}} \delta_\varphi$, and $\vdash^{\mu\text{-TL}} (AG\delta^b_\varphi \Rightarrow \varphi^b)$.*

Proof. Let φ be a valid μ-EPPL formula, and δ_φ be as defined before. Suppose that

$$\nvdash^{\mu\text{-TL}} (AG\delta^b_\varphi \Rightarrow \varphi^b).$$

By completeness of the $HC^{\mu\text{-TL}}$ calculus we get that $\nVdash^{\mu\text{-TL}} (AG\delta^b_\varphi \Rightarrow \varphi^b)$. So, there is a Kripke structure $m = (S, R, L)$ and $s \in S$, such that

$$m, s \nVdash (AG\delta^b_\varphi \Rightarrow \varphi^b), \text{ and hence } \quad m, s \Vdash AG\delta^b_\varphi \text{ and } m, s \nVdash \varphi^b.$$

For all $s' \in [AG\delta^b_\varphi]^K$, we have that $s' \in [\delta^b_\varphi]^K$, and hence there is an $u_{s'} \in C$ such that $s' \in [\delta_{u_{s'}}]^K$. Therefore, there is a EPPL model $m_{u_{s'}}$ and assignment $\rho_{u_{s'}}$ such that $m_{u_{s'}}, \rho_{u_{s'}} \Vdash \delta_{u_{s'}}$. Consider the EPPL-Kripke structure $M_m = (S', R', L')$ such that $S' = [AG\delta^b_\varphi]^m$, $R' \subseteq R$ is the restriction of R to S', and $L'(s') = (m_{u_{s'}}, \rho_{u_{s'}})$ for all $s' \in S'$.

From Lemma 6.4.6, $(m')^b, s \not\Vdash^{\mu\text{-}\mathsf{TL}} \varphi^b$, and using Corollary 6.4.1 we get $M_m, s \not\Vdash^{\mu\text{-}\mathsf{EPPL}} \varphi$, which is a contradiction with the hypotheses $\Vdash^{\mu\text{-}\mathsf{EPPL}} \varphi$. Hence, $\vdash^{\mu\text{-}\mathsf{TL}} (AG\delta_\varphi^b \Rightarrow \varphi^b)$. □

Theorem 6.4.1. *The axiomatization $HC^{\mu\text{-}EPPL}$ is weakly complete.*

Proof. Let φ be a valid EPPL formula. From Proposition 6.4.2 we have that $\vdash^{\mu\text{-}\mathsf{EPPL}} (AG\delta_\varphi \supset \varphi)$. Using Lemma 6.4.4 we get $\vdash^{\mu\text{-}\mathsf{EPPL}} AG\delta_\varphi$. By modus ponens we get $\vdash^{\mu\text{-}\mathsf{EPPL}} \varphi$. □

6.4.5 Satisfaction

Given a μ-EPPL formula φ consider the EPPL formula δ_φ as constructed above. We have that δ_φ is a valid EPPL formula, $\models^{\mathsf{EPPL}} \delta_\varphi$. Let $m = (S, R, L)$ be a Kripke structure such that $m \Vdash^{\mu\text{-}\mathsf{TL}} (\delta_\varphi^b \wedge \varphi^b)$. Therefore, for each $s \in S$ there is an $u_s \in C_\varphi$ such that $s \in [\delta_{u_s}]^K$. Since each formula δ_{u_s} is satisfiable, there is an EPPL model m_{u_s} and assignment ρ_{u_s} such that $m_{u_s}, \rho_{u_s} \Vdash \delta_{u_s}$. Consider the EPPL-Kripke structure $M_m = (S, R, L_m)$ with $L_m(s) = (m_{u_s}, \rho_{u_s})$, for all $s \in S$. Clearly, $(M_m)^b$ and m satisfy the same formulae over the propositional symbols $\{\delta^b : \delta \in at(\varphi)\}$.

Theorem 6.4.2. *Let φ be a μ-EPPL formula. The formula φ is satisfiable iff $(\delta_\varphi^b \wedge \varphi^b)$ is satisfiable.*

Proof. ($\rightarrow$) Let φ be a satisfiable μ-EPPL formula. Therefore, there is a EPPL-Kripke structure $m = (S, R, L)$ such that $m \Vdash \varphi$. From Corollary 6.4.1, $m^b \Vdash^{\mu\text{-}\mathsf{TL}} \varphi^b$. By construction, and as stated before $\Vdash^{\mu\text{-}\mathsf{EPPL}} \delta_\varphi$. Therefore, $m^b \Vdash^{\mu\text{-}\mathsf{EPPL}} (\delta_\varphi^b \wedge \varphi^b)$.

($\leftarrow$) Suppose that $(\delta_\varphi^b \wedge \varphi)$ is a satisfiable formula. Therefore, there is a Kripke structure $m = (S, R, L)$, such that $m \Vdash^{\mu\text{-}\mathsf{TL}} (\delta_\varphi^b \wedge \varphi)$. Hence, $(M_m)^b \Vdash^{\mu\text{-}\mathsf{TL}} \varphi^b$ iff $M_m \Vdash^{\mu\text{-}\mathsf{EPPL}} \varphi$. □

The satisfaction problem for μ-calculus is EXPTIME-complete [67]. Furthermore, since the formula $\delta_\varphi^b \wedge \varphi^b$ is exponentially larger than φ, one could imagine that the complexity of deciding satisfiability of μ-EPPL formulae is exponentially worse than that for μ-TL. However, as we shall see in the next section, for some cases we need not apply the SAT algorithm directly to the entire formula δ_φ^b.

Exogenous Probabilistic Linear Temporal Logic

An important sub-logic of μ-EPPL is the one obtained by taking just the path connectives next X and until U, such that $\mathsf{X}\varphi := (\Diamond\varphi)$, and $\varphi_1\mathsf{U}\varphi_2 := (\mu.\xi(\varphi_2 \cup (\Diamond(\varphi_1 \cap \xi))))$. We call this logic exogenous probabilistic linear temporal logic (EPLTL).

Due to the correspondence of LTL formulae with Büchi automata given by Proposition 1.1.1, the problem of finding a model is equivalent to the emptiness problem of the associated Büchi automaton. The size of the Büchi automaton is exponential relative to the formula size. Although the emptiness problem is NLOG-complete by Proposition 1.1.2, we have the PSPACE bound for the satisfaction relative to the formula size.

The algorithms introduced in [96, 49] to solve satisfiability can be extended to EPLTL.

Theorem 6.4.3. *The satisfiability problem for EPLTL is in PSPACE.*

Proof. (Proof sketch) In both cases [96] and [49], the algorithms presented solve the problem by constructing a witness path, where states are defined by a set of subformulae. In the first case by guessing the next state, and in the second by expanding a graph with tableaux techniques. In each step, consistence tests are performed, and the syntactic states are discarded if they fail such tests. To extend this algorithms to EPLTL formulae we just need to add the EPPL consistence test, via EPPL SAT. Since EPPL SAT is in PSPACE (Theorem 2.2.2), we get that the satisfiability problem for EPLTL is also in PSPACE. □

This proof is further expanded in Chapter 8.

6.5 Applications in planning

We will now show some of the properties μ-EPPL can specify, illustrating also some of its usefulness (although we leave a more complex example for the full paper where we also incorporate uncertainty of positions). We will not use full μ-EPPL, but we will use another important sublogic, a restriction of the μ-TL to LTL, as it will be enough for our demonstration.

In this 4x4 grid, the robot will have to navigate through the cells until it finds the target, spending fuel to do so, and regaining it at some locations yet to be discovered. The fuel will be carried and delivered in discrete amounts, for instance $\{0, \frac{1}{10}, \ldots, \frac{9}{10}, 1\}$ using k for any of these discrete amounts, and thus we are modeling the fuel level as a probability. We will stipulate that

the robot will start its mission with a full tank, and that the developers are willing to spend K extra fuel units to supply the robot.

We will require the following propositional symbols and constants: (i) $\{e_{(i,j)} : i, j \in \{1, \ldots, 4\}\}$, representing the location of the robot; (ii) *fuel*, representing the amount of fuel available; (iii) $\{sup_{(i,j)} : i, j \in \{1, \ldots, 4\}\}$ representing the distribution of the refuel supplies, (iv) $c_{(i,j),(i',j')}$ representing the fuel cost of moving from cell (i, j) to cell (i', j'); (v) K the total amount of fuel the developers are able to deliver for refuel; and (vi) T the threshold cost.

Our specification is as follows:

1. The robot can only be in one place at once:
   $\mathsf{G}((\bigcap_{i,j} \int e_{(i,j)} = 0 \cup \int e_{(i,j)} = 1) \cap (\sum_{i,j} \int e_{(i,j)} = 1))$

2. The robot can only move to adjacent cells and if it has some fuel left, spending $c_{(i',j')}$ fuel to move from cell $e_{(i,j)}$ to cell $e_{(i',j')}$, where $C_{(i,j)} = \{(i', j') \in \{1, ..., 4\} \times \{1, ..., 4\} : |i - i'| \leq 1, |j - j'| \leq 1, (i, j) \neq (i', j')\}$:

$$\begin{aligned}
&\mathsf{G}((\int \mathit{fuel} = \mathsf{k} \cap \mathsf{k} > 0 \cap \int e_{(i,j)} = 1) \supset \\
&\mathsf{X}((\bigcup_{(i',j') \in C_{(i,j)}} \int e_{(i',j')} = 1) \cap \\
&((\int \mathit{fuel} = 0 \bigcap \mathsf{k} + K * \int sup_{(i,j)} - c_{(i',j')} < 0) \oplus \\
&(\int \mathit{fuel} = 1 \bigcap \mathsf{k} + K * \int sup_{(i,j)} - c_{(i',j')} > 1) \oplus \\
&(\int \mathit{fuel} = \mathsf{k} + K * \int sup_{(i,j)} - c_{(i,j),(i',j')}))));
\end{aligned}$$

3. The probability distribution of supplies is set at the beginning:
   $(\bigcup_{\mathsf{k}} (\int sup_{(i,j)} = \mathsf{k}) \supset \mathsf{G}(\int sup_{(i,j)} = \mathsf{k})) \cap \sum_{i,j} \int sup_{(i,j)} = 1;$

4. The cost of deployment of supplies is limited by a threshold T; moreover, the cost grows quadratically with the distance:
   $\sum_{i,j} (\int sup_{(i,j)}) \frac{(i+j)^2}{4+4} \leq T;$

5. If the robot has no fuel, it will not leave the cell:
   $\bigcup_{i,j} \mathsf{F}((\int \mathit{fuel} = 0 \cap \int e_{(i,j)} = 1) \supset \mathsf{G} \int e_{(i,j)} = 1);$

6. The robot begins at cell $(1,1)$, and its gas tank is full:
   $\int e_{(1,1)} = 1 \cap \int \mathit{fuel} = 1$;
7. The robot will not revisit any cell:
   $(\bigcup_{i,j} \mathsf{G}(\int e_{(i,j)} = 1 \supset \mathsf{XG}(\int \mathit{fuel} > 0 \supset \int e_{(i,j)} = 0))$;
8. All paths should eventually lead to the target location:
   $\mathsf{F}(\int e_{(4,4)} = 1)$.

So, using the SAT algorithm, one is able to decide whether the set of constraints has a model. Note that if we omit the last axiom, the model generated is the model of all possible paths *Healer 2.0* can take. However, if the last axiom is included, the generated model is a trimmed down version of the earlier model, such that all paths are viable and the robot can choose from any of them. Either way, the developers obtain their unknown distribution of the fuel supplies.

Remark 6.5.1. Since μ-EPPL has the finite model theorem, there will exist, for every infinite model of a formula, a finite model that will also satisfy the same formula. This automatically restricts the application of μ-EPPL since for instance, Markov chains are unspecifiable. Moreover, even if one does know that the model has a finite number of probability distributions, the logic has no efficient way to specify relations of probability distributions.

For instance, in our example, we do know that the amount of fuel is finite, and that moving requires fuel, therefore we know that there will only exist a finite set of distributions. However, the formula $\mathsf{AG}((\int e_{i,j} = 1 \cap \int fuel = x) \supset \mathsf{AX}(\int e_{i',j'} = 1 \cap \int fuel = f(x))$ will not work, since the assignment of the variable x is local.

6.6 Conclusions

The proposed logic can be used to specify any type of probabilistic interaction provided that there are a finite number of probability distributions that model the problem. So the example provided in Section 6.5 can be extended to consider robots with random movements, since we only consider finite amount of fuel. This finitary constraint is necessary for the SAT algorithm to remain decidable, as any attempt to go beyond leads easily to undecidability. Note that one advantage of this approach compared to planning using SAT over propositional logic is that we do not require a bound on time, although both methods require to put a bound on the number of states (in

our case in the number of probability distributions). Finally, completeness of the Hilbert calculus may help to derive additional properties about the behavior of all solutions.

6.7 Concluding Remarks

While μ-EPPL and its restrictions are unable to adequately model Markov chains or Markov decision processes due to the infinite state space needed, its satisfiability algorithm still provides an important extension of the most common specification languages used in Planning. While this paper does not approach the model-checking problem, it is straightforward to see that in the case of finite EPPL-Kripke structures, the model-checking algorithm for μ-TL can easily be adapted.

However, even if one is unable to specify Markov chains, or Markov decision processes, one could still hope (in principle) to be able to verify μ-EPPL properties over the infinite EPPL-Kripke structures they generate. In the case of Markov decision processes, that is not so, as it was already pointed out in Section 3.3. In the case of Markov chains, the problem is not as well studied.

Consider the following probabilistic program that implements the toss of a coin, where the subprogram $dieRoll(3)$ rolls a balanced die with 3 possible results $\{1, 2, 3\}$:

```
coinToss():=
0 c=1;
1 while true, do:
2 c:=dieRoll(3);
3 if c<= 2 break;
4 return c;
```

The program can be translated to a Markov Chain shown in Figure 6.2, where the labeling function labels each state with the value of c followed by the line of the program.

Note, first and foremost, that the if the program terminates (represented in the Markov chain by reaching line 4 of the program), then it outputs 1 or 2 with probability $\frac{1}{2}$ each. Moreover, it is also easy to see that the probability of termination is 1; both these facts can be specified and verified in a probabilistic temporal logic, like pCTL or PQLTL. Using PQLTL syntax, these statements can be written as $\int \mathsf{F}(s_{1,4} \vee s_{2,4}) = 1$ and $\int \mathsf{F}s_{1,4} = \frac{1}{2} \cap \int \mathsf{F}s_{2,4} = \frac{1}{2}$.

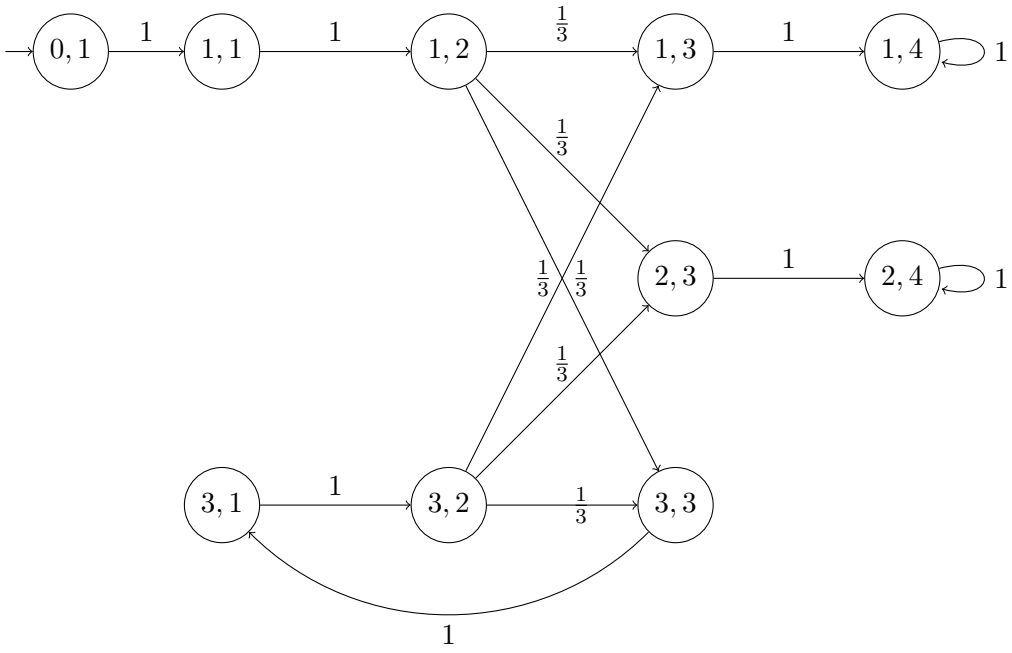


Figure 6.2: The reachable part of the Markov chain that models a finite memory probabilistic program

The logics μ-EPPL and in particular EPLTL are able to specify related properties. For instance consider the property $\mathsf{F} \int (s_{1,4} \vee s_{2,4}) = 1$. This property is not satisfiable in the EPPL-Kripke structure induced by the Markov chain, due to the fact that the limit distribution that assigns probability $\frac{1}{2}$ to each of the halting states is only reached in the limit; as such this property allows us to confirm that the program may not in fact ever terminate; note if we relax the equality by considering equality up to ϵ, for any small value of ϵ, then the formula is satisfied.

The equidistribution of the outputs can be specified by $\mathsf{F}(\int s_{1,4} \vee s_{2,4} > 0 \supset (\int s_{1,4} = \frac{\int s_{1,4} \vee s_{2,4}}{2} \cap \int s_{2,4} = \frac{\int s_{1,4} \vee s_{2,4}}{2})$; note however that this formula is satisfied in the EPPL-Kripke structure induced by the Markov chain, but not necessarily because of its limit distribution. In the fourth instant of time $(\int s_{1,4} \vee s_{2,4} > 0 \supset (\int s_{1,4} = \frac{\int s_{1,4} \vee s_{2,4}}{2} \cap \int s_{2,4} = \frac{\int s_{1,4} \vee s_{2,4}}{2})$ does hold, because in that instant $\int s_{1,4} = \int s_{2,4} = \frac{1}{3}$.

So what does in fact a property like $\mathsf{FG}(\int s_{1,4} =_\epsilon \frac{1}{2} \cap \int s_{2,4} =_\epsilon \frac{1}{2})$ guarantees us? It assures us that if we are running $coinToss()$ on a computer, after some instant, the act of sampling an halting state simulates a balanced

coin up to some unspecified precision ϵ; a priori, we do not even know if the computer is in an halting state; we did not observe the state of the system at any time whatsoever; the argument in this case relies only on the limit behavior of the program. Moreover one could design programs such that $\int \mathsf{F}\ halt_1 = \int \mathsf{F}\ halt_2$ but the probability of being in $halt_1$ is never equal the probability of being in $halt_2$ in any instant of time.

Finally, we would like to point out that these verification problems concerning the dynamics of a probability distribution driven by a Markov chain are not as well studied as those associated with the calculation of probabilities of evolutions. The next chapter sheds some light on this subject.

Chapter 7

Decidability of Approximate Skolem Problem and Applications to Logical Verification of Dynamical Properties of Markov Chains

This chapter concerns an article published in ACM Transactions on Computational Logic, written by myself, David Henriques and Paulo Mateus. It concerns a topic that emerged from the development of μ-EPPL and its sublogic EPLTL. While neither μ-EPPL nor EPLTL were suitable to the specification of probabilistic systems encompassing infinite probability distributions, we recognized that not only testing whether a EPLTL formula is satisfied by a Markov chain was an interesting practical problem, which is quite distinct from the pCTL* model-checking paradigm, but also it was not obvious whether the problem would be decidable.

Herein, we show that the problem of verifying a property specified using an adapted version of EPLTL over a Markov chain is equireducible to a long-standing (and simple to state) open problem in Number Theory. However, the proof of the equireducibility turns out to be quite useful, as it can be turned into an algorithm to decide the approximate version of the problem.

Some of the preliminary contents in this Chapter were introduced in the previous chapters, however, we will require some adaptations, namely in the syntax and semantics of EPLTL. The major change in the syntax of EPLTL is that one no longer allows arbitrary algebraic operations in proba-

bilistic terms, being only able to express linear combinations of probabilities of propositional formulae. Similar modifications can already be seen in Definition 2.1.9, or in Subsection 4.4.1. The change in semantics reflects the fact that instead of (infinite) arbitrary EPPL-Kripke structures, we are now focused upon those induced by Markov chains.

7.1 Introduction

Verification of properties in probabilistic deterministic systems is a critical area of research in the field of Computer Science. Currently, there are many tools that verify properties of systems modeled as Markov chains [68, 26, 57]. However most of the work is focused on verifying path-like specifications, that is, what proportion of possible executions of the system satisfy a given property [102, 5, 7]. These specifications are undoubtedly interesting since they are ubiquitous; however there are many interesting and intuitive properties that they can not express, in particular, considerations about the dynamical evolution of state probabilities are either convoluted or impossible to state in these frameworks, as pointed in e.g. [69, 63, 11].

Recently **Agrawal etal.** [1] developed work that builds upon well-known results in Probability Theory in order to reason with the dynamical system induced by a Markov chain. This work focuses on characterizing the behavior of a Markov chain by the use of symbolic distributions that evolve dynamically and by considering error bounds for these evolutions. We work in the same setting, but follow a different approach. We display the close connection between the exact verification of linear properties over the dynamical evolution of state probabilities and a famous open problem in Number Theory, the Skolem problem [84]. In fact, as remarked by Agrawal [1],"[the verification problem for linear time properties over the dynamical of state probabilities] seems to be strongly related to the long-standing open problem on linear recurrent sequences known as the Skolem problem".

The Skolem problem, originally formulated by Thoralf Skolem [97], was partially solved by himself using non-constructive techniques. More recently, thanks to the work in [74, 14], it has been reformulated as a decision problem which is only known to be decidable for low dimensions [84, 85].

Skolem problem can be stated [84] as follows:

Proposition 7.1.1. *Given $x, y \in \mathbb{Q}^m$, and $L \in \mathbb{Q}^{m^2}$, can we decide if the following statement is true:*

$$\exists n. x^T L^n y = 0.$$

The problem itself can be seen whether the repeated mapping of a starting vector, x, through some linear system, L, and later projected upon another vector, y, ever reaches 0. In our probabilistic setting, we have a similar problem, a reachability query, where we ask whether, from an initial distribution (the analogue of x), a Markovian evolution (the analogue of L^i) ever reaches a point where a combination of the probabilities of states (the analogue of y) is exactly 0. It is easy to show that these two problems are equireducible. A more interesting question is whether the Skolem problem is fundamentally easier than the verification of more complex linear time properties (other than simple reachability queries). We show that in fact, for linear time properties, the verification problem is as difficult as the Skolem problem.

We also pursue the subject in a different perspective. If the verification problem is as hard as the (open) Skolem problem, can we at least solve approximate versions of them? Let us relax Problem 7.1.1 in the following way:

Proposition 7.1.2. *Given $x, y \in \mathbb{Q}^m$, and $L \in \mathbb{Q}^{m^2}$, does there exist $d \in \mathbb{Q}$, such that for all $\epsilon \in]0; d[\cap \mathbb{Q}$ we can decide if the following statement is true:*

$$\exists n. -\epsilon < x^T L^n y < \epsilon.$$

Can we present a procedure to solve Problem 2? And can we solve the equivalent relaxation of the verification problem?

Agrawal [1] considered a similar problem from the verification of dynamical properties perspective. Our approach focuses more on the relation between the approximate Skolem problem and the approximate verification problem, solving both.

Our contributions

- We show the equireducibility between the Skolem problem and the verification of linear time properties for dynamical systems induced by Markov chains.

- We present decision procedures for approximate versions of both problems.

7.2 Preliminaries

In this section we introduce the main definitions and prior results used throughout the article.

7.2.1 Markov chains

Markov chains are a widely used formalism to model memoryless discrete probabilistic dynamic systems. Let Λ be a finite set of propositional symbols. We recall its definition from Section 3.1.

Definition 7.2.1. A *(state labelled) Markov chain* $\mathcal{M}$ is a tuple $\mathcal{M} = (S, M, \mu, L)$ over the propositional symbols Λ where:

- $S = \{s_1, \ldots, s_m\}$ is a finite set of states;
- M is a stochastic matrix of dimension m with entries in $\mathbb{Q}$, named the *transition matrix*. Intuitively, from s_i we can move towards s_j in one step with probability $M_{i,j}$;
- μ is a probability distribution over S, named the *initial distribution*;
- $L : S \mapsto \{0,1\}^\Lambda$ is the *labeling function*. Intuitively, it represents which predicates are true in each state.

A Markov chain induces a discrete dynamical system of finite probability distributions over states given by $\mu_0 = \mu, \mu_{i+1}^T = \mu_i^T M$. We are interested in verifying properties of this dynamical system. In order to do so, we need to express properties of probability distributions, and their evolution.

7.2.2 EPLTL as a logic for verification of probability distributions given by Markov chains

We consider the following logic, named EPLTL, already described and studied in [9], which has the following syntax:

Definition 7.2.2. The *well-formed formulae* in EPLTL over the propositional symbols Λ are given below in Backus-Naur notation:

$$\beta := p \quad | \quad (\beta \wedge \beta) \quad | \quad (\neg\beta) \qquad \text{(basic formulae)}$$

$$t := r \quad | \quad (t + t) \quad | \quad r \int \beta \qquad \text{(rational terms)}$$

$$\delta := \underbrace{\sim \delta \quad | \quad \delta \cap \delta \quad | \quad \mathsf{X}\delta \quad | \quad \delta \mathsf{U} \delta}_{\text{temporal formulae}} \quad | \quad \underbrace{t = t}_{\text{comparison formulae}} \qquad \text{(formulae)}$$

where $p \in \Lambda$ and $r \in \mathbb{Q}$.

The logic will enable us to reason about distributions over the propositional symbols Λ using terms of the form $\int \beta$ and comparison formulae; the linear temporal reasoning is performed using the outer propositional negation $\sim$, conjunction $\cap$, the *neXt* time connective, and the *Until* connective. The common abbreviations for *sometime in the Future* and *Globally* will also be considered.

The semantics considered in this article are designed specifically towards Markov chains.

Definition 7.2.3. The denotation of a rational term t in a Markov chain $\mathcal{M} = (S, M, \mu, L)$ in the instant $i \in \mathbb{N}$, denoted by $[\![t]\!]_{\mathcal{M},i}$, is defined inductively as:

- $[\![r]\!]_{\mathcal{M},i} = r$,
- $[\![r \int \beta]\!]_{\mathcal{M},i} = r\mu_i(\{s \in S : L(s) \Vdash^{\mathsf{PL}} \beta\})$,
- $[\![(t_1 + t_2)]\!]_{\mathcal{M},i} = [\![t_1]\!]_{\mathcal{M},i} + [\![t_2]\!]_{\mathcal{M},i}$,

where $\Vdash^{\mathsf{PL}}$ is the satisfaction relation in propositional logic. Finally the satisfaction relation between a Markov chain $\mathcal{M}$, an instant i and a formula δ can be defined.

Definition 7.2.4. The satisfaction relation between a Markov chain $\mathcal{M}$, an instant i, and a EPLTL formula δ is defined inductively:

- $\mathcal{M}, i \Vdash (t_1 = t_2)$ iff $[\![t_1]\!]_{\mathcal{M},i} = [\![t_2]\!]_{\mathcal{M},i}$,
- $\mathcal{M}, i \Vdash (\delta_1 \cap \delta_2)$ iff $\mathcal{M}, i \Vdash \delta_1$ and $\mathcal{M}, i \Vdash \delta_2$,
- $\mathcal{M}, i \Vdash (\sim \delta)$ iff $\mathcal{M}, i \nVdash \delta$,
- $\mathcal{M}, i \Vdash (\mathsf{X}\delta)$ iff $\mathcal{M}, i + 1 \Vdash \delta$,
- $\mathcal{M}, i \Vdash (\delta_1 \mathsf{U} \delta_2)$ iff there exists $j \geq i$ such that $\mathcal{M}, j \Vdash \delta_2$ and $\mathcal{M}, k \Vdash \delta_1$, for all $i \leq k < j$.

We say that a Markov chain $\mathcal{M}$ is a *model* of a EPLTL formula δ if $\mathcal{M}, 0 \Vdash \delta$.

Remark 7.2.1. Even though the classical probabilistic temporal logics like PCTL [27] are quite well suited to deal with properties about *probabilities of evolutions*, they are unable to deal with properties about the *evolution of*

probabilities; for instance, consider the Markov chain in Figure 7.2.1. The assertion A "the probability of reaching s_2 sometime in the future is $\frac{1}{2}$" is checked by considering the *probability of the set of paths* passing by the state s_2; we are measuring sets of evolutions. However, the statement B "there is an instant of time such that the probability of being in s_2 is $\frac{1}{2}$" requires, with absolute certainty, that there exists an instant in time such that s_2 holds with probability $\frac{1}{2}$.

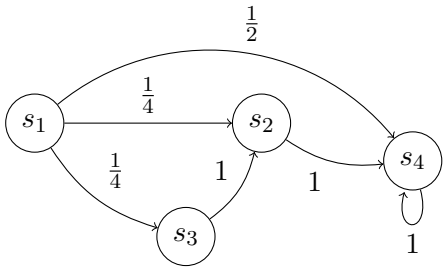


Figure 7.1: An example of the problem described. For $\mu_0 = (1, 0, 0, 0)$, there is no evolution such that s_2 holds with probability $\frac{1}{2}$, but the set of all evolutions such that s_2 holds has probability $\frac{1}{2}$.

Remark 7.2.2. Finally, while we allow any general labeling function on our Markov chains, we will just consider, without loss of generality, that the labeling function $L : S \mapsto \{0, 1\}^S$ maps a state s to the valuation that is only true on s. We can easily rewrite EPLTL formulae under one of the labelings into equivalent formulae under the other. In order to shorten notation, henceforth we will represent a Markov chain $\mathcal{M} = (S, M, \mu, L)$ simply by (M, μ).

7.2.3 The Skolem Problem

We now describe a problem from Number Theory that, surprisingly enough, is the cornerstone for verification of EPLTL formulae in Markov chains.

The Skolem problem is usually stated as a decision problem over linear recurrence sequences. A linear recurrence sequence x_n of order k over the rationals is defined as:

$$
\begin{aligned}
&x_0 = c_0, \ldots, x_{k-1} = c_{k-1}, \\
&x_n = a_{k-1}x_{n-1} + a_{k-2}x_{n-2} + \ldots + a_0 x_{n-k}, \\
&\quad c_0, \ldots, c_{k-1}, a_0, \ldots, a_{k-1} \in \mathbb{Q}.
\end{aligned}
$$

Thoralf Skolem investigated [97] whether one could characterize the set $Z(\{x_n\}) = \{n \in \mathbb{N} : x_n = 0\}$, for each sequence x_n. Originally Skolem proved that the set $Z(\{x_n\})$ could be written as:

$$
\begin{aligned}
Z(\{x_n\}) = F \cup G_1 \cup G_2 \cup \ldots \cup G_j, \\
\text{where } F \text{ is a finite (possibly empty) set and} \\
G_1, \ldots G_j \text{ are arithmetical progressions.}
\end{aligned}
$$

Unfortunately the proof was non-constructive. Later on, it was proved that all coefficients of all the arithmetical progressions are effectively computable [14]. However, there is still no known means of producing F, or testing its emptiness, which leaves the original problem open. The Skolem problem can also be approached by considering matrices instead of linear recurrence sequences:

$$
\text{Given } x, y \in \mathbb{Q}^k, L \in \mathbb{Q}^{k^2} \text{ characterize } Z(x^T L^n y) = \{n \in \mathbb{N} : x^T L^n y = 0\}
$$

Both versions are equireducible and so we also know that $Z(x^T L^n y)$ can be written as the union of a finite set and a finite union of arithmetical progressions. Since the coefficients of the arithmetical progressions are computable, the characterization problem can be restated as the following decision problem:

Proposition 7.2.1 (Skolem Problem)**.** *Given $x, y \in \mathbb{Q}^m$ and $L \in \mathbb{Q}^{m^2}$, decide if the following statement is true:*

$$
\exists n . x^T L^n y = 0.
$$

This more modern restatement is the one followed by the literature [84, 85]; we will motivate this restatement with a simple example:

Example 7.2.1. Assume that the Skolem problem is decidable. Now, suppose that $Z(x^T L^n y) = F \cup G_{q\mathsf{k}+r}$, $\mathsf{k} \in \mathbb{N}$, such that we do not know F, but we can compute the period of the arithmetical progression q, and the shift r. Then, we can compute F by repeatedly querying whether some subsequences of $x^T L^n y$ have any zeros.

- Check whether there exists any exceptional zeros until the residue r; if there are, add them to the set F.
- Check whether $Z(x^T(L^q)^n L^{r+1} y)$ is empty. If so, we know that there are no zeros of the form $q\mathsf{k} + r + 1$. Otherwise, compute the index of the first zero of this form, then consider the subsequence beginning at that position and reiterate the process until the Skolem oracle returns that there are no more zeroes. Repeat the procedure for each of the $q-1$ residue classes except the one with infinitely many zeroes, $q\mathsf{k}+r$.
- This procedure will terminate, since there are only a finite number of non-periodic zeroes.
- If $Z(x^T L^n y)$ is described by more than one arithmetical progression, for instance with periods q_1, q_2, then one adapts the algorithm to be applied to the least common multiple amongst the periods.

We will denote the union of a finite set F and finitely many arithmetical progressions $p_1\mathsf{k} + r_1, \ldots, p_n\mathsf{k} + r_n$ by a pair (F, G), where $G = \{(p_1, r_1), \ldots, (p_n, r_n)\}$ (henceforth named a *representation*). We say that $m \in \mathbb{N}$ is *represented* in (F, G) if either $m \in F$ or $m = p_i\mathsf{k} + r_i$ for some $k \in \mathbb{N}$ and $(p_i, r_i) \in G$. Likewise, we say that a set S is *representable* if there exists a pair (F, G) such that $m \in S$ iff m is represented in (F, G). We state without proof the following fact: the Skolem problem is decidable iff for any rational matrix L, vectors x, y and representation (F, G), we can decide whether the set of elements represented by (F, G) is precisely the set $Z(x^T L^n y)$.

We note, furthermore, that the Skolem problem is equireducible to the following problem [84]:

Proposition 7.2.2. *Given $x, y \in \mathbb{Q}^m$, $c \in \mathbb{Q}$ and $L \in \mathbb{Q}^{m^2}$, decide if the following statement is true:*

$$\exists n. x^T L^n y = c.$$

We can now see why this problem is relevant for verification of EPLTL formulae. For example, checking whether $\mathcal{M} \Vdash \mathsf{F}(\int s_1 - \int s_2) = \frac{1}{2}$ can be seen to be an instance of the Skolem problem with μ as x, the Markov chain matrix as L, and y as the vector $(1, -1, 0, ..., 0)$ and $c = \frac{1}{2}$. One might entertain the thought that, due to L being a Markov chain, these instances of the Skolem problem might be easier than the general statement. However, that is not case, as we now show.

Skolem Problem over Markov chains

The reduction between the general Skolem problem and the case for Markov chains can be done in two simple steps; assume that we are given $x, y \in \mathbb{Q}^m$ and $L \in \mathbb{Q}^{m^2}$; then define $x', y' \in \mathbb{Q}^{2m+2}$ and $L' \in \mathbb{Q}^{(2m+2)^2}$ as follows:

- $x' \leftarrow (1, 0, \mathbf{0})$,
- $L' \leftarrow \begin{pmatrix} 0 & x \\ \mathbf{0} & L \end{pmatrix}^{\dagger}$,
- $y' \leftarrow (0, 0, y_1, -y_1, \ldots, y_m, -y_m)$

where $\mathbf{0}$ is a subvector or matrix of the appropriate size.

The transformation $(N)^{\dagger}$ applied to the $(m+1)\times(m+1)$ matrix consists in rewriting each entry $n_{i,j}$ as the submatrix $\begin{pmatrix} n^1_{i,j} & n^2_{i,j} \\ n^2_{i,j} & n^1_{i,j} \end{pmatrix}$, where $n^1_{i,j}, n^2_{i,j}$ are non-negative numbers such that $n_{i,j} = n^1_{i,j} - n^2_{i,j}$. It is easy to see that due to the change in the vector y', $x'^T L'^{i+1} y' = x^T L^i y$ holds. Now, we have a stochastic vector as the initial distribution, and a non negative matrix L'. In order to obtain a stochastic matrix, we will add an extra dimension which allows us to normalize each line. Let $K = \max_i \{\sum_j L'_{i,j}\}$:

- $x'' \leftarrow (x', 0)$,
- $L'' \leftarrow \frac{1}{K} \begin{pmatrix} L'_{1,1} & \ldots & L'_{1,n} & K - \sum_j L'_{1,j} \\ \ldots & \ldots & \ldots & \ldots \\ L'_{n,1} & \ldots & L'_{n,n} & K - \sum_j L'_{n,j} \\ 0 & \ldots & 0 & K \end{pmatrix}$,
- $y'' \leftarrow (y', 0)$.

So, in fact, by considering stochastic matrices and an initial stochastic vector we do not obtain an easier problem. We can also consider the probabilistic version with equality to a constant (rather than equality to 0 only) by adapting the proof of the reduction used in Problem 7.2.2. The major issue now is the fact that the Skolem problem is open. Given two vectors x, y and a matrix L, there is no known algorithm to decide whether there exists an index n such that $x^T L^n y = 0$ (except for cases of dimension less or equal to 4). Moreover, while the Skolem problem is clearly an extremely specific case of a verification problem for EPLTL, we would like to know whether more complex EPLTL formulae will be even harder to verify.

7.3 Using Skolem problem as an oracle

We now show that the problem of verification of EPLTL formulae in Markov chains is equireducible to the Skolem problem. Without loss of generality, we will assume that any comparison term is of the form $t = 0$. We will show that given representations, (F_i, G_i), for the atomic equalities of rational terms $t_i = 0$ occuring in a formula, we can compute a representation for the whole formula. This process is possible due to the fact that sets describable as the union of a finite set and a finite union of arithmetical progressions are closed under EPLTL connectives.

Remark 7.3.1. Note that each rational term $\sum_{i=1}^{N} r_i \int s_i + c$ is an affine combination of probabilities of individual states; therefore we can represent this term as the vector t^* of $N+1$ components $(r_1, \ldots, r_N, -c)$. In order to compute whether $t_i = 0$ holds after n iterations of a Markov chain (M, μ) we just need to compute whether $\mu^T M^n t^* = 0$. We will overload the notation by also denoting t^* by t.

Definition 7.3.1. The satisfaction set of $\delta \in \mathsf{EPLTL}$ in a Markov chain $\mathcal{M} = (M, \mu)$, represented as I_δ, is given inductively as:

- $I_{t=0} = \{i \in \mathbb{N} : \mu^T M^i t = 0\}$,
- $I_{\sim\delta_1} = \overline{I_{\delta_1}}$,
- $I_{\delta_1 \cap \delta_2} = I_{\delta_1} \cap I_{\delta_2}$,
- $I_{\delta_1 \cup \delta_2} = I_{\delta_1} \cup I_{\delta_2}$,
- $I_{\mathsf{X}\delta_1} = \{i \in \mathbb{N} : i + 1 \in I_{\delta_1}\}$,
- $I_{\mathsf{F}\delta_1} = \{i \in \mathbb{N} : \exists j \text{ s.t. } j \geq i, j \in I_{\delta_1}\}$,
- $I_{\delta_1 \mathsf{U} \delta_2} = \{i \in \mathbb{N} : \exists j \text{ s.t. } j \in I_{\delta_2} \text{ and } \forall k \text{ s.t. } i \leq k < j, k \in I_{\delta_1}\}$.

The proof of the following lemma is a simple exercise in structural induction.

Lemma 7.3.1. *Let $\mathcal{M} = (M, \mu)$ be a Markov chain and let I_δ be the satisfaction set of $\delta \in$ EPLTL in $\mathcal{M}$; then $(M, \mu), i \Vdash \delta$ iff $i \in I_\delta$.*

We will now show that given representations (F_i, G_i) for the basic terms $t_i = 0$ of a formula δ, then I_δ is the union of a finite set and arithmetical progressions, and we can also provide a representation (F_δ, G_δ) for I_δ.

Proposition 7.3.1. *Let $(F_1, G_1), (F_2, G_2)$ be representations for $I_{\delta_1}, I_{\delta_2}$, formulae in EPLTL. Then there exist representations for:*

$$I_{\sim\delta_1} \quad I_{\delta_1 \cup \delta_2} \quad I_{\delta_1 \cap \delta_2} \quad I_{\mathsf{X}\delta_1} \quad I_{\mathsf{F}\delta_1}.$$

Proof. The case of the **union** connective is trivial. If (F_1, G_1) is a representation of I_{δ_1} and (F_2, G_2) is a representation of I_{δ_2}, then a representation for $I_{\delta_1 \cup \delta_2}$ is simply $(F_1 \cup F_2, G_1 \cup G_2)$.

The case of the **intersection** is slightly more involved, relying on the fact that the intersection of two arithmetical progressions is still an arithmetical progression or the empty set. The resulting arithmetical progression can be computed using the Chinese Remainder Theorem [29, pp 873-876], even if the periods are not coprime. If $I_{\delta_1} = F_1 \cup G_{p_1\mathsf{k}+r_1} \cup \ldots \cup G_{p_m\mathsf{k}+r_m}$ and $I_{\delta_2} = F_2 \cup G_{q_1\mathsf{k}+s_1} \cup \ldots \cup G_{q_n\mathsf{k}+s_n}$, then their intersection can be written as:

$$I_{\delta_1} \cap I_{\delta_2} = (F_1 \cap F_2) \cup \bigcup_{j=1}^{n} (F_1 \cap G_{q_j\mathsf{k}+s_j}) \cup \bigcup_{i=1}^{m} (F_2 \cap G_{p_i\mathsf{k}+r_i}) \cup \bigcup_{i,j}^{n,m} (G_{q_i\mathsf{k}+s_i} \cap G_{p_j\mathsf{k}+r_j})$$

Now clearly the finite part can be computed, and since each intersection of arithmetical progressions is either the empty set or another arithmetical progression, then it is possible to obtain a representation for $I_{\delta_1 \cap \delta_2}$.

The case of the **complement** is naturally related with the case of the intersection. The complement of an arithmetical progression $p\mathsf{k} + r$ can be seen to be the union of the finite set $\{0, 1, \ldots, r-1\}$ and $p-1$ arithmetical progressions $p\mathsf{k} + r + i, i \in \{1, 2, \ldots, p-1\}$. Moreover the complement of a finite set F is again the union of the finite set $\{0, \ldots, max(F)\} - F$ with the trivial arithmetical progression $\mathsf{k} + max(F) + 1$. Therefore, if $I_{\delta_1} = F \cup G_{p_1\mathsf{k}+r_1} \cup ... \cup G_{p_n\mathsf{k}+r_n}$,

$$I_{\sim\delta_1} = \overline{I_{\delta_1}} = \overline{F} \cap \bigcap_{i}^{n} \overline{G_{p_i\mathsf{k}+r_i}},$$

and so, thanks to the results already proved about the intersection, it follows that $I_{\sim\delta_1}$ is still the union of a finite set and a finite number of arithmetical progressions. All the coefficients can be computed and so we can obtain a representation for $I_{\sim\delta_1}$.

Regarding the temporal connectives, the X **connective** is quite straightforward; assuming $(F_1, \{(p_1, r_1), \ldots, (p_n, r_n)\})$ is a representation for I_{δ_1}, a representation for $I_{\mathsf{X}\delta_1}$ is $(F_1', \{(p_1, r_1'), \ldots, (p_n, r_n')\})$, where F_1' is the set of

predecessors (in the natural numbers) of F_1 and $r'_i = r_i - 1$, if $r_i > 0$ or $r'_i = p_i - 1$ otherwise.

While the F **connective** can be obtained from the U connective (which we will present below), we explicitly present a construction for F here, for completeness sake. Suppose we have a representation of $I_{\delta_1} = (F, G)$. Then there are three cases: either $G \neq \emptyset$, and in this case a representation for $I_{\mathsf{F}\delta_1}$ is for example $(\emptyset, \{(1, 0)\})$, or $G = \emptyset$; if it is the latter case, if $F \neq \emptyset$ then a representation for $I_{\mathsf{F}\delta_1}$ would be $(\{0, 1, \ldots, max(F)\}, \emptyset)$, and $(\emptyset, \emptyset)$, otherwise. □

We will finally extend the result to consider the U connective. The proof relies on computing the representation of $I_{\delta_1 \mathsf{U} \delta_2}$ from a suitable finite domain, which captures all the information required. We motivate the proof with an extremely simple example:

Example 7.3.1. Suppose that we have representations for $I_{\delta_2} = (\{1\}, \{(6, 2)\})$ and $I_{\delta_1} = (\emptyset, \{(3, 3), (2, 1)\})$, so δ_1 is true at the indexes given by $3\mathsf{k} + 3$ and $2\mathsf{k} + 1$, while δ_2 is true at the indexes $6\mathsf{k} + 2$ and also at the exceptional index 1 as depicted on Figure 7.2. Now let $T = lcm(3, 2, 6) = 6$ and $K = max\{\emptyset \cup \{1\} \cup \{3, 1, 2\}\} = 3$ so that the pattern of δ_1 and δ_2 repeats with period $T = 6$ after the index $K = 3$ (as can be seen in the Figure).

We now wish to capture at which indexes is $\delta_1 \mathsf{U} \delta_2$ satisfied. In this case a representation (F, G) for $I_{\delta_1 \mathsf{U} \delta_2}$ would take $F = \{1, 2\}$, as both exceptional indexes have δ_2 as label. To build G we analyse the truth values of δ_1 and of δ_2 in the first sequence of $T = 6$ indexes after $K = 3$. This sequence will repeat because:

- its period is a multiple of all the periods involved (all arithmetic progressions will repeat with this period, some more than once),
- the indexes considered are large enough to make sure all arithmetical progressions are represented (since they all start at most at K),
- the indexes considered are large enough to make sure that none of the exceptional indexes are represented (since they appear only until K).

So, to identify G, we can just consider the labelling from $i = K + 1 = 4$ until $i = K + T = 3 + 6 = 9$. It is clear that, in this range, $\delta_1 \mathsf{U} \delta_2$ holds at indexes $5, 6, 7, 8$, so we can consider $G = \{(6, 5), (6, 6), (6, 7), (6, 8)\}$.

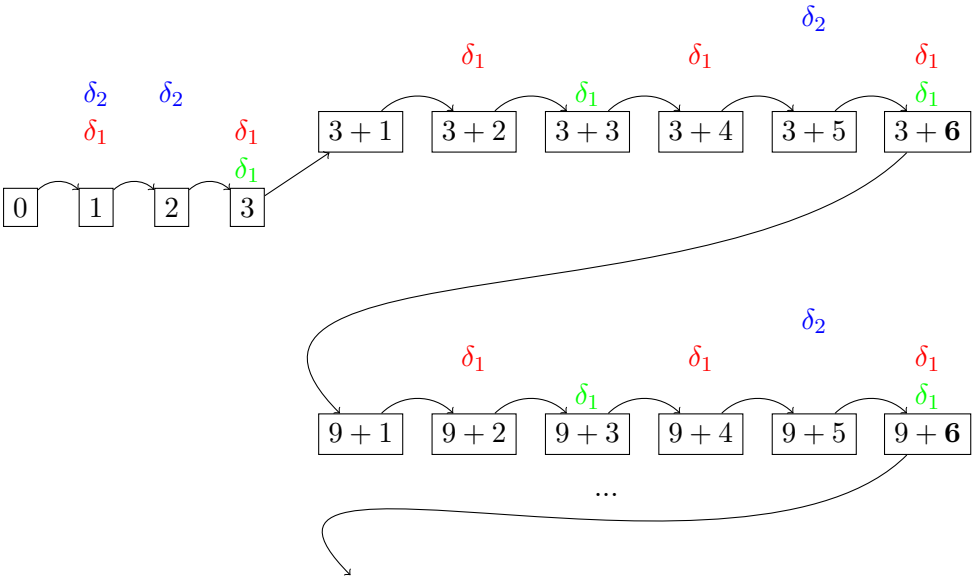


Figure 7.2: A labeling of each index assuming that the arithmetical progressions for δ_1 are $2\mathsf{k}+1$ and $3\mathsf{k}+3$ and the arithmetical progression for δ_2 is $6\mathsf{k}+2$. Furthermore, index 1 is labeled with δ_2 as I_{δ_2} has an exceptional zero. Note how any possible irregularities are discarded by setting K as large as necessary to consider all the exceptional zeroes and the starting point of all arithmetical progressions; also note that all arithmetical progressions repeat with period 6.

We start by showing that the labeling function used in the example is (eventually) periodic with period T, given by the least common multiple of all the periods of the arithmetical progressions, after an initial segment of irregularities with size at most K, given by the maximum of all the residues from the arithmetical progressions and the exceptional zeroes.

Lemma 7.3.2. *Let $\delta_1, \delta_2 \in$ EPLTL. Given a Markov chain (M, μ), let I_{δ_i} be the satisfaction sets of δ_i and suppose that there exist representations (F_i, G_i) for them. Then $f : \mathbb{N} \to \mathcal{P}(\{\delta_1, \delta_2\})$ defined by $\delta_i \in f(x)$ iff $x \in I_{\delta_i}$ is eventually periodic.*

Proof. Assume that $G_1 = \{(p_1, r_1), \ldots, (p_n, r_n)\}$, $G_2 = \{(q_1, s_1), \ldots, (q_m, s_m)\}$. Let $K = \max(F_1 \cup F_2 \cup \{r_1, \ldots, r_n\} \cup \{s_1, \ldots, s_m\})$, $T = lcm(p_1, \ldots p_n, q_1, \ldots q_m)$. Then for all natural numbers $n > K$, we need to prove that $f(n+T) = f(n)$.

Suppose that $\delta_1 \in f(n)$; then n must be of the form $p_i\mathsf{k} + r_i$, since $n > K$. Then $n + T = p_i\mathsf{k} + r_i + T = p_i(\frac{T}{p_i} + \mathsf{k}) + r_i$, because $p_i|T$. Therefore $\delta_1 \in f(n + T)$. The same argument can be applied to δ_2. □

Consider the following algorithm to compute a representation for $I_{\delta_1 \mathsf{U} \delta_2}$ given representations for I_{δ_1} and I_{δ_2}, and the function defined as above:

Lemma 7.3.3. *Let $\delta_1, \delta_2 \in$ EPLTL.*

Assume that $(F_1, G_1 = \{(p_1, r_1), \ldots (p_n, r_n)\})$ is a representation for I_{δ_1} and $(F_2, G_2 = \{(q_1, s_1), \ldots (q_m, s_m)\})$ is a representation for I_{δ_2}. Then $(F, G = (T, a_1), \ldots (T, a_o))$ computed using Algorithm 7 is a representation for $I_{\delta_1 \mathsf{U} \delta_2}$.

Proof. We note that Algorithm 7 halts for any input. We start by showing that if $n \in \mathbb{N}$ is represented in (F, G) then $n \in I_{\delta_1 \mathsf{U} \delta_2}$.

Suppose that $n \in F$. Then either $\delta_2 \in f(n)$ or $\delta_1 \in f(n)$ and there exists $n \leq m \leq T + K$ s.t $\delta_2 \in f(m)$. These are the only possible conditions in Algorithm 7 where n could have been added to F.

- if $\delta_2 \in f(n)$, by the definition of f in Lemma 7.3.2, we know that $n \in I_{\delta_2}$, and as such $n \in I_{\delta_1 \mathsf{U} \delta_2}$.

- if $\delta_1 \in f(n)$ and $n \leq m \leq T + K$ s.t $\delta_2 \in f(m)$, we also know that all indexes l with $n \leq l < m$ are such that $\delta_1 \in f(l)$. Therefore, by the definition of f in Lemma 7.3.2 we obtain that $n \in I_{\delta_1}$, $l \in I_{\delta_1}$ and $m \in I_{\delta_2}$; as such $n \in I_{\delta_1 \mathsf{U} \delta_2}$.

Now suppose that $n = T\mathsf{k} + a_1$, that is the pair (T, a_1) was added to G; in any of the three possible commands where we might have added (T, a_1), we are guaranteed that $K < a_1 \leq T + K$.

- suppose that $\delta_2 \in f(a_1)$. Then, we automatically know that $\delta_2 \in f(a_1 + T\mathsf{k})$, as f is periodic after K, with period T. But if $\delta_2 \in f(a_1)$ then $a_1 \in I_{\delta_2}$, $a_1 + T\mathsf{k} \in I_{\delta_2}$ and therefore $n \in I_{\delta_1 \mathsf{U} \delta_2}$.

- otherwise suppose that $\delta_1 \in f(a_1)$ and that there exists an index m s.t $a_1 \leq m \leq T + K$ and $\delta_2 \in f(m)$. Again, we are assured that for any indexes l between a_1 and m, all of them are such that $\delta_1 \in f(l)$. Therefore $\delta_1 \in f(a_1 + T\mathsf{k})$, $\delta_1 \in f(l + T\mathsf{k})$, for $a_1 \leq l < m$, and $\delta_2 \in f(m + T\mathsf{k})$; so we can conclude that in fact $n \in I_{\delta_1 \mathsf{U} \delta_2}$.

Algorithm 7: UntilRepresentation computes a representation for the satisfaction set of a Until formula

Input: Representations $(F_1, G_1 = \{(p_1, r_1), \ldots, (p_n, r_n)\}), (F_2, G_2 = \{(q_1, s_1), \ldots, (q_n, s_n)\})$ for I_{δ_1}, I_{δ_2}

Output: A representation for $I_{\delta_1 \mathsf{U} \delta_2}$

$T \leftarrow lcm(p_1, \ldots, p_n, q_1, \ldots, q_n)$;
$K \leftarrow \max(\{r_1, \ldots, r_n\} \cup \{s_1, \ldots, s_n\} \cup F_1 \cup F_2)$;
$L \leftarrow \{(0, f(0)), \ldots (T + K, f(T + K))\}$ (as in Lemma 7.3.2) ;
foreach $(i, f(i)) \in L$ **do**
  if $\delta_2 \in f(i)$ **then**
    if $i \leq K$ **then**
      $F \leftarrow F \cup \{i\}$;
    else
      $G \leftarrow G \cup \{(T, i)\}$;
    end
  else if $\delta_1 \in f(i)$ **then**
    $j \leftarrow$ **Search for** *the least* j *s.t.* $(j, f(j)) \in L, j \geq i, \delta_2 \in f(j)$;
    if *Search succeeds* **then**
      if $\delta_1 \in f(k)$ *for every* $i \leq k < j$ **then**
        if $i \leq K$ **then**
          $F \leftarrow F \cup \{i\}$;
        else
          $G \leftarrow G \cup \{(T, i)\}$;
        end
      end
    else
      $j \leftarrow$
      Search for *the least* j *s.t.* $(j, f(j)) \in L, j \geq K + 1, \delta_2 \in f(j)$;
      if *Search succeeds* **then**
        if $\delta_1 \in f(k)$ *for every*
        $k \in \{i, \ldots, T + K, K + 1, \ldots, j - 1\}$ **then**
          $G \leftarrow G \cup \{(T, i)\}$;
        end
      end
    end
  end
end
return (F, G);

- finally, the remaining possibility is the following: $\delta_1 \in f(a_1)$, and there exists $m \in \{K+1, \ldots, a_1 - 1\}$ such that $\delta_2 \in f(m)$; furthermore, we also know that $\delta_1 \in f(l)$ for any $l \in \{a_1, \ldots T+K\} \cup \{K+1, \ldots, m-1\}$. But then, $\delta_1 \in f(a_1), \ldots, \delta_1 \in f(T+K), \delta_1 \in f(T+K+1), \ldots \delta_1 \in f(T+K+m-1)$ and $\delta_2 \in f(T+K+m)$; by using the periodicity of f, we obtain that in fact $a_1 + T\mathsf{k} \in I_{\delta_1 \mathsf{U} \delta_2}$.

Suppose that $n \in I_{\delta_1 \mathsf{U} \delta_2}$. We show that n is represented in (F, G). If $n \in I_{\delta_1 \mathsf{U} \delta_2}$, then there exists $m \in I_{\delta_2}$, with $m \geq n$ and for all $n \leq k < m$, $k \in I_{\delta_1}$. Then, we know that $\delta_1 \in f(k)$, with $n \leq k < m$ and $\delta_2 \in f(m)$. Assuming that $n > K$, we can use the periodicity of f to guarantee that there exists n' s.t. $K < n' \leq T + K$,$\delta_1 \in f(n')$. The same argument can be applied to the index m, obtaining $m' \in \{K+1, \ldots, T+K\}$, $\delta_2 \in f(m')$. Afterwards we will always denote l' as the index obtained by the use of the periodicity of f applied to l in the range between $\{K+1, \ldots T+K\}$.

Suppose that $m' = n' + l$, $l \geq 0$. Then, consider the indexes $n+1, \ldots, n+l-1$; using the periodicity of f we obtain that their versions over the indexes $\{K+1, \ldots, T+K\}$, $(n+1)', \ldots, (n+l-1)'$ are such that all of them belong to I_{δ_1}; we conclude that then $n' \in I_{\delta_1 \mathsf{U} \delta_2}$ and our algorithm would have added to G the pair (T, n'), which represents n. However, it is possible that although $m \geq n$, their version over $\{K, \ldots, T+K\}$ may verify $n' > m'$. This case is only possible because there will exist an index k between n and m such that $k - K \mod T = 0$; in this case following a similar argument as above we can conclude that $n', \ldots, k' \in I_{\delta_1}$ and $k'+1 \ldots m'-1 \in I_{\delta_1}$; using the fact that $m' \in I_{\delta_2}$ we get that $n' \in I_{\delta_1 \mathsf{U} \delta_2}$. In this case the final condition of Algorithm 7 is fulfilled and so we in fact have added to G the pair (T, n'), which represents n. □

Theorem 7.3.1. *Let δ be an EPLTL formula, and (M, μ) a Markov chain. Assume that for any subterm $t_i = 0$ of δ, there exists a representation of $Z(\mu^T M^i t_i)$ which is (F_i, G_i). Then there exists (F, G) a representation for I_δ.*

Proof. The proof follows from using structural induction. If $\delta \equiv (t_i = 0)$, then by hypothesis we already have a representation for I_δ. So, for each of the connectives, we need to show that, assuming that the connectives' subformulae have representations (and therefore are unions of finite sets and arithmetical progressions), the satisfaction set for the connective will also be the union of a finite set and arithmetical progressions. Since the proofs for all connectives are constructive, we can obtain a representation for each of them. Using Proposition 7.3.1, it remains to prove the case of

the U connective. With the Algorithm 7, we obtain that we can construct a representation for any EPLTL formulae. □

Corollary 7.3.1. The verification problem of EPLTL formulae in Markov chains is equireducible to the Skolem problem.

7.4 Approximate Skolem Problem and verification of EPLTL formulae

As the Skolem problem is not known to be decidable, using it to verify EPLTL seems to be a doomed enterprise for the time being. However, in most applications, we are willing to accept results carrying a small error, at least to deal with finite precision representations. We intend on using the closure results for representations of EPLTL formulae for approximate model-checking of Markov chains. We will need a small adjustment to the syntax and semantics of EPLTL in order to cope with error bounds.

Definition 7.4.1. For any $\epsilon \in \mathbb{Q}$, the well formed formulae in ϵ-approximate EPLTL^ϵ over the propositional symbols Λ are given below in Backus-Naur notation:

$$\begin{aligned}
\beta &:= p \quad | \quad (\beta \wedge \beta) \quad | \quad (\neg\beta) \\
t &:= r \quad | \quad (t+t) \quad | \quad r \int \beta \\
\delta^\epsilon &:= \underbrace{\sim \delta^\epsilon \quad | \quad \delta^\epsilon \cap \delta^\epsilon \quad | \quad \mathsf{X}\delta^\epsilon \quad | \quad \delta^\epsilon \mathsf{U} \delta^\epsilon}_{\text{temporal formulae}} \quad | \quad \underbrace{(t=t)^\epsilon}_{\text{comparison formulae}}
\end{aligned}$$

where $p \in \Lambda$ and $r \in \mathbb{Q}$.

For an EPLTL formula δ and $\epsilon \in \mathbb{Q}$, we define the syntactic translation δ^ϵ of δ in the expected way. The denotation of terms is as in Definition 7.2.3 and the satisfaction relation is as expected:

Definition 7.4.2. The satisfaction relation $\Vdash^\epsilon$ between a Markov chain $\mathcal{M}$, an instant i, and a EPLTL^ϵ formula δ^ϵ is defined inductively:

- $\mathcal{M}, i \Vdash^\epsilon (t_1 = t_2)^\epsilon$ iff $|[\![t_1]\!]_{\mathcal{M},i} - [\![t_2]\!]_{\mathcal{M},i}| < \epsilon$,
- $\mathcal{M}, i \Vdash^\epsilon (\delta_1^\epsilon \cap \delta_2^\epsilon)$ iff $\mathcal{M}, i \Vdash^\epsilon \delta_1^\epsilon$ and $\mathcal{M}, i \Vdash^\epsilon \delta_2^\epsilon$,
- $\mathcal{M}, i \Vdash^\epsilon (\sim \delta^\epsilon)$ iff $\mathcal{M}, i \not\Vdash^\epsilon \delta$,

- $\mathcal{M}, i \Vdash^\epsilon (\mathsf{X}\delta^\epsilon)$ iff $\mathcal{M}, i+1 \Vdash^\epsilon \delta^\epsilon$,
- $\mathcal{M}, i \Vdash^\epsilon (\delta_1^\epsilon \mathsf{U} \delta_2^\epsilon)$ iff there exists $j \geq i$ such that $\mathcal{M}, j \Vdash^\epsilon \delta_2^\epsilon$ and $\mathcal{M}, k \Vdash^\epsilon \delta_1^\epsilon$, for all $i \leq k < j$.

In this section our main goal is to prove the following result:

Theorem 7.4.1. *For any $\delta \in$ EPLTL and Markov chain $\mathcal{M}$, there exists a computable error margin $d(\delta, \mathcal{M})$ such that for all $\epsilon \in]0; d[\cap\mathbb{Q}$ we can decide whether $\mathcal{M} \Vdash^\epsilon \delta^\epsilon$.*

In order to do so we will first prove the following result:

Theorem 7.4.2. *For any Markov chain $\mathcal{M} = (M, \mu)$, and any rational vector y, there exists a computable error margin d such that for all $\epsilon \in]0; d[\cap\mathbb{Q}$ we can decide whether there exists $i \in \mathbb{N}$ such that $-\epsilon < \mu^T M^i y < \epsilon$.*

Given a rational term t and a Markov chain $\mathcal{M} = (M, \mu)$, we will be interested in characterizing the set $\{i \in \mathbb{N} : -\epsilon < \mu^T M^i t < \epsilon\}$ for a suitable ϵ. In fact, this set will be the union of a finite set and a finite number of arithmetical progressions. However, unlike in the Skolem problem, we can actually compute a representation for this set (for suitable $\epsilon \in \mathbb{Q}$). Therefore, using the results already proven in last section about the temporal and propositional connectives, it will follow that the index set of δ^ϵ will also be the union of a finite set and a finite number of arithmetical progressions.

We will first show how to characterize the set $\{i \in \mathbb{N} : -\epsilon < \mu^T M^i t < \epsilon\}$, for a suitable precision ϵ. Note first that, using Jordan decomposition:

$$\begin{aligned}\mu^T M^n t &= \mu^T (SDS^{-1})^n t^* \\ &= \mu^T S D^n S^{-1} t^* \\ &= \sum_j p_j(n) \lambda_j^n\end{aligned}$$

The polynomials $p_j(n)$ have degree bounded by the size of the matrix. Notice that these polynomials will, in general, be complex and that the eigenvalues may also be complex; however one needs to remember that, in the end, the sum must still be a rational value.

Perron-Frobënius Theorem applied to irreducible stochastic matrices allows us to state the following:

- All eigenvalues verify $|\lambda_j| \leq 1$;
- There exists at least one eigenvalue j such that $\lambda_j = 1$;

- Other eigenvalues of absolute value 1 are all the roots of 1 for some degree.

In general, our stochastic matrix M is not necessarily irreducible; however, if we rewrite it using permutation matrices, we can obtain a matrix in upper-triangular block form such that:

- $PMP^{-1} = \begin{pmatrix} B_1 & ... & ... & ... \\ \mathbf{0} & B_2 & ... & ... \\ \mathbf{0} & \mathbf{0} & ... & ... \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & B_n \end{pmatrix}$
- Each matrix B_i is either stochastic and irreducible, or substochastic and irreducible.

The spectrum of M is the union of the spectra of each component B_i; therefore applying Perron Frobënius for each B_i, we conclude that the spectrum of M can be divided in several sets of all roots of unity for some degrees (which have absolute value 1), and all other eigenvalues (which have absolute value less than 1).

Then, we can expand the summation as follows:

$$\begin{aligned} \mu^T M^n t &= \sum_j p_j(n)\lambda_j^n \\ &= \sum_{\{j:|\lambda_j|=1\}} p_j(n)\lambda_j^n + \sum_{\{j:|\lambda_j|<1\}} p_j(n)\lambda_j^n \\ &= \sum_{\{j:|\lambda_j|=1\}} p_j(n) r_j^n e^{i\theta_j n} + \sum_{\{j:|\lambda_j|<1\}} p_j(n) r_j^n e^{i\theta_j n} \\ &= \underbrace{\sum_{\{j:|\lambda_j|=1\}} p_j(n) e^{2\pi i \frac{p_j}{q_j} n}}_{P(n)} + \underbrace{\sum_{\{j:|\lambda_j|<1\}} p_j(n) r_j^n e^{i\theta_j n}}_{D(n)}. \end{aligned}$$

Lemma 7.4.1. *Let $\mathcal{M} = (M, \mu)$ be a Markov chain, t^* the vector associated with* EPLTL *term t (which we will overload as in Remark 7.3.1). Then $P(n)$, and $D(n)$, defined as above are such that:*

- *$P(n)$ is periodic,*
- *$|D(n)| \leq q(n)R^n$, for some $0 \leq R < 1$ and polynomial $q(n)$, such that $q(n)R^n$ is monotonically decreasing after some $m \in \mathbb{N}$.*

- $P(n), D(n) \in \mathbb{R}$.

Proof. We start by showing that $|D(n)| \leq q(n)R^n$, for some $0 \leq R < 1$ and polynomial $q(n)$:

$$\begin{aligned} |D(n)| &= |\sum_{\{j:|\lambda_j|<1\}} p_j(n) r_j^n e^{i\theta_j n}| \\ &\leq \sum_{\{j:|\lambda_j|<1\}} |p_j(n)| r_j^n \\ &\leq q(n)R^n. \end{aligned}$$

With, for instance $q(n) = c(n^d + 1)$, for $d = \max \deg(p_j(n))$ and sufficiently large $c > 0$, and $R \in]0;1[\cap \mathbb{Q}$ chosen s.t. $r_j < R$; therefore not only we can compute m s.t. $q(n)R^n$ is monotonically decreasing but also $\lim |D(n)| = 0$.

In order to prove the other two assertions, let T be the least common multiple of all the denominators of the roots of unity in $P(n)$. Furthermore, let $P_s(k)$ and $D_s(k)$ with $s \in \{0, 1, \ldots, T-1\}$ be defined as:

$$P_s(k) = P(s + Tk), D_s(k) = D(s + Tk), k \in \mathbb{N}.$$

The first thing to note is that for each $s \in \{0, 1, \ldots, T-1\}$, $P_s(k)$ is a complex polynomial on k:

$$\begin{aligned} P_s(k) &= \sum_{\{j:|\lambda_j|=1\}} p_j(s + Tk) e^{2\pi i \frac{p_j}{q_j}(s+Tk)}, \\ &= \sum_{\{j:|\lambda_j|=1\}} p_j(s + Tk) e^{2\pi i \frac{p_j}{q_j} s} e^{2\pi i \frac{p_j}{q_j} Tk}, \\ &= \sum_{\{j:|\lambda_j|=1\}} p_j(s + Tk) e^{2\pi i \frac{p_j}{q_j} s}, \\ &= \sum_{\{j:|\lambda_j|=1\}} p_j(s + Tk) e^{2\pi i \frac{p_j}{q_j} s}. \end{aligned}$$

Note that $p_j(s+Tk)$ are complex polynomials on k, $e^{2\pi i \frac{p_j}{q_j} s}$ are complex constants, and therefore $P_s(k)$ is a complex polynomial on k, too. So, in fact $P_s(k) = \sum a_j k^j + i \sum b_j k^j$ for some real coefficients a_j, b_j. Now, for $k \in \mathbb{N}$, the imaginary part of $P_s(k)$, $\sum b_j k^j$, is a real polynomial on k.

However, we know that $\lim D_s(k) = 0$ and so both $\lim \mathrm{Re}(D_s(k)) = 0$ and $\lim \mathrm{Im}(D_s(k)) = 0$. Since

$$\begin{aligned} Im(\mu^T M^{s+Tk} t) &= Im(P_s(k)) + Im(D_s(k)) \\ &= \sum b_j k^j + Im(D_s(k)) \\ &= 0 \end{aligned}$$

then $\lim \sum b_j k^j = 0$. Given that $\lim \sum b_j k^j$ is a real polynomial that tends to 0, it must be 0. Therefore $P_s(k)$ is a real polynomial on k. This also entails that $D_s(k)$ has to be real (algebraic) valued.

We will now prove that $P_s(k)$ is in fact constant. So far we have shown that $P_s(k)$ is a real sequence. Suppose $P_s(k)$ is not a constant; then $\lim |P_s(k)| = \infty$. Naturally, this implies that $\lim |P_s(k) + D_s(k)| = \infty$, since $\lim |D_s(k)| = 0$. Note, however, that $|\mu^T M^n t| = |(\mu^t M^n).t)| \leq L||t||_1$, with L the size of the matrix, because $(\mu^T M^n)$ is a probabilistic vector. This implies that $P_s(k)$ is bounded and therefore, being a polynomial, has to be constant;

This argument holds for each s and as such, the whole sequence $P(n)$ has to be periodic. □

We now show that we can characterize the set $\{n \in \mathbb{N} : -\epsilon < \mu^T M^n t < \epsilon\}$:

Theorem 7.4.3. *There exists d, dependent on the Markov chain $\mathcal{M} = (M, \mu)$, and term t, such that, for all $\epsilon \in]0, d[$ we can compute a representation (F, G) for $\{n \in \mathbb{N} : -\epsilon < \mu^T M^n t < \epsilon\}$.*

Proof. Consider $P(n)$ and $D(n)$ as computed above. We are interested in characterizing the set $\{n \in \mathbb{N} : -\epsilon < P(n) + D(n) < \epsilon\}$. Recalling that $P(n)$ is periodic with a certain period T (computed in Lemma 7.4.1), we will consider $P_s(k) = P(s + Tk)$ and $D_s(k) = D(s + Tk)$, for $s \in \{0, \ldots, T-1\}$. We will also compute $q(n)$ and R to be such that $|D(n)| < q(n)R^n$ (as in Lemma 7.4.1). Moreover, we know that $q(n)R^n$ is decreasing after some index m.

Set $d = \min_{s \in \{0,\ldots,T-1\}} \{|P_s(0)| : P_s(0) \neq 0\}$; for any $\epsilon \in]0; d[$ and for any residue class s, there are two possibilities: either $P_s(0) = 0$, (in which case $P_s(0) = P_s(k) \in]-\epsilon, \epsilon[$, for all k), or $P_s(0) = P_s(k) \notin]-\epsilon, \epsilon[$ for any k. Now, we will collect the indices such that $\{k \in \mathbb{N} : -\epsilon < P_s(k) + D_s(k) < \epsilon\}$ for each residue class s.

Suppose that the residue class of s does not lie within $]-\epsilon, \epsilon[$. We note that then for almost all k, $P_s(k) + D_s(k) \notin]-\epsilon, \epsilon[$, since $\lim(P_s(k) + D_s(k)) =$

$P_s(0)$; so, it is a simple matter of collecting all the indices in this class of residues that fall into $]-\epsilon,\epsilon[$ until $s+Tk > m$ (as in Lemma 7.4.1) and $q(s+Tk)R^{s+Tk} < |P_s(0)-\epsilon|$. In this case, the indices so collected are added to the finite part of the representation F, and we are guaranteed that no more exceptional zeroes may occur in this residue class.

Suppose that the residue class of s lies within $]-\epsilon,\epsilon[$, or equivalently $P_s(0) = 0$. Then, we know that, after m, when $q(s+Tk)R^{s+Tk} < \epsilon$, then all the subsequent indices ($i > k$) will lie inside $]-\epsilon,\epsilon[$. In this case, we compute this index k and add the arithmetical progression $(T, Tk+s)$ to the arithmetical part of the representation, G. Furthermore, from 0 to k we collect the indices i that lie inside $]-\epsilon,\epsilon[$ and add $Ti+s$ to the finite part of the representation, F. □

With this result, we can now use the results about the characterizations of satisfaction sets from the preceding section to verify EPLTL^ϵ formulae. One must take into account that in the case of multiples comparison terms $t_i = 0$, the choice of ϵ must be smaller than the minimum of all d computed in Theorem 7.4.3.

Theorem 7.4.4. *Given a Markov chain (M,μ), and formula $\delta \in \mathsf{EPLTL}$, there exists $d > 0$, such that for all $\epsilon \in]0,d[\cap \mathbb{Q}$ we can decide $(M,\mu) \Vdash^\epsilon \delta^\epsilon$.*

Extending EPLTL with inequalities

Natural extensions for EPLTL and EPLTL^ϵ, already considered in [9], is to allow comparison formulae with inequalities ($t_1 > t_2$, resp. $(t_1 > t_2)^\epsilon$ and $t_1 < t_2$, resp. $(t_1 < t_2)^\epsilon$) endowed with the expected semantics.

In both cases, if we have representations (F,G) for the sets of indices where the comparison terms hold, we can use the closure under temporal connectives results presented in Section 7.3 to verify extended EPLTL formulae.

Regarding the approximate case, we note that the problem reduces to that of characterizing sets of the form $\{n \in \mathbb{N} : -\epsilon < \mu^T M^n t\}$ and $\{n \in \mathbb{N} : \mu^T M^n t < \epsilon\}$. The proof of Theorem 7.4.3 can be adapted for these cases. Without loss of generality we sketch the case for $t > -\epsilon$: we consider the same d; if $P_s(0)$ falls into the interval $]-\epsilon;+\infty[$ we know that almost always $P_s(k) + D_s(k)$ will belong to that interval. Therefore we would need to collect the exceptional indices that may appear up to the point after m, where $q(s+Tk)R^{s+Tk} < |P_s(0)-\epsilon|$, adding the arithmetical progression afterwards. Otherwise, if $P_s(0)$ falls in $]-\infty,-\epsilon]$, we note that we are

guaranteed that $P_s(0) \neq -\epsilon$ and we just need to compute $P_s(k) + D_s(k)$ up to the point where we are certain that the remaining sequence will be in $]-\infty, -\epsilon]$, which will happen when $q(s+Tk)R^{s+Tk} < \epsilon$.

For the non-approximate case, the question ties in with another well-known open problem related with the Skolem problem:

Proposition 7.4.1 (Positivity Problem)**.** *Given $x, y \in \mathbb{Q}^m$ and $L \in \mathbb{Q}^{m^2}$, decide if the following statement is true:*

$$\forall n. x^T L^n y > 0.$$

It is known [85] that if the Positivity problem is decidable so is the Skolem problem. However the converse of this result has not been established. One might be tempted to take the Positivity problem as a stronger oracle instead of the Skolem problem and to adapt the reasoning presented in Section 7.3. Unfortunately, this is not enough. Since we do not know if there are representations (F, G) for sets of the form $\{n \in \mathbb{N} : \mu^T M^n t > 0\}$ (in fact there are no known characterizations for such sets) the arguments of Section 7.3 would not follow through.

7.5 Conclusions and Future Work

In this work we have shown the connection between the Skolem problem and the verification problem for temporal properties on probabilistic dynamical systems induced by Markov chains. Since significant advances on the decidability (or lack thereof) of the Skolem problem seem unlikely in the near future [85], we turned our attention towards approximate versions of these problems. In this context, we have presented procedures to decide the problems.

A natural progression for this work would be an implementation of the verification algorithm for the approximate version of the problem. This approach ties to a study of the computational complexity of the problems at hand since, in this work, we were more concerned with clarity of exposition of the procedures rather than efficiency.

In this work, we have provided a Turing reduction between the verification problem of a EPLTL formula over a Markov chain, and the Skolem problem.

Let

$$\mathsf{MC} = \{< M, \delta >: \text{ M is Markov chain, } \delta \text{ is EPLTL formula, and } M \Vdash \delta\}$$

and $\mathsf{SK} = \{< x, y, M >: \exists n. x^T M^n y = 0\}$, where x, y are rational vectors of appropriate dimensions, and M is a Markov chain. Now, note that MC is either a recursive set, or it is not even recursively enumerable. If it was recursively enumerable either $< M, \delta > \in \mathsf{MC}$ or $< M, \sim \delta > \in \mathsf{MC}$, therefore MC would be recursive. Note, furthermore, that SK is a recursively enumerable set, as it is an existential quantification of a recursive predicate. This implies that if one is able to prove the existence of a many-to-one reduction $\mathsf{MC} \leq \mathsf{SK}$, one would establish that MC would have to be recursively enumerable, which in turn would imply the decidability of the verification problem and consequently the decidability of the Skolem problem, as a many-to-one reduction of the Skolem problem to the verification problem is easy to obtain.

Chapter 8

Propositional Quantification in EPLTL

In this Chapter, we extend the (linear) temporalization of EPPL with propositional quantifiers. We capitalize on the results shown for propositional quantification in EPPL, namely that EPPL admits quantifier elimination. While we do not hope to be able to apply the satisfiability algorithm for QEPLTL proposed herein, we believe that the results obtained can be generalized greatly. Furthermore, there are some ideas with practical relevance, related with Chapter 6. For any LTL formula there exists a way of describing succinctly each and every model that satisfies the formula; in LTL this can be done by finite state automata with Büchi or Rabin accepting conditions. This work is also an application of work done by **Franceschet et al.** [41, 42], further extended due to propositional quantification.

This Chapter concerns an adaptation of such techniques, based on the idea that instead of having transitions labeled with propositional valuations, we will have transitions labeled with valuations over EPPL literals. These labels can be seen in two distinct ways: either as propositional valuations, or either as EPPL formulae that must be satisfied by the EPLTL model in order to "use" the transition.

8.1 Syntax and Semantics of Quantified Exogenous Probabilistic Linear Temporal Logic

The syntax of QEPLTL is an extension of EPLTL. Our QEPLTL formulae are given in prenex normal form. This follows the original presentation of **Sistla** [94].

Definition 8.1.1. The *well-formed formulae* in $\mathsf{QEPLTL}(\Lambda)$ over the set of variables Z are given below in Backus-Naur notation:

$$\begin{aligned}
\beta &:= p \quad | \quad (\beta \wedge \beta) \quad | \quad (\neg\beta) \quad | \quad \top \\
t &:= r \quad | \quad \int \beta \quad | \quad (t.t) \quad | \quad (t+t) \quad | \quad z \\
\gamma &:= (\Box\beta) \quad | \quad (t \leq t) \quad | \quad (\sim\gamma) \quad | \quad (\gamma \cap \gamma) \quad | \quad (\mathsf{X}\gamma) \quad | \quad (\gamma \mathsf{U} \gamma) \quad | \quad \bot \\
\delta &:= \gamma \quad | \quad \exists p.\delta \quad | \quad \sim \delta
\end{aligned}$$

where $p \in \Lambda$, $r \in \mathbb{R}_{\mathsf{Alg}}$ and $z \in Z$.

We introduced the global $\bot$ constant for technical reasons, only. Furthermore, we consider the abbreviation $(\forall p.\delta) \equiv (\sim (\exists p.(\sim \delta)))$.

The semantics of QEPLTL are given with respect to a sequence of EPPL models, and as such they extend Definition 3.3.2. Two EPLTL models Σ, Ω are said to be $\sim_p$*-equivalent* if $\Sigma(i) \sim_p \Omega(i)$ for all $i \in \mathbb{N}$. This definition requires the definition of $\sim_p$ equivalence between probabilistic models given in Definition 5.2.1.

Definition 8.1.2. Let $\Sigma \in (\mathfrak{M}(\mathsf{EPPL}(\Lambda)) \times \mathbb{R}_{\mathsf{Alg}}^{Z})^{\omega}$, and let $\delta \in \mathsf{QEPLTL}(\Lambda)$. The satisfaction relation between a EPLTL model Σ and a QEPLTL formula δ, represented by $\Sigma \Vdash^{\mathsf{QEPLTL}} \delta$, is defined below:

- $\delta \equiv \Box\beta$: $\Sigma \Vdash^{\mathsf{QEPLTL}} \Box\beta$ iff $\Sigma(0) \Vdash^{\mathsf{EPPL}} \beta$;
- $\delta \equiv t_1 \leq t_2$: $\Sigma \Vdash^{\mathsf{QEPLTL}} t_1 \leq t_2$ iff $[\![t_1]\!]_{\Sigma(0)} \leq [\![t_2]\!]_{\Sigma(0)}$;
- $\delta \equiv \bot$: $\Sigma \nVdash^{\mathsf{QEPLTL}} \bot$;
- $\delta \equiv \gamma_1 \cap \gamma_2$: $\Sigma \Vdash^{\mathsf{QEPLTL}} \gamma_1 \cap \gamma_2$ iff $\Sigma \Vdash^{\mathsf{QEPLTL}} \gamma_1$ and $\Sigma \Vdash^{\mathsf{QEPLTL}} \gamma_2$;
- $\delta \equiv \sim \gamma$: $\Sigma \Vdash^{\mathsf{QEPLTL}} \sim \gamma$ iff $\Sigma \nVdash^{\mathsf{QEPLTL}} \gamma$;
- $\delta \equiv \mathsf{X}\gamma$: $\Sigma \Vdash^{\mathsf{QEPLTL}} \mathsf{X}\gamma$ iff $\Sigma[1] \Vdash^{\mathsf{QEPLTL}} \gamma$;
- $\delta \equiv \gamma_1 \mathsf{U} \gamma_2$: $\Sigma \Vdash^{\mathsf{QEPLTL}} \gamma_1 \mathsf{U} \gamma_2$ iff there exists $j \in \mathbb{N}$ such that $\Sigma[j] \Vdash^{\mathsf{QEPLTL}} \gamma_2$ and for all $0 \leq i < j$, $\Sigma[i] \Vdash^{\mathsf{QEPLTL}} \gamma_1$;
- $\delta \equiv \exists p.\delta$: $\Sigma \Vdash^{\mathsf{QEPLTL}} \exists p.\delta$ iff there exists Σ' such that $\Sigma' \sim_p \Sigma$ and $\Sigma' \Vdash^{\mathsf{QEPLTL}} \delta$.
- $\delta \equiv \sim \delta_1$: $\Sigma \Vdash^{\mathsf{QEPLTL}} \sim \delta_1$ iff $\Sigma \nVdash^{\mathsf{QEPLTL}} \delta_1$.

We note that EPLTL can not express alternating properties like "p holds almost surely in all even instants", by adapting the proof of **Wolper** [106].

Lemma 8.1.1. *Let $(t_1 \leq t_2)$ be a comparison term, and let $m_1 = (\mathcal{P}, \rho), m_2 = (\mathcal{Q}, \xi)$ be such that $(\mathcal{P}, \rho) \Vdash^{EPPL} (t_1 \leq t_2)$ and $(\mathcal{Q}, \rho) \nVdash^{EPPL} (t_1 \leq t_2)$. Furthermore, let δ be a EPLTL formula with n X connectives.*

Then, $m_1^i m_2 m_1^\omega \Vdash^{EPLTL} \delta$ iff $m_1^k m_2 m_1^\omega \Vdash^{EPLTL} \delta$ for any $k, i > n$.

Proof. The proof follows the same argument as **Wolper** [106], but we show it for completeness sake. Let $|\delta|_i$ represent whether $m_1^i m_2 m_1^\omega \Vdash^{\mathsf{EPLTL}} \delta$ holds. We wish to prove that $|\delta|_i{=}|\delta|_k$, for any $k, i > n$ where n is the number of X connectives in δ.

Suppose that δ is a comparison term, or $\Box\beta$ or $\bot$; in either case $|\delta|_i{=}|\delta|_k$, because both sequences have the same first model.

If $\delta \equiv\sim \delta_1$, $|\delta|_i = 1$ iff $|\delta_1|_i = 0$; by the induction hypothesis, we get $|\delta_1|_k = 0$ iff $|\delta|_k = 1$. The cases for the other propositional connectives are similar.

Now let $\delta \equiv \mathsf{X}\delta_1$. Then $|\delta|_i = 1$ iff $|\delta_1|_{i-1} = 1$, and since $i - 1 > n - 1$ (which is the number of X connectives on δ_1), we can apply the induction hypothesis to conclude that $|\delta_1|_{k-1} = 1$ which is equivalent to $|\mathsf{X}\delta_1|_k = 1$.

Finally, consider $\delta \equiv \delta_1 \mathsf{U} \delta_2$; in this case $|\delta|_i = |\delta_2|_i \vee (|\delta_1|_i \wedge |\delta|_{i-1})$, which can be further expanded into:

$$|\delta|_i = |\delta_2|_i \vee (|\delta_1|_i \wedge (|\delta_2|_{i-1} \vee (|\delta_1|_{i-1} \wedge (\ldots \wedge (|\delta_2|_{n+1} \vee (|\delta_1|_{n+1} \wedge |\delta|_n)) \ldots)))).$$

Now, applying the induction hypothesis to each $|\delta_1|_j$ and $|\delta_2|_j$, with $n + 1 < j \leq i$ is enough to conclude that $|\delta|_i = (|\delta_2|_{n+1} \vee (|\delta_1|_{n+1} \wedge |\delta|_n))$; by repeating the same argument with k, we conclude that $|\delta|_i = |\delta|_k$. □

Let $\mathsf{G}_k(t_1 \leq t_2)$ represent the property that $t_1 \leq t_2$ holds in all instants divisible by k, where $(t_1 \leq t_2)$ is a satisfiable but not valid comparison term.

Proposition 8.1.1. *Property $\mathsf{G}_k(t_1 \leq t_2)$ can not be expressed in EPLTL, for $k > 1$.*

Proof. Let δ be a EPLTL formula. Then δ has a finite number of X connectives, denoted by l. Consider the EPLTL models introduced in the previous proof, $m_1^{qk} m_2 m_1^\omega$ and $m_1^{qk-1} m_2 m_1^\omega$. Note that the property G_k has to hold only on $m_1^{qk} m_2 m_1^\omega$, for any choice of $q \in \mathbb{N}^+$. Choose q such that $qk - 1 > l$. By the previous proof $|\delta|_{qk-1} = |\delta|_{qk}$ and as such δ does not express G_k. □

However, by adapting the proof shown in Example 1.1.1 it can be seen

that $\mathsf{G}_2(t_1 \leq t_2)$ may be expressed precisely by:

$$\mathsf{G}_2(t_1 \leq t_2) \equiv \exists q.(\int q = 1) \cap (\mathsf{X} \sim (\int q = 1)) \cap$$
$$\mathsf{G}(\int q = 1 \Leftrightarrow \mathsf{XX}(\int q = 1)) \cap$$
$$\mathsf{G}((\int q = 1) \supset (t_1 \leq t_2)).$$

8.2 Büchi automata for EPLTL models

The fact that the axiom system for μ-EPPL considers as axioms any EPPL instantiation of μ-TL axioms provide us with an important clue about the structure of valid properties in EPLTL. The temporalization of EPPL does not permit temporal restrictions that do not emerge from the temporal connectives, since the EPPL reasoning is completely local. This should not come as a surprise since $x = 0 \supset (x = 0 \cap (\mathsf{X}x = 0))$ is not a valid formula, as the assignment of variables is local.

This fact is not unique in EPLTL; in fact any (linear) temporalization, following the framework of **Finger et al.**, follows the same perspective. Therefore, in order to decide the satisfiability of EPLTL formulae, one needs simply to consider its LTL version over the set of its literals, seen as propositional symbols, and then test the consistency of valuations over the set of literals.

This Section generalizes Büchi automata as a method for describing all the satisfying models of a EPLTL formula. Therefore, in order to decide the satisfiability of a EPLTL formula, one simply needs to use the classical automata algorithms to decide the emptiness of the language of the Büchi automaton that represents the formula.

Let Lit be a finite set of literals of EPPL. Given a valuation $v \in \{0,1\}^{Lit}$, let:

$$\delta_v \equiv \bigcap_{\substack{lit \in Lit \\ v(lit) = 1}} lit \cap \bigcap_{\substack{lit \in Lit \\ v(lit) = 0}} \sim lit.$$

Now, given any Büchi automaton $N = (S, \{0,1\}^{Lit}, \tau, s^0, F)$ we define three languages; each language corresponds to a different perspective on N.

Definition 8.2.1. Let $N = (S, \{0,1\}^{Lit}, \tau, s^0, F)$ be a Büchi automaton; we define the *syntactic language* of N, denoted by $L_{\mathsf{Syn}}(N)$, the *syntactically*

consistent language of N, denoted by $L_{\mathsf{SynCon}}(N)$ and the *semantic language* of N denoted by $L_{\mathsf{Sem}}(N)$:

- $L_{\mathsf{Syn}}(N) = \{\sigma \in (\{0,1\}^{Lit})^\omega : \exists \rho \in \tau^\omega \text{ s.t. } \rho \text{ is accepting run, } trace(\rho) = \sigma.\}$;
- $L_{\mathsf{SynCon}}(N) = \{\sigma \in (\{0,1\}^{Lit})^\omega : \exists \rho \in \tau^\omega \text{ s.t. } \rho \text{ is accepting run, } trace(\rho) = \sigma, \text{ and for any } i \in \mathbb{N}, \delta_{v_i} \text{ is } \mathsf{EPPL}\text{-satisfiable, where } \rho(i) = (s_i, v_i, s_{i+1}).\}$;
- $L_{\mathsf{Sem}}(N) = \{\Sigma \in (\mathfrak{M}(\mathsf{EPPL}) \times \mathbb{R}^Z_{\mathsf{Alg}})^\omega : \exists \rho \in \tau^\omega \text{ s.t. } \rho \text{ is accepting run, } trace(\rho) = \sigma, \text{ and for any } i \in \mathbb{N}, \Sigma_i \Vdash^{\mathsf{EPPL}} \delta_{\sigma_i}.\}$.

These three languages provide us with different viewpoints about an automaton N over a set of literals. The syntactic language of N is the classical definition of the language accepted by a Büchi automaton. In terms of EPLTL formulae, it corresponds to the models that satisfy the EPLTL formulae when seen as a LTL formulae over its set of literals.

The syntactically consistent language of N is the subset of the syntactic language of N that acknowledges that in fact its set of literals are not propositional symbols, because there are relations between them; therefore, in order to use a transition from the Büchi automaton N, one needs to make sure that the transition is consistent, as otherwise it can not be used.

The semantic language of N is then the set of EPLTL models that are accepted by N; it has a strong relation with the syntactically consistent language of N, as seen in Lemma 8.2.3.

Definition 8.2.2. Let Lit be a finite set of literals of EPPL. Then the set $Con^\omega(Lit) = \{\sigma \in (\{0,1\}^{Lit})^\omega : \delta_{\sigma_i} \text{ is } \mathsf{EPPL}\text{-satisfiable.}\}$ is named the *consistency set of Lit*.

Lemma 8.2.1. *Let $N = (S, \{0,1\}^{Lit}, \tau, s_0, F)$ be a Büchi automaton; and let $\overline{N}$ be the complementary Büchi automaton given by Proposition 1.1.3, such that $L_{Syn}(\overline{N}) = \overline{L_{Syn}(N)}$:*

1. $L_{SynCon}(N) = L_{Syn}(N) \cap Con^\omega(Lit)$;
2. $\overline{L_{SynCon}(N)} \cap Con(Lit) = L_{Syn}(\overline{N}) \cap Con^\omega(Lit)$.

Proof. Suppose that $\sigma \in L_{\mathsf{SynCon}}(N)$; then clearly $\sigma \in Con^\omega(Lit)$ and $\sigma \in L_{\mathsf{Syn}}(N)$; suppose that $\sigma \in L_{\mathsf{Syn}}(N)$ and $\sigma \in Con^\omega(Lit)$; in this case the accepting run ρ that witnesses $\sigma \in L_{\mathsf{Syn}}(N)$ is only done by consistent transitions, and so $\sigma \in L_{\mathsf{SynCon}}(N)$.

For the second statement:

$$\begin{aligned}\overline{L_{\mathsf{SynCon}}(N)} \cap Con^{\omega}(Lit) &= \overline{L_{\mathsf{Syn}}(N) \cap Con^{\omega}(Lit)} \cap Con^{\omega}(Lit) \\ &= (\overline{L_{\mathsf{Syn}}(N)} \cup \overline{Con^{\omega}(Lit)}) \cap Con^{\omega}(Lit) \\ &= \overline{L_{\mathsf{Syn}}(N)} \cap Con^{\omega}(Lit) \\ &= L_{\mathsf{Syn}}(\overline{N}) \cap Con^{\omega}(Lit).\end{aligned}$$

□

We define a consistent Büchi automaton, where each inconsistent transition is eliminated.

Definition 8.2.3. Let $N = (S, \{0,1\}^{Lit}, \tau, s_0, F)$ be a Büchi automaton. The *consistent adaptation of* N is defined to be the Büchi automaton $N^{+} = (S, \{0,1\}^{Lit}, \tau^{+}, s_0, F)$, where $(s, v, t) \in \tau^{+}$ if $(s, v, t) \in \tau$ and δ_v is EPPL-satisfiable.

Lemma 8.2.2. *Let $N = (S, \{0,1\}^{Lit}, \tau, s_0, F)$ be a Büchi automaton. Then,*

$$L_{\mathsf{SynCon}}(N) = L_{\mathsf{SynCon}}(N^{+}) = L_{\mathsf{Syn}}(N^{+}).$$

We define $\epsilon : \mathfrak{M}(\mathsf{EPPL}) \times \mathbb{R}^{Z}_{\mathsf{Alg}} \mapsto \{0,1\}^{Lit}$ in the following way:

$$\epsilon((\mathcal{P}, \rho))(lit) = 1 \text{ iff } \mathcal{P}, \rho \Vdash^{\mathsf{EPPL}} lit,$$

with $lit \in Lit$. Furthermore, let the extension of ϵ to EPLTL models be denoted by ϵ^{*}

Lemma 8.2.3. *Let $N = (S, \{0,1\}^{Lit}, \tau, s_0, F)$ be a Büchi automaton; then:*

$$\Sigma \in L_{\mathsf{Sem}}(N) \text{ iff } \epsilon^{*}(\Sigma) \in L_{\mathsf{SynCon}}(N)$$

Proof. Suppose that $\Sigma \in L_{\mathsf{Sem}}(N)$.

Then there exists a run $\rho = (s_0, \sigma_0, s_1) \dots (s_i, \sigma_i, s_{i+1})$ which is accepting and is such that $\Sigma(i) \Vdash \delta_{\sigma_i}$ for all $i \in \mathbb{N}$; we can conclude immediately that $\sigma = trace(\rho) \in L_{\mathsf{SynCon}}(N)$, since each δ_{σ_i} is satisfiable. We now prove that $\epsilon^{*}(\Sigma) = \sigma$, that it $\epsilon(\Sigma(i)) = \sigma_i$. Given that

$$\Sigma(i) \Vdash^{\mathsf{EPPL}} \bigcap_{\substack{lit \in Lit \\ \sigma_i(lit) = 1}} lit \cap \bigcap_{\substack{lit \in Lit \\ \sigma_i(lit) = 0}} \sim lit,$$

we know that $\Sigma(i) \Vdash^{\mathsf{EPPL}} lit$ iff $\sigma_i(lit) = 1$; but then $\Sigma(i) \Vdash^{\mathsf{EPPL}} lit$ iff $\epsilon(\Sigma(i))(lit) = 1$, which ends the proof of $\Rightarrow$.

Suppose that $\epsilon^*(\Sigma) \in L_{\mathsf{SynCon}}(N)$.

Then, there exists an accepting run $\rho = (s_0, \epsilon(\Sigma(0)), s_1) \dots (s_i, \epsilon(\Sigma(i)), s_{i+1})$ such that each $\delta_{\epsilon(\Sigma(i))}$ is EPPL-satisfiable. We just need to prove that $\Sigma_i \Vdash^{\mathsf{EPPL}} \delta_{\epsilon(\Sigma(i))}$, that is:

$$\Sigma(i) \Vdash^{\mathsf{EPPL}} \bigcap_{\substack{lit \in Lit \\ \epsilon(\Sigma(i))(lit) = 1}} lit \cap \bigcap_{\substack{lit \in Lit \\ \epsilon(\Sigma(i))(lit) = 0}} \sim lit,$$

which is follows directly from the fact that $\epsilon(\Sigma(i))(lit) = 1$ iff $\Sigma_i \Vdash lit$. $\square$

These Lemmas already provide us with a satisfiability procedure for EPLTL. Given a formula δ, one collects its set of literals, and constructs the Büchi automaton that satisfies δ when seen a LTL formula over the set of literals. Afterwards, we test the consistency of each transition in the Büchi automaton, and finally, we simply need to test the emptiness of its syntactic language, using Theorem 1.1.2.

We now define transformation of a Büchi automaton that will capture the $\sim_p$ equivalence relation between EPLTL models. In the case of QLTL, the transformation is quite straightforward; for each transition, we project the valuation in order to "forget' its existential quantified propositional symbol, that is if $(s, (p \mapsto 1, q \mapsto 0, r \mapsto 1), t)$ is a transition in the Büchi automaton, the projected Büchi automaton will have as transition $(s, (q \mapsto 0, r \mapsto 1), t)$; or equivalently $(s, (p \mapsto 1, q \mapsto 0, r \mapsto 1), t)$ and $(s, (p \mapsto 0, q \mapsto 0, r \mapsto 1), t)$. This operation is due to the how the quantifier elimination technique occurs in PL. Recall that $(\exists p.\phi) \Leftrightarrow \lceil \phi \rceil^p_\top \vee \lceil \phi \rceil^p_\bot$. In our case, the projection must take into account a much more quantifier elimination technique.

Definition 8.2.4 (Projection of a Büchi automaton)**.** Let N be a Büchi automaton, with $N = (S, \{0,1\}^{Lit^0}, \tau, s_0, F)$. The *p-projection of the Büchi automaton* N is the Büchi automaton $p(N) = (S, \{0,1\}^{Lit^1}, \tau', s_0, F)$ where Lit^1 and τ' are defined as follows:

- For each $v \in \{0,1\}^{Lit^0}$, compute the set of literals of the EPPL formula obtained by propositional quantifier elimination of $\exists p.\delta_v$. We represent such set by $lit(QE(\exists p.\delta_v))$. The set Lit^1 is then the union of all the literals so obtained, that is $Lit^1 = \bigcup_{v \in \{0,1\}^{Lit^0}} lit(QE(\exists p.\delta_v))$.

- The transition relation τ' is defined as $(s, w, t) \in \tau'$, with $w \in \{0,1\}^{Lit^1}$, iff there exists $(s, v, t) \in \tau$ such that δ_w is a conjunct in the disjunctive normal form of $QE(\exists p.\delta_v)$, where each conjunct has to have all the literals in Lit^1 and has to be satisfiable if $QE(\exists p.\delta_v)$ is satisfiable; otherwise it is assumed to be the introduced global atom $\bot$.

Remark 8.2.1. If one sees $QE(\exists p.\delta_v)$ as a propositional formula over Lit^1, we could instead require that $w \Vdash^{\mathsf{PL}} QE(\exists p.\delta_v)$.

Proposition 8.2.1. *Let $N = (S, \{0,1\}^{Lit^0}, \tau, s_0, F)$ be a Büchi automaton. Then,*

- $L_{\mathsf{SynCon}}(p(N)) = L_{\mathsf{Syn}}(p(N))$,
- $\Sigma \in L_{\mathsf{Sem}}(p(N))$ *iff there exists a $\sim_p$ equivalent model Σ' such that $\Sigma' \in L_{\mathsf{Sem}}(N)$.*

Proof. Let $p(N) = (S, \{0,1\}^{Lit^1}, \tau', s_0, F)$. Let $\rho = (s_0, w_0, s_1)\ldots(s_i, w_i, s_{i+1})\ldots$ be an accepting run in $p(N)$. We just need to prove that δ_{w_i} is EPPL-satisfiable. Note that $(s_i, w_i, s_{i+1}) \in \tau'$ if δ_{w_i} is a conjunct in the disjunctive normal form of $QE(\exists p.\delta_v)$, for some $v \in \{0,1\}^{Lit^0}$ such that $(s, v, t) \in \tau$. But if the disjunctive normal form of $QE(\exists p.\delta_v)$ is not the formula representing falsum, each conjunct has to be satisfiable and as such δ_{w_i} in particular is satisfiable.

Suppose now that $\Sigma \in L_{\mathsf{Sem}}(p(N))$. Then there exists a run $\rho = (s_0, w_0, s_1)\ldots(s_i, w_i, s_{i+1})\ldots \in \tau'^{\omega}$ such that $\Sigma(i) \Vdash^{\mathsf{EPPL}} \delta_{w_i}$. If $(s_i, w_i, s_{i+1}) \in \tau'$, then δ_{w_i} is a conjunct in $QE(\exists p.\delta_v)$, with $(s_i, v, s_{i+1}) \in \tau$. But if $\Sigma(i) \Vdash^{\mathsf{EPPL}} \delta_{w_i}$, then $\Sigma(i) \Vdash^{\mathsf{EPPL}} QE(\exists p.\delta_v)$ which allows us to conclude that there exists $\Sigma'(i)$ which is $\sim_p$ equivalent to $\Sigma(i)$ such that $\Sigma'(i) \Vdash^{\mathsf{EPPL}} \delta_v$; therefore $\Sigma' \in L_{\mathsf{Sem}}(N)$.

Suppose that there exists a EPLTL model Σ' which is equivalent to Σ such that $\Sigma' \in L_{\mathsf{Sem}}(N)$. Then $\rho = (s_0, v_0, s_1)\ldots(s_i, v_i, s_{i+1}) \in \tau^{\omega}$ is an accepting run, and $\Sigma'(i) \Vdash^{\mathsf{EPPL}} \delta_{v_i}$. Then $\Sigma(i) \Vdash \exists p.\delta_{v_i}$ and as such it satisfies the quantifier elimination of δ_{v_i}, $\Sigma(i) \Vdash QE(\exists p.\delta_{v_i})$. But then, there has to exist at least one conjunct $QE(\exists p.\delta_{v_i})$ which is satisfiable by $\Sigma(i)$. Name it δ_{w_i}. Therefore not only the transition (s_i, w_i, s_{i+1}) has to exist but also $\Sigma(i) \Vdash \delta_{w_i}$, which allows to conclude that $\Sigma \in L_{\mathsf{Sem}}(p(N))$. □

8.3 Decidability of QEPLTL

Let δ be a EPLTL formula, and let $Lit(\delta)$ be its set of literals; furthermore let $\lambda(\delta)$ be the associated LTL formula where each literal in $Lit(\delta)$ is seen as propositional symbol.

Proposition 8.3.1. *Let δ be a EPLTL formula, $N = (S, \{0,1\}^{Lit(\delta)}, \tau, s_0, F)$ the Büchi automaton such that $L^\omega(N) = L^\omega(\lambda(\delta))$. Then $\Sigma \Vdash^{EPLTL} \delta$ iff $\Sigma \in L_{Sem}(N)$.*

Proof. We will show that $\Sigma \Vdash^{\mathsf{EPLTL}} \delta$ iff $\epsilon^*(\Sigma) \Vdash^{\mathsf{LTL}} \lambda(\delta)$. Then, $\epsilon^*(\Sigma) \Vdash^{\mathsf{LTL}} \lambda(\delta)$ iff $\epsilon^*(\Sigma) \in L_{\mathsf{SynCon}}(N)$ iff $\Sigma \in L_{\mathsf{Sem}}(N)$, due to Lemma 8.2.3.

The proof of $\Sigma \Vdash^{\mathsf{EPLTL}} \delta$ iff $\epsilon^*(\Sigma) \Vdash \lambda(\delta)$ follows from structural induction on δ. Suppose that $\delta \equiv (t_1 \leq t_2)$ as the other atomic case is similar. Then $\Sigma \Vdash^{\mathsf{EPLTL}} (t_1 \leq t_2)$ iff $\Sigma(0) \Vdash^{\mathsf{EPPL}} (t_1 \leq t_2)$ iff $\epsilon(\Sigma(0))((t_1 \leq t_2)) = 1$, by the definition of ϵ; but then $\epsilon(\Sigma(0)) \Vdash^{\mathsf{PL}} \lambda(\delta)$ and as such $\epsilon^*(\Sigma) \Vdash^{\mathsf{LTL}} \lambda(\delta)$.

All the structural induction steps are direct applications of the induction hypothesis. □

Theorem 8.3.1. *Let δ be a QEPLTL(Λ) formula. Then there exists a Büchi automaton N such that $\Sigma \Vdash^{QEPLTL} \delta$ iff $\Sigma \in L_{Sem}(N)$.*

Proof. The proof follows from induction on the number of blocks of quantifiers of the same type. Let $\delta \equiv \exists \ldots \forall \exists \gamma$, where $\gamma \in \mathsf{EPLTL}$. If there are 0 blocks of quantifiers, by Proposition 8.3.1, there exists a Büchi automaton N such that $\Sigma \Vdash^{\mathsf{QEPLTL}} \delta$ iff $\Sigma \in L_{\mathsf{Sem}}(N)$.

Now suppose that the number of blocks of quantifiers of the same type is $n+1$. Then either (1) $\delta \equiv \exists \delta_1$ with δ_1 having only n blocks of quantifiers of the same type, or (2) $\delta \equiv \forall \delta_1$.

We prove case (1), by induction on the number k of existential quantifiers. Suppose that $k = 0$; then the proof follows from the induction hypothesis, since $\delta_1 = \delta$. Suppose that number of existential quantifiers is $k+1$, and that the last quantifier is $\exists p$, that is $\delta \equiv \exists p. \exists \delta_1$. Then by the induction hypothesis there exists a Büchi automaton N such that $\Sigma \Vdash^{\mathsf{QEPLTL}} \exists \delta_1$ iff $\Sigma \in L_{\mathsf{Sem}}(N)$. Using the Projection proposition (8.2.1), the Büchi automaton $p(N)$ is such that $\Sigma \Vdash^{\mathsf{QEPLTL}} \exists p. \exists \delta_1$ iff $\Sigma \in L_{\mathsf{Sem}}(p(N))$.

In case (2), the proof also follows by induction on the number k of universal quantifiers. For $k = 0$, the argument is similar. Now suppose that the number of universal quantifiers is $k+1$, that is $\delta \equiv \forall p. \forall \delta_1$. Let N be a Büchi automaton such that $\Sigma \in L_{\mathsf{Sem}}(N)$ iff $\Sigma \Vdash^{\mathsf{QEPLTL}} \forall \delta_1$. We show that $\Sigma \in L_{\mathsf{Sem}}(\overline{p(\overline{N})})$ iff $\Sigma \Vdash^{\mathsf{QEPLTL}} \forall p. \forall \delta_1$, in three steps.

Step 1: First we show that $\Sigma \in L_{\mathsf{Sem}}(\overline{N})$ iff $\Sigma \not\Vdash^{\mathsf{EPLTL}} \forall\delta_1$.

$$\begin{aligned}
\Sigma \not\Vdash^{\mathsf{EPLTL}} \forall\delta_1 \text{ iff } & \Sigma \notin L_{\mathsf{Sem}}(N) \\
\text{iff } & \epsilon^*(\Sigma) \notin L_{\mathsf{SynCon}}(N), \text{ by Lemma 8.2.3} \\
\text{iff } & \epsilon^*(\Sigma) \in \overline{L_{\mathsf{SynCon}}(N)} \\
\text{iff } & \epsilon^*(\Sigma) \in \overline{L_{\mathsf{SynCon}}(N)} \cap Con^{\omega}(Lit) \\
\text{iff } & \epsilon^*(\Sigma) \in L_{\mathsf{SynCon}}(\overline{N}), \text{ by Lemma 8.2.1} \\
\text{iff } & \Sigma \in L_{\mathsf{Sem}}(\overline{N}), \text{ by Lemma 8.2.3 .}
\end{aligned}$$

Step 2: Then we show that $\Sigma \in L_{\mathsf{Sem}}(p(\overline{N}))$ iff $\Sigma \Vdash^{\mathsf{QEPLTL}} \exists p. \sim \forall\delta_1$. Note that $\Sigma \Vdash^{\mathsf{QEPLTL}} \exists p. \sim \forall\delta_1$ iff there exists $\Sigma' \sim_p$ equivalent to Σ such that $\Sigma' \Vdash \sim \forall\delta_1$ iff there exists $\Sigma' \sim_p$ equivalent to Σ such that $\Sigma' \in L_{\mathsf{Sem}}(\overline{N})$ which again by Proposition 8.2.1 allows us to obtain the desired result.

The final step is similar to Step 1. □

Corollary 8.3.1. The satisfiability problem for QEPLTL is decidable.

Proof. Let $\delta \in \mathsf{QEPLTL}$. Construct a Büchi automaton N such that $\Sigma \Vdash^{\mathsf{QEPLTL}} \delta$ iff $\Sigma \in L_{\mathsf{Sem}}(N)$. But then $L_{\mathsf{Sem}}(N) = \emptyset$ iff $L_{\mathsf{SynCon}}(N) = \emptyset$, and as such testing the emptiness of $L_{\mathsf{Sem}}(N)$ can be done by using the standard algorithm for Büchi automata. Note that if δ has no propositional quantifiers, one needs to consider the Büchi automaton N^+ instead of the Büchi automaton N . □

Note that as usual, the decidability of the SAT problem for QEPLTL implies that the validity problem is also decidable.

8.4 Concluding Remarks

We established the decidability of QEPLTL by using Büchi automata as representations of EPLTL formulae, and modifying the projection algorithm used to cope with propositional quantifiers. Even if the main point of this chapter was to prove the decidability of QEPLTL, we believe that the method proposed herein can be adapted to the temporalization of any decidable first-order theory that admits quantifier elimination, when considering fixed domains, as the techniques used herein did not require any special property of EPPL, or of the first-order theory of the ordered real closed fields. We note that there exists a restriction at this time as we are just considering formulae in prenex normal form; further research is needed to understand

when a propositional quantified generic temporalization has a prenex normal form.

Moreover, the use of Büchi automata as representations of formulae also seem to be quite general. In our case, we can now use these concepts as tools for more elaborate planning mechanisms, effectively enlarging the scope of LTL as a planning logic.

We can now extend the concluding remark on Chapter 5 by considering evolutions of probability spaces. Suppose that a certain protocol is adequately modeled by a EPLTL model, M_P; the fact that a protocol across multiple sessions is successful can usually be specified by a EPLTL formula like $\mathsf{GF}success(p_i, q_i)$, where *success* is a EPPL formula, p_i represents an instantiation of the protocol with certain values, and q_i the inner components of the protocol. Again suppose that the adversary has access to some of the inner components, by tampering with the channel contents; these actions can be modeled by (universally) quantifying the appropriate inner propositional symbols; however, if there exist some a posterior adjustable information, one should instead consider existential quantifiers prefixed by the universal quantifiers. We note that further alternations of quantifiers like for instance $\forall\exists\forall$ allow us to express even more complex properties, which are akin to strategies: $\forall\Pi_1\exists\Pi_2\forall\Pi_3\phi$ can be interpreted as "whatever may happen in Π_1, there exists an adjustment in Π_2 such that anything that might happen in Π_3 still satisfies ϕ."

Conclusions and Future Work

We have investigated logics that combine both probabilistic and temporal components with the primary aim of obtaining formal languages with useful capabilities for both verification (Chapters 4 and Chapter 7) and synthesis (Chapter 6 and Chapter 8) of automated systems.

Moreover, we have explored the theoretical possibilities of Exogenous Probabilistic Propositional Logic regarding its temporalizations, by considering EPLTL, its extension with propositional quantifiers QEPLTL, and the μ-calculus extension μ-EPPL, showing model-checking and satisfiability procedures for these logics. In each of Chapters in Part 2, we have provided a thorough analysis of the important questions that emerge naturally from the presentation. We summarize however the more significant points left unanswered:

- Regarding the procedure for quantifier elimination in QEPPL, we believe that the algorithm for quantifier elimination for the global fragment of EPPL is much easier than the one presented. Nevertheless obtaining tight complexity bounds for both the global fragment of EPPL and for the full algorithm is a relevant research topic, even if the complexity of the underlying algorithm for deciding the satisfiability of existential ordered real closed field formula is not known tightly.

- On Chapter 7, we address the verification problem of EPLTL formulae over Markov chains. In order to do so, we relied on results about the Skolem problem. In the concluding remarks we mentioned that if one was able to obtain a many-to-one reduction between the verification problem and the Skolem problem, one could establish the decidability of both problems. Now, consider the satisfiability problem for EPLTL when the set of models are Markov chains. This problem

seems connected to the existence of the many to one reduction mentioned in the concluding remarks relating the verification problem and the Skolem problem. A first step to address the satisfiability problem and the many to one reduction is to understand whether for any given index set I with representation (F, G) with $G = \{(p_1, r_1), \ldots, (p_n, r_n)\}$ whether we can construct an instance of the Skolem problem x, M, y whose zeros are precisely I. In a way, this problem is the dual of Skolem problem and to our knowledge it has not been researched at all.

- The Skolem problem has an continuous version named the Skolem-Pisot problem which shares many characteristics with the version presented herein. By choosing a suitable continuous time temporal logic, it would be relevant to understand whether the results shown here are adaptable. This would be an important contribution to the study of continuous time Markov chains.

- On Chapter 8, the results presented regarding quantification of EPLTL mainly require the existence of a quantifier elimination technique in EPPL; if one considers the decidable temporalization of a suitable logic L, where L admits quantifier elimination, is the decidability result preserved when we consider the quantified version of the temporalization?

- In order to better capture the adversary capabilities we would also like to investigate epistemic and multi-agent extensions of the proposed logics, possibly building upon propositional quantification results from [72].

Bibliography

[1] M. Agrawal, S. Akshay, B. Genest, and P. S. Thiagarajan. Approximate verification of the symbolic dynamics of Markov chains. *Journal of the ACM*, 62(1):2, 2014.

[2] R. B Ash and C. Doleans-Dade. *Probability and measure theory*. Academic Press, 2000.

[3] A. Aziz, V. Singhal, F. Balarin, and R. K. Brayton. It usually works: The temporal logic of stochastic systems. In *Procedings of the Seventh International Conference on Computer Aided Verification*, volume 939 of *Lecture Notes in Computer Science*, pages 155–165. Springer, 1995.

[4] M. Baaz, A. Ciabattoni, and R. Zach. Quantified propositional Gödel logics. In *Logic for Programming and Automated Reasoning*, volume 1955 of *Lecture Notes in Artificial Intelligence*, pages 240–256. Springer, 2000.

[5] C. Baier, E. M. Clarke, V. Hartonas-Garmhausen, M. Z. Kwiatkowska, and M. Ryan. Symbolic model checking for probabilistic processes. In *Proceedings of the Twenty-Fourth International Colloquium on Automata, Languages and Programming*, volume 1256 of *Lecture Notes in Computer Science*, pages 430–440. Springer, 1997.

[6] C. Baier, M. Größer, and N. Bertrand. Probabilistic ω-automata. *Journal of the ACM (JACM)*, 59(1):1, 2012.

[7] C. Baier and J.-P. Katoen. *Principles of Model Checking*. MIT Press, 2008.

[8] P. Baltazar. *Probabilization of Logic Systems*. PhD thesis, Instituto Superior Técnico, 2010.

[9] P. Baltazar and P. Mateus. Temporalization of probabilistic propositional logic. In *Logic Foundations of Computer Science 2009*, volume 5407 of *Lecture Notes in Computer Science*, pages 46–60. Springer, 2009.

[10] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in Real Algebraic Geometry*. Springer-Verlag, USA, 2006.

[11] D. Beauquier, A. Rabinovich, and A. Slissenko. A logic of probability with decidable model checking. *Journal of Logic and Computation*, 16(4):461–487, 2006.

[12] M. Ben-Or, D. Kozen, and J. H. Reif. The complexity of elementary algebra and geometry. *Journal of Computer and System Sciences*, 32(2):251–264, 1986.

[13] G. Bernot, J. P. Comet, A. Richard, and J. Guespin. Application of formal methods to biological regulatory networks: extending Thomas' asynchronous logical approach with temporal logic. *Journal of Theoretical Biology*, 220:339–347, 2004.

[14] J. Berstel and M. Mignotte. Deux propriétés décidables des suites récurrentes linéaires. *Bulletin de la Société Mathématique de France*, 104:175–184, 1976.

[15] A. Bianco and L. De Alfaro. Model checking of probabilistic and nondeterministic systems. In *Foundations of Software Technology and Theoretical Computer Science*, volume 1026 of *Lecture Notes in Computer Science*, pages 499–513. Springer, 1995.

[16] G. V. Bochmann. Hardware specification with temporal logic: an example. *IEEE Transactions on Computers*, 100(3):223–231, 1982.

[17] M. Bojańczyk. The common fragment of ACTL and LTL. In *Foundations of Software Science and Computational Structures*, volume 4962 of *Lecture Notes in Computer Science*, pages 172–185. Springer, 2008.

[18] B. Bonet. Learning depth-first search: a unified approach to heuristic search in deterministic and non-deterministic settings, and its application to mdps. In *Proceedings of the Sixteenth International Conference on Automated Planning & Scheduling*, pages 3–23, 2006.

[19] R. Brafman and J. Hoffmann. Conformant planning via heuristic forward search: A new approach. In *Proceedings of the Fourteenth International Conference in Automated Planning & Scheduling*, pages 355–364, 2004.

[20] T. Brázdil, V. Forejt, J. Kretínský, and A. Kucera. The satisfiability problem for probabilistic CTL. In *Proceedings of the Twenty-third Annual IEEE Symposium on Logic in Computer Science*, pages 391–402, USA, 2008. IEEE Computer Society.

[21] J. Büchi. Weak second-order arithmetic and finite automata. *Mathematical Logic Quarterly*, 6(1-6):66–92, 1960.

[22] R. A. Bull. On modal logic with propositional quantifiers. *Journal of Symbolic Logic*, 34(2):257–263, 1969.

[23] C. S. Calude. *Information and randomness: an algorithmic perspective*. Springer Science & Business Media, 2013.

[24] R. Chadha, P. Mateus, A. Sernadas, and C. Sernadas. Extending classical logic for reasoning about quantum systems. In *Handbook of Quantum Logic and Quantum Structures: Quantum Logic*, pages 325–372. Elsevier, 2009.

[25] V. Chávtal. *Linear programming*. WH Freeman, 1983.

[26] F. Ciesinski and C. Baier. LiQuor: A tool for qualitative and quantitative linear time analysis of reactive systems. In *Proceedings of the Third International Conference on the Quantitative Evaluation of Systems*, pages 131–132. IEEE CS Press, 2006.

[27] F. Ciesinski and M. Größer. On probabilistic computation tree logic. In *Validation of Stochastic Systems*, volume 2925 of *Lecture Notes in Computer Science*, pages 147–188. Springer, 2004.

[28] E. M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 1999.

[29] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press and McGraw-Hill Book Company, Second edition, 2001.

[30] C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *Journal of ACM*, 42:857–907, 1995.

[31] M.J. Cresswell. Temporal reference in linear tense logic. *Journal of philosophical logic*, 39(2):173–200, 2010.

[32] B. A. Davey and H. A. Priestley. *Introduction to lattices and order*. Cambridge University Press, 2002.

[33] E. A. Emerson. *Temporal and Modal Logic*, pages 995–1072. Elsevier and MIT, 1990.

[34] E. A. Emerson and E. M. Clarke. Using branching time temporal logic to synthesize synchronization skeletons. *Science Computer Programming*, 2(3):241–266, 1982.

[35] E. A. Emerson and J. Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. In *Proceedings of the Fourteenth Annual ACM symposium on Theory of Computing*, pages 169–180. ACM, 1985.

[36] E. A. Emerson and J. Y. Halpern. Sometimes and not never revisited: on branching versus linear time temporal logic. *Journal of the ACM (JACM)*, 33(1):151–178, 1986.

[37] R. Fagin and J. Y. Halpern. Reasoning about knowledge and probability. *Journal of the ACM*, 41(2):340–367, 1994.

[38] R. Fagin, J. Y. Halpern, and N. Megiddo. A logic for reasoning about probabilities. *Information and computation*, 87(1):78–128, 1990.

[39] A. Fehnker and P. Gao. Formal verification and simulation for performance analysis for probabilistic broadcast protocols. In *Proceedings of the Fifth International Conference on Ad-Hoc, Mobile, and Wireless Networks*, volume 4104 of *Lecture Notes in Computer Science*, pages 128–141. Springer, 2006.

[40] M. Finger and D. Gabbay. Combining temporal logic systems. *Notre Dame Journal of Formal Logic*, 37(2):204–232, 1996.

[41] M. Franceschet and A. Montanari. Towards an automata-theoretic counterpart of combined temporal logics. In *Proceedings of the Second International Workshop on Verification and Computational Logic*, pages 55–74, 2001.

[42] M. Franceschet and A. Montanari. Temporalized logics and automata for time granularity. *Theory and Practice of Logic Programming*, 4(5-6):621–658, 2004.

[43] M. Frappier, B. Fraikin, R. Chossart, R. Chane-Yack-Fa, and M. Ouenzar. Comparison of model checking tools for information systems. In *Proceedings of the Twelfth International Conference on Formal Engineering Methods*, volume 6447 of *Lecture Notes in Computer Science*, pages 581–596. Springer, 2010.

[44] T. French. Decidability of quantified propositional branching time logics. In *Proceedings of the Fourteenth Australian Joint Conference on Artificial Intelligence*, volume 2256 of *Lecture Notes in Computer Science*, pages 165–176. Springer, 2001.

[45] T. French and M. Reynolds. A sound and complete proof system for QPTL. In *Advances in Modal Logic*, pages 127–148. King's College Publications, 2002.

[46] D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the temporal analysis of fairness. In *Proceedings of the Seventh ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 163–173. ACM, 1980.

[47] S. Gay, R. Nagarajan, and N. Papanikolaou. QMC: A model checker for quantum systems. In *Proceedings of the Twentieth International Conference in Computer Aided Verification*, volume 5123 of *Lecture Notes in Computer Science*, pages 543–547. Springer, 2008.

[48] G. Georgakopoulos, D. Kavvadias, and C. Papadimitriou. Probabilistic satisfiability. *Journal of complexity*, 4(1):1–11, 1988.

[49] R. Gerth, D. Peled, M. Y. Vardi, and P. Wolper. Simple on-the-fly automatic verification of linear temporal logic. In *Proceedings of the Fifteenth International Symposium on Protocol Specification, Testing and Verification*, pages 3–18, 1996.

[50] J. Y. Halpern. An analysis of first-order logics of probability. *Artificial Intelligence*, 46:311–350, 1990.

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

[52] I. Hodkinson, F. Wolter, and M. Zakharyaschev. Decidable fragments of first-order temporal logics. *Annals of Pure and Applied logic*, 106(1):85–134, 2000.

[53] P. Øhrstrøm and P. Hasle. *Temporal Logic: From Ancient Ideas to Artificial Intelligence*. Kluwer, 1995.

[54] S. Jha, E. M. Clarke, C. J. Langmead, A. Legay, A. Platzer, and P. Zuliani. A bayesian approach to model checking biological systems. In *Computational Methods in Systems Biology*, pages 218–234. Springer, 2009.

[55] S. Jiang and R. Kumar. Supervisory control of discrete event systems with CTL* temporal logic specifications. *SIAM Journal on Control and Optimization*, 44(6):2079–2103, 2006.

[56] H. Kamp. *Tense logic and the theory of linear order*. PhD thesis, UCLA, USA, 1968.

[57] J.-P. Katoen, E. M. Hahn, H. Hermanns, D. Jansen, and I. Zapreev. The ins and outs of the probabilistic model checker MRMC. In *Sixth International Conference on the Quantitative Evaluation of Systems*, pages 167–176. IEEE CS Press, 2009.

[58] M. Kattenbelt and M. Huth. Verification and refutation of probabilistic specifications via games. In *Proceedings of the Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 4 of *Leibniz International Proceedings in Informatics*, pages 251–262. Schloss Dagstuhl, 2009.

[59] H. Kautz and B. Selman. Planning as satisfiability. In *Proceedings of the Tenth European Conference on Artificial Intelligence*, pages 359–363, 1992.

[60] J. G. Kemeny, J. L. Snell, and A. W. Knapp. *Denumerable Markov Chains*. Van Nostrand, New Jersey, 1966.

[61] A. Klenke. *Probability theory: a comprehensive course*. Springer Science & Business Media, 2013.

[62] F. Königsbuch, G. Infantes, and U. Kuter. RFF: a robust, ff-based MDP planning algorithm for generating policies with low probability of failure. *Sixth International Planning Competition at the International Conference on Automated Planning & Scheduling*, 8, 2008.

[63] V. Korthikanti, M. Viswanathan, G. Agha, and Y. Kwon. Reasoning about MDPs as transformers of probability distributions. In *Seventh*

International Conference on the Quantitative Evaluation of Systems, pages 199–208. IEEE CS Press, 2010.

[64] R. Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4):255–299, 1990.

[65] D. Kozen. Results on the propositional mu-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

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

[67] D. Kozen and R. Parikh. A decision procedure for the propositional mu-calculus. In *Proceedings of the Carnegie Mellon Workshop on Logic of Programs*, pages 313–325. Springer-Verlag, 1984.

[68] M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. In *Proceedings of 23rd Conference in Computer Aided Verification*, volume 6806 of *Lecture Notes in Computer Science*, pages 585–591. Springer, 2011.

[69] Y. Kwon and G. Agha. Linear inequality LTL (iLTL): A model checker for discrete time Markov chains. In *Proceedings of the 6th International Conference on Formal Engineering Methods*, volume 3308 of *Lecture Notes in Computer Science*, pages 194–208. Springer, 2004.

[70] B. Lacerda. *Supervision of Discrete Event Systems Based on Temporal Logic Specifications*. PhD thesis, Instituto Superior Técnico, 2013.

[71] L. Landweber. Decision problems for ω-automata. *Mathematical Systems Theory*, 3:376–384, 1969.

[72] A. Lopes, F. Laroussinie, and N. Markey. Quantified CTL: expressiveness and model checking - (extended abstract). In *Proceedings of Twenty-Third International Conference on Concurrency Theory*, volume 7454 of *Lecture Notes in Computer Science*, pages 177–192. Springer, 2012.

[73] A. Macintyre and A. J. Wilkie. On the decidability of the real exponential field. In *Kreiseliana. About and Around Georg Kreisel*, pages 441–467. A. K. Peters, 1996.

[74] K. Mahler and J. W. S. Cassels. On the Taylor coefficients of rational functions. *Mathematical Proceedings of the Cambridge Philosophical Society*, 52:39–48, 1956.

[75] P. Mateus, A. Pacheco, J. Pinto, A. Sernadas, and C. Sernadas. Probabilistic situation calculus. *Annals of Mathematics and Artificial Intelligence*, 32(1-4):393–431, 2001.

[76] P. Mateus and A. Sernadas. Reasoning about quantum systems. In *Proceedings of the Ninth Conference on Logics in Artificial Intelligence*, volume 3229 of *Lecture Notes in Artificial Intelligence*, pages 239–251. Springer-Verlag, 2004.

[77] P. Mateus and A. Sernadas. Weakly complete axiomatization of exogenous quantum propositional logic. *Information and Computation*, 204:771–794, 2006.

[78] P. Mateus, A. Sernadas, and C. Sernadas. Exogenous semantics approach to enriching logics. In *Essays on the Foundations of Mathematics and Logic, volume 1 of Advanced Studies in Mathematics and Logic*, pages 165–194. Polimetrica, 2005.

[79] F. Moller and A. Rabinovich. Counting on CTL*: on the expressive power of monadic path logic. *Information and Computation*, 184(1):147–159, 2003.

[80] A. Mordido and C. Caleiro. An equation-based classical logic. In *Proceedings of the Twenty-Second International Workshop in Logic, Language, Information, and Computation*, volume 9160 of *Lecture Notes in Computer Science*, pages 38–52. Springer, 2015.

[81] B. C. Moszkowski and Z. Manna. Reasoning in interval temporal logic. In *Proceedings of the Workshop in Logics of Programs, Carnegie Mellon University*, pages 371–382, 1983.

[82] G. Norman and V. Shmatikov. Analysis of probabilistic contract signing. *Journal of Computer Security*, 14(6):561–589, 2006.

[83] Z. Ognjanović. Discrete linear-time probabilistic logics: Completeness, decidability and complexity. *Journal of Logic and Computation*, 16(2):257–285, 2006.

[84] J. Ouaknine and J. Worrell. Decision problems for linear recurrence sequences. In *Proceedings of the Sixth International Workshop on Reachability Problems*, volume 7550 of *Lecture Notes in Computer Science*, pages 21–28. Springer, 2012.

[85] J. Ouaknine and J. Worrell. Positivity problems for low-order linear recurrence sequences. *Computing Research Repository*, abs/1307.2779, 2013.

[86] A. Paz. *Introduction to Probabilistic Automata*. Academic Press, Inc., USA, 1971.

[87] A. Platzer. *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Springer, Heidelberg, 2010.

[88] A. Platzer and J.-D. Quesel. KeYmaera: A hybrid theorem prover for hybrid systems. In *Proceedings of the Fourth International Joint Conference on Automated Reasoning*, volume 5195 of *Lecture Notes in Computer Science*, pages 171–178. Springer, 2008.

[89] A. Pnueli. The temporal logic of programs. In *Eighteenth Annual Symposium on Foundations of Computer Science*, pages 46–57, 1977.

[90] M. Reynolds. An axiomatization of full computation tree logic. *The Journal of Symbolic Logic*, 66(03):1011–1057, 2001.

[91] J. Rintanen. Planning as satisfiability: Heuristics. *Artificial Intelligence*, 193:45–86, 2012.

[92] Shmuel Safra. On the complexity of ω-automata. In *29th Annual Symposium on Foundations of Computer Science*, pages 319–327. IEEE, 1988.

[93] M. Sipser. *Introduction to the theory of computation*. PWS Pub., Boston, 2nd edition, 2006.

[94] A. P. Sistla. *Theoretical issues in the design and verification of distributed systems*. PhD thesis, Harvard University, USA, 1983.

[95] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. In *Proceedings of the Fourteenth Annual ACM symposium on Theory of Computing*, pages 159–168, USA, 1982. ACM.

[96] A. P. Sistla, M. Y. Vardi, and P. Wolper. The complementation problem for Büchi automata with applications to temporal logic (extended abstract). In *Proceedings of the Twelfth Colloquium, Automata, Languages and Programming,*, volume 194 of *Lecture Notes in Computer Science*, pages 465–474. Springer, 1985.

[97] T. Skolem. Ein Verfahren zur Behandlung gewisser exponentialer Gleichungen und diophantischer Gleichungen. 8th Skand. Mat. Kongr., Stockhohn, 163-188, 1934.

[98] L. J. Stockmeyer and A. R. Meyer. Word problems requiring exponential time: Preliminary report. In *Proceedings of the Fifth Annual ACM Symposium on Theory of Computing*, pages 1–9. ACM, 1973.

[99] Alfred Tarski. A decision method for elementary algebra and geometry. 1951.

[100] Wolfgang Thomas. Automata on Infinite Objects. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science (Vol. B)*, pages 133–191. MIT Press, 1990.

[101] M. Y. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *Proceedings of the Twenty-Sixth Annual Symposium on Foundations of Computer Science*, pages 327–338, 1985.

[102] M. Y. Vardi. Probabilistic linear-time model checking: an overview of the automata-theoretic approach. In *Proceedings of the Fifth International Workshop on Formal Methods for Real-Time and Probabilistic Systems*, volume 1601 of *Lecture Notes in Computer Science*, pages 265–276. Springer, 1999.

[103] M. Y. Vardi. Branching vs. linear time: Final showdown. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 1–22. Springer, 2001.

[104] M. Y. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification. In *Proceedings of the Symposium on Logic in Computer Science*, pages 332–344. IEEE Computer Society, 1986.

[105] I. Walukiewicz. Completeness of Kozen's axiomatisation of the propositional mu-calculus. In *Proceedings of the Tenth Annual IEEE Symposium on Logic in Computer Science*, pages 14–24, Washington, DC, 1995. IEEE Computer Society.

[106] P. Wolper. Temporal logic can be more expressive. *Information and Control*, 56(1 - 2):72 – 99, 1983.

[107] B. Yordanov, J. Tumova, I. Cerna, J. Barnat, and C. Belta. Temporal logic control of discrete-time piecewise affine systems. *IEEE Transactions on Automatic Control*, 57(6):1491–1504, 2012.