Model Checking Discounted Temporal Properties*

Luca de Alfaro[1], Marco Faella[1,3], Thomas A. Henzinger[2], Rupak Majumdar[4], and Mariëlle Stoelinga[1]

[1] CE, Universitity of California, Santa Cruz, USA
[2] EECS, University of California, Berkeley, USA
[3] Università degli Studi di Salerno, Italy
[4] CS, University of California, Los Angeles, USA

Abstract. Temporal logic is two-valued: a property is either true or false. When applied to the analysis of stochastic systems, or systems with imprecise formal models, temporal logic is therefore fragile: even small changes in the model can lead to opposite truth values for a specification. We present a generalization of the branching-time logic CTL which achieves robustness with respect to model perturbations by giving a quantitative interpretation to predicates and logical operators, and by discounting the importance of events according to how late they occur. In every state, the value of a formula is a real number in the interval [0,1], where 1 corresponds to truth and 0 to falsehood. The boolean operators and and or are replaced by min and max, the path quantifiers $\exists$ and $\forall$ determine sup and inf over all paths from a given state, and the temporal operators $\Diamond$ and $\Box$ specify sup and inf over a given path; a new operator averages all values along a path. Furthermore, all path operators are discounted by a parameter that can be chosen to give more weight to states that are closer to the beginning of the path.
We interpret the resulting logic DCTL over transition systems, Markov chains, and Markov decision processes. We present two semantics for DCTL: a *path* semantics, inspired by the standard interpretation of state and path formulas in CTL, and a *fixpoint* semantics, inspired by the μ-calculus evaluation of CTL formulas. We show that, while these semantics coincide for CTL, they differ for DCTL, and we provide model-checking algorithms for both semantics.

1 Introduction

Boolean state-transition models are useful for the representation and verification of computational systems, such as hardware and software systems. A boolean state-transition model is a labeled directed graph, whose vertices represent system states, whose edges represent state changes, and whose labels represent

* This research was supported in part by the AFOSR MURI grant F49620-00-1-0327, the ONR grant N00014-02-1-0671, and the NSF grants CCR-0132780, CCR-9988172, CCR-0225610, and CCR-0234690.

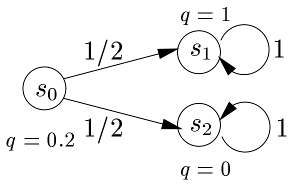
boolean observations about the system, such as the truth values of state predicates. Behavioral properties of boolean state-transition systems can be specified in temporal logic [15, 4] and verified using model-checking algorithms [4].

For representing systems that are not purely computational but partly physical, such as hardware and software that interacts with a physical environment, boolean state-transition models are often inadequate. Many quantitative extensions of state-transition models have been proposed for this purpose, such as models that embed state changes into the real time line, and models that assign probabilities to state changes. These models typically contain real numbers, e.g., for representing time or probabilities. Yet previous research has focused mostly on purely boolean frameworks for the specification and verification of quantitative state-transition models, where observations are truth values of state predicates, and behavioral properties are based on such boolean observations [10, 3, 1, 14]. These boolean specification frameworks are *fragile* with respect to imprecisions in the model: even arbitrarily small changes in a quantitative model can cause different truth values for the specification.

We submit that a proper framework for the specification and verification of quantitative state-transition models should itself be quantitative. To start with, we consider observations that do not have boolean truth values, but real values [13]. Using these quantitative observations, we build a temporal logic for specifying quantitative temporal properties. A CTL-like temporal logic has three kinds of operators. The first kind are boolean operators such as "and" and "or" for locally combining the truth values of boolean observations. These are replaced by "min" and "max" operators for combining the real values of quantitative observations. In addition, a "weighted average" ($\oplus_c$) operator is useful to generate new quantitative observations. The second kind of constructs are modal operators such as "always" ($\Box$) and "eventually" ($\Diamond$) for temporally combining the truth values of all boolean observations along an infinite path. These are replaced by "inf" ("lim min") and "sup" ("lim max") operators over infinite sequences of real values. We introduce a "lim avg" ($\triangle$) operator that captures the long-run average value of a quantitative observation. For nondeterministic models, where there is a choice of future behaviors, there is a third kind of constructs: the path quantifiers "for-all-possible-futures" ($\forall$) and "for-some-possible-future" ($\exists$) turn path properties into state properties by quantifying over the paths from a given state. These are replaced by "inf-over-all-possible-futures" and "sup-over-all-possible-futures." Once boolean specifications are replaced by quantitative specifications, it becomes possible to discount the future, that is, to give more weight to the near future than to the far away future. This principle is well-understood in economics and in the theory of optimal control [2], but is equally natural in studying quantitative temporal properties of systems [7]. We call the resulting logic DCTL ("Discounted CTL"). While quantitative versions of dynamic logics [13], μ-calculi [11, 16, 17, 7], and Hennessy-Milner logics [8] exist, DCTL is the first temporal logic in which the non-local temporal operators $\Diamond$ and $\Box$, along with the new temporal operator $\triangle$ and the path quantifiers $\forall$ and $\exists$, are given a quantitative interpretation.

We propose two semantics for DCTL: a *path* semantics and a *fixpoint* semantics. In the undiscounted path semantics, the $\Diamond$ (resp. $\Box$) operator computes the sup (resp. inf) over a path, and the $\triangle$ operator computes the long-run average. The discounted versions $\Diamond_\alpha$, $\Box_\alpha$, and $\triangle_\alpha$ of these operators weigh the value of a state that occurs k steps in the future by a factor α^k, where $\alpha \leq 1$ is the discount factor. The $\forall$ and $\exists$ operators then combine these values over the paths: in transition systems, $\forall$ and $\exists$ associate with each state the inf and sup of the values for the paths that leave the state; in probabilistic systems, $\forall$ and $\exists$ associate with each state the least and greatest expectation of the value for those paths (for Markov chains, there is a single expected value at each state, but for Markov decision processes, the least and greatest expected value are generally different). Thus, the path semantics of DCTL is obtained by lifting to a quantitative setting the classical interpretation of path and state formulas in CTL.

The *fixpoint* semantics is obtained by lifting to a quantitative setting the connection between CTL and the μ-calculus [4]. In a transition system, given a set r of states, denote by $\exists\mathrm{Pre}(r)$ the set of all states that have a one-step transition to r. Then, the semantics of $\exists\Diamond r$ for a set r of states can be defined as the least fixpoint of the equation $x = r \cup \exists\mathrm{Pre}(x)$, denoted $\mu x.(r \cup \exists\mathrm{Pre}(x))$. We lift this definition to a quantitative setting by interpreting $\cup$ as pointwise maximum, and $\exists\mathrm{Pre}(x)$ as the maximal expected value of x achievable in one step [7]. The discounted semantics $\exists\Diamond_\alpha r$ is obtained by multiplying the next-step expectation with α, i.e., $\mu x.(r \cup \alpha \cdot \exists\mathrm{Pre}(x))$.



The path and fixpoint semantics coincide on transition systems, but differ on Markov chains (and consequently on Markov decision processes). This is illustrated by the Markov chain depicted at right. Consider the DCTL formula ϕ: $\exists\Diamond_\alpha q$, for $\alpha = 0.8$. According to the path semantics, there are two paths from s_0, each followed with probability $1/2$: the first path has the discounted sup equal to 0.8, and the second has the discounted sup equal to 0.2; hence, ϕ has the value $(0.8 + 0.2)/2 = 0.5$ at s_0. According to the fixpoint semantics, $q \cup 0.8 \cdot \exists\mathrm{Pre}(q)$ has the value $\max\{0.2, 0.8 \cdot (1+0)/2)\} = 0.4$ at s_0, and this is also the value of ϕ at s_0. This example highlights the different perspective taken by the two semantics. The path semantics of $\exists\Diamond q$ is an "observational" semantics: if q represents, for instance, the level of water in a vessel (0 is empty, 1 is full), then $\exists\Diamond q$ is the expected value of the maximum level that occurs along a system behavior. Such a semantics is well-suited for system specification. The fixpoint semantics of $\exists\Diamond q$ is a "controlling" semantics: if q represents the retirement bonus that we receive if we decide to retire, then $\exists\Diamond q$ is the maximal expected bonus we will receive (discounting accounts for inflation). The difference is that in the fixpoint semantics we must decide when to stop: the choice of retiring, or working for one more day, corresponds to the choice between the two sides q and $\exists\mathrm{Pre}(x)$ of the $\cup$ operator (interpreted as pointwise maximum) in the fixpoint. Hence the fixpoint semantics is better suited for system control: if the goal is to reach a state with a high value of q, we must not only reach such a state, but also stop, once satisfied by a sufficiently high value of q, and move on to some subsequent control goal. In the path semantics, on the other hand, we have no control over stopping: we can only observe the value of q over infinite runs, and compute the expected value of the sup it reaches.

In DCTL, discounting serves two purposes. First, it leads to a notion of "quality" with which a specification is satisfied. For example, if we wish to reach a state with a high value of q, then the undiscounted formula $\exists\Diamond q$ is valid regardless of when the high q value is reached, whereas $\exists\Diamond_\alpha q$, for $\alpha < 1$, has a higher value if the high q value is reached earlier. Likewise, if q represents the "level of functionality" of a system, then the specification $\forall\Box_\alpha q$ will have a value that is higher the longer the system functions well, even if the system will eventually always break. Second, discounting is instrumental in achieving robustness with respect to system perturbations. Indeed, we will show that for discount factors smaller than 1, the value of DCTL formulas in both semantics is a continuous function of the values of the numerical quantities (observations, transition probabilities) of the model.

We present algorithms for model checking both semantics of DCTL over transition systems, Markov chains, and Markov decision processes (MDPs). Note that, for discount factors less than 1, DCTL is a quantitative logic even when interpreted over purely boolean state-transition systems. Over transition systems, the algorithms for $\Box_\alpha$ and $\Diamond_\alpha$ are based on iterating quantitative fixpoint expressions; the main result in this regard is that the iteration always terminates within a number of steps bounded by the diameter of the system. The algorithm for $\triangle_\alpha$ (discounted long-run average along a path) is more involved, but still polynomial: it builds on both Karp's algorithm for computing minimum mean-weight cycles and a discounted version of Bellman-Ford for computing shortest paths. For Markov chains and MDPs, we can model check the fixpoint semantics of DCTL by relying on a mix of results from optimal control [2] and the quantitative μ-calculus [7]. On the other hand, model checking the path semantics of DCTL over Markov chains and MDPs requires novel algorithms. In all cases, the model checking problem for DCTL can be solved in time polynomial in the size of the system. For transition systems, the time required is also polynomial in the size of the DCTL formula. For Markov chains and MDPs, the time required is polynomial in the size of the DCTL formula for the fixpoint semantics, and exponential for the path semantics. The latter exponential complexity is caused by the fact that, in the path semantics, the bit-wise encodings of the valuations of Markov chains and MDPs grows exponentially with respect to the number of nestings of temporal operators (in practice, of course, one would be unlikely to implement arbitrary-precision arithmetic).

2 Discounted CTL

Syntax. Let Σ be a set of propositions and let A be a set of parameters. The DCTL formulas over (Σ, A) are generated by the grammar

$$\begin{aligned} \phi &::= r \mid \mathrm{T} \mid \mathrm{F} \mid \phi \vee \phi \mid \phi \wedge \phi \mid \neg\phi \mid \phi \oplus_c \phi \mid \exists\psi \mid \forall\psi \\ \psi &::= \Diamond_c \phi \mid \Box_c \phi \mid \triangle_c \phi \end{aligned}$$

where $r \in \Sigma$ is a proposition and $c \in A$ is a parameter. The formulas generated by ϕ are *state formulas*; the formulas generated by ψ are *path formulas*. The DCTL formulas are the state formulas.

2.1 Semantics for Labeled Transition Systems

We define two semantics for DCTL: the path semantics, and the fixpoint semantics. In the path semantics, the path operators $\Diamond$ and $\Box$ determine the discounted sup and inf values over a path, and the $\exists$ and $\forall$ operators determine the minimum and maximum values of the path formula over all paths from a given state. The fixpoint semantics is defined by lifting to a quantitative setting the usual connection between CTL and μ-calculus.

Discount factors. Let A be a set of parameters. A *parameter interpretation* of A is a function $\langle\cdot\rangle\colon A \to [0,1]$ that assigns to each parameter a real number between 0 and 1, called a *discount factor*. The interpretation $\langle\cdot\rangle$ is *contractive* if $\langle c\rangle < 1$ for all $c \in A$; it is *undiscounted* if $\langle c\rangle = 1$ for all $c \in A$. We write $\mathcal{I}_A$ for the set of parameter interpretations of A. We denote by $|q|_b$ the length of the binary encoding of a number $q \in \mathbb{Q}$, and we denote by $|\langle\cdot\rangle|_b = \sum_{c\in A} |\langle c\rangle|_b$ the size of the interpretation $\langle\cdot\rangle$ of A.

Valuations. Let S be a set of states. A *valuation* on S is a function $v\colon S \to [0,1]$ that assigns to each state a real between 0 and 1. The valuation v is *boolean* if $v(s) \in \{0,1\}$ for all $s \in S$. We write $\mathcal{V}_S$ for the set of valuations on S. We write $\mathbf{0}$ for the valuation that maps all states to 0, and $\mathbf{1}$ for the valuation that maps all states to 1. For two real numbers u_1, u_2 and a discount factor $\alpha \in [0,1]$, we write $u_1 \sqcup u_2$ for $\max\{u_1, u_2\}$, $u_1 \sqcap u_2$ for $\min\{u_1, u_2\}$, and $u_1 +_\alpha u_2$ for $(1-\alpha)\cdot u_1 + \alpha\cdot u_2$. We lift operations on reals to operations on valuations in a pointwise fashion; for example, for two valuations $v_1, v_2 \in \mathcal{V}_S$, by $v_1 \sqcup v_2$ we denote the valuation that maps each state $s \in S$ to $v_1(s) \sqcup v_2(s)$.

Labeled transition systems. A *labeled transition system* (LTS) $\mathcal{S} = (S, \delta, \Sigma, [\cdot])$ consists of a set S of states, a transition relation $\delta\colon S \to 2^S \setminus \emptyset$ that assigns to each state a finite nonempty set of successor states, a set Σ of propositions, and a function $[\cdot]\colon \Sigma \to \mathcal{V}_S$ that assigns to each proposition a valuation. We denote by $|\delta|$ the value $\sum_{s\in S} |\delta(s)|$. The LTS $\mathcal{S}$ is *boolean* if for all propositions $r \in \Sigma$, the valuation $[r]$ is boolean. A *path* of $\mathcal{S}$ is an infinite sequence $s_0 s_1 s_2 \ldots$ of states such that $s_{i+1} \in \delta(s_i)$ for all $i \geq 0$. Given a state $s \in S$, we write Traj_s for the set of paths that start in s.

The path semantics. The DCTL formulas over (Σ, A) are evaluated w.r.t. an LTS $\mathcal{S} = (S, \delta, \Sigma, [\cdot])$ whose propositions are Σ, and w.r.t. a parameter interpretation $\langle\cdot\rangle \in \mathcal{I}_A$. Every state formula ϕ defines a valuation $[\![\phi]\!]^{\mathrm{p}} \in \mathcal{V}_S$:

$$
\begin{array}{lcl}
[\![r]\!]^{\mathrm{p}} &=& [r] \\
[\![\mathrm{T}]\!]^{\mathrm{p}} &=& \mathbf{1} \\
[\![\mathrm{F}]\!]^{\mathrm{p}} &=& \mathbf{0} \\
[\![\neg\phi]\!]^{\mathrm{p}} &=& \mathbf{1} - [\![\phi]\!]^{\mathrm{p}}
\end{array}
\qquad
\begin{array}{lcl}
[\![\phi_1 \vee \phi_2]\!]^{\mathrm{p}} &=& [\![\phi_1]\!]^{\mathrm{p}} \sqcup [\![\phi_2]\!]^{\mathrm{p}} \\
[\![\phi_1 \wedge \phi_2]\!]^{\mathrm{p}} &=& [\![\phi_1]\!]^{\mathrm{p}} \sqcap [\![\phi_2]\!]^{\mathrm{p}} \\
[\![\phi_1 \oplus_c \phi_2]\!]^{\mathrm{p}} &=& [\![\phi_1]\!]^{\mathrm{p}} +_{\langle c\rangle} [\![\phi_2]\!]^{\mathrm{p}} \\
[\![\exists\psi]\!]^{\mathrm{p}}(s) &=& \sup\{[\![\psi]\!]^{\mathrm{p}}(\rho) \mid \rho \in \mathit{Traj}_s\} \\
[\![\forall\psi]\!]^{\mathrm{p}}(s) &=& \inf\{[\![\psi]\!]^{\mathrm{p}}(\rho) \mid \rho \in \mathit{Traj}_s\}
\end{array}
$$

A path formula ψ assigns a real $[\![\psi]\!]^{\mathrm{p}}(\rho) \in [0,1]$ to each path ρ of $\mathcal{S}$:

$$
\begin{aligned}
[\![\Diamond_c \phi]\!]^{\mathrm{p}}(s_0 s_1 \ldots) &= \sup\{\langle c\rangle^i \cdot [\![\phi]\!]^{\mathrm{p}}(s_i) \mid i \geq 0\} \\
[\![\Box_c \phi]\!]^{\mathrm{p}}(s_0 s_1 \ldots) &= \inf\{1 - \langle c\rangle^i \cdot (1 - [\![\phi]\!]^{\mathrm{p}}(s_i)) \mid i \geq 0\} \\
[\![\triangle_c \phi]\!]^{\mathrm{p}}(s_0 s_1 \ldots) &= \begin{cases} (1-\langle c\rangle) \cdot \sum\{\langle c\rangle^i \cdot [\![\phi]\!]^{\mathrm{p}}(s_i) \mid i \geq 0\} & \text{if } \langle c\rangle < 1 \\ \lim_{i \geq 0}(\frac{1}{i+1} \cdot \sum_{0 \leq j \leq i} [\![\phi]\!]^{\mathrm{p}}(s_j)) & \text{if } \langle c\rangle = 1 \end{cases}
\end{aligned}
$$

Notice that the limit of the first clause for $\triangle_c$ when $\langle c\rangle \to 1$ gives the second clause. If the LTS $\mathcal{S}$ is boolean and the parameter interpretation $\langle\cdot\rangle$ is undiscounted, then 1 can be interpreted as truth, 0 as falsehood, and DCTL without the operator $\triangle$ coincides with CTL.

The fixpoint semantics. In this semantics, the DCTL formulas are evaluated with respect to an LTS $\mathcal{S}$ and a *contractive* parameter interpretation $\langle\cdot\rangle \in \mathcal{I}_A$. Given a valuation $x \in \mathcal{V}_S$, we denote by $\exists\mathrm{Pre}(x) \in \mathcal{V}_S$ the valuation defined by $\exists\mathrm{Pre}(x)(s) = \max\{x(t) \mid t \in \delta(s)\}$, and we denote by $\forall\mathrm{Pre}(x) \in \mathcal{V}_S$ the valuation defined by $\forall\mathrm{Pre}(x)(s) = \min\{x(t) \mid t \in \delta(s)\}$. The fixpoint semantics $[\![\cdot]\!]^{\mathrm{f}}$ for the propositions, the boolean operators, and $\oplus_c$ is similar to the path semantics, only that $[\![\cdot]\!]^{\mathrm{p}}$ is replaced by $[\![\cdot]\!]^{\mathrm{f}}$. The other operators are defined as follows:

$$
\begin{aligned}
[\![\exists\Diamond_c \phi]\!]^{\mathrm{f}} &= \mu x.([\![\phi]\!]^{\mathrm{f}} \sqcup (\mathbf{0} +_{\langle c\rangle} \exists\mathrm{Pre}(x))) \\
[\![\forall\Diamond_c \phi]\!]^{\mathrm{f}} &= \mu x.([\![\phi]\!]^{\mathrm{f}} \sqcup (\mathbf{0} +_{\langle c\rangle} \forall\mathrm{Pre}(x))) \\
[\![\exists\Box_c \phi]\!]^{\mathrm{f}} &= \mu x.([\![\phi]\!]^{\mathrm{f}} \sqcap (\mathbf{1} +_{\langle c\rangle} \exists\mathrm{Pre}(x))) \\
[\![\forall\Box_c \phi]\!]^{\mathrm{f}} &= \mu x.([\![\phi]\!]^{\mathrm{f}} \sqcap (\mathbf{1} +_{\langle c\rangle} \forall\mathrm{Pre}(x))) \\
[\![\exists\triangle_c \phi]\!]^{\mathrm{f}} &= \mu x.([\![\phi]\!]^{\mathrm{f}} +_{\langle c\rangle} \exists\mathrm{Pre}(x)) \\
[\![\forall\triangle_c \phi]\!]^{\mathrm{f}} &= \mu x.([\![\phi]\!]^{\mathrm{f}} +_{\langle c\rangle} \forall\mathrm{Pre}(x))
\end{aligned}
$$

Above, for a function $F\colon \mathcal{V}_S \to \mathcal{V}_S$, the notation $\mu x.F(x)$ indicates the unique (as $\langle c\rangle < 1$ for all $c \in A$) valuation x_* such that $x_* = F(x_*)$.

2.2 Semantics for Markov Processes

Given a finite set S, let $\mathrm{Distr}(S)$ be the set of probability distributions over S; for $a \in \mathrm{Distr}(S)$, we denote by $\mathrm{Supp}(a) = \{s \in S \mid a(s) > 0\}$ the support of a. A probability distribution a over S is *deterministic* if $a(s) \in \{0,1\}$ for all $s \in S$.

Markov decision processes. A *Markov decision process* (MDP) $\mathcal{S} = (S, \tau, \Sigma, [\cdot])$ consists of a set S of states, a probabilistic transition relation τ: $S \to 2^{\mathrm{Distr}(S)} \setminus \emptyset$, which assigns to each state a finite nonempty set of probability distributions over the successor states, a set Σ of propositions, and a function $[\cdot]\colon \Sigma \to \mathcal{V}_S$ that assigns to each proposition a valuation. The MDP $\mathcal{S}$ is *boolean* if for all propositions $r \in \Sigma$, the valuation $[r]$ is boolean. We denote by $|\tau|_b$ the length of the binary encoding of τ, defined by $\sum_{s\in S}\sum_{a\in\tau(s)}\sum_{t\in\mathrm{Supp}(a)} |a(t)|_b$, and we denote by $|[\cdot]|_b = \sum_{q\in\Sigma}\sum_{s\in S} |[q](s)|_b$ the size of the binary encoding of $[\cdot]$. Then, the binary size of $\mathcal{S}$ is given by $|\mathcal{S}|_b = |\tau|_b + |[\cdot]|_b$.

A finite (resp. infinite) *path* of $\mathcal{S}$ is a finite (resp. infinite) sequence $s_0 s_1 s_2 \ldots s_m$ (resp. $s_0 s_1 s_2 \ldots$) of states such that for all $i < m$ (resp. $i \in \mathbb{N}$) there is $a_i \in \tau(s_i)$ with $s_{i+1} \in \mathrm{Supp}(a_i)$. We denote by *FTraj* and *Traj* the sets of finite and infinite paths of $\mathcal{S}$; for $s \in S$, we denote by Traj_s the infinite paths starting from s. A *strategy* π for $\mathcal{S}$ is a mapping from *FTraj* to $\mathrm{Distr}(\bigcup_{s\in S} \tau(s))$: once the MDP has followed the path $s_0 s_1 \ldots s_m \in \mathit{FTraj}$, the strategy π prescribes the probability $\pi(s_0 s_1 \ldots s_m)(a)$ of using a next-state distribution $a \in \tau(s_m)$. For all $s_0 s_1 \ldots s_m \in \mathit{FTraj}$, we require that $\mathrm{Supp}(\pi(s_0 s_1 \ldots s_m)) \subseteq \tau(s_m)$. Thus, under strategy π, after following a finite path $s_0 s_1 \ldots s_m$ the MDP takes a transition to state s_{m+1} with probability $\sum_{a\in\tau(s_m)} a(s_{m+1}) \cdot \pi(s_0 s_1 \ldots s_m)(a)$. We denote by Π the set of all strategies for $\mathcal{S}$. The transition probabilities corresponding to a strategy π, together with an initial state s, give rise to a probability space $(\mathit{Traj}_s, \mathcal{B}_s, \mathrm{Pr}_s^\pi)$, where $\mathcal{B}_s$ is the set of measurable subsets of 2^{Traj_s}, and Pr_s^π is the probability measure over $\mathcal{B}_s$ induced by the next-state transition probabilities described above [12, 18]. Given a random variable X over this probability space, we denote its expected value by $\mathrm{E}_s^\pi[X]$. For $i \in \mathbb{N}$, the random variable Z_i: $\mathit{Traj}_s \to S$ defined by $Z_i(s_0 s_1 \ldots) = s_i$ yields the state of the stochastic process after i steps.

Special cases of MDPs: Markov chains and transition systems. Markov chains and LTSs can be defined as special cases of MDPs. An MDP $\mathcal{S} = (S, \tau, \Sigma, [\cdot])$ is a *Markov chain* if $|\tau(s)| = 1$ for all $s \in S$. It is customary to specify the probabilistic structure of a Markov chain via its *probability transition matrix* $P = [p_{s,t}]_{s,t\in S}$, defined for all $s, t \in S$ by $p_{s,t} = a(t)$, where a is the unique distribution $a \in \tau(s)$. An initial state $s \in S$ completely determines a probability space $(\mathit{Traj}_s, \mathcal{B}_s, \mathrm{Pr}_s)$, and for a random variable X over this probability space, we let $\mathrm{E}_s[X]$ denote its expectation. An MDP $\mathcal{S} = (S, \tau, \Sigma, [\cdot])$ is an LTS if, for all $s \in S$ and all $a \in \tau(s)$, the distribution a is deterministic; in that case, we define δ: $S \to 2^S$ by $\delta(s) = \{t \in S \mid \exists a \in \tau(s).\, a(t) = 1\}$.

The path semantics. The DCTL formulas over (Σ, A) are evaluated with respect to a MDP $\mathcal{S} = (S, \tau, \Sigma, [\cdot])$ and with respect to a parameter interpretation $\langle\cdot\rangle \in \mathcal{I}_A$. The semantics $[\![\psi]\!]^{\mathrm{p}}$ of a path formula ψ is defined as for LTSs; we note that $[\![\psi]\!]^{\mathrm{p}}$ is a random variable over the probability space $(\mathit{Traj}_s, \mathcal{B}_s, \mathrm{Pr}_s)$. Every state formula ϕ defines a valuation $[\![\phi]\!]^{\mathrm{p}} \in \mathcal{V}_S$: the clauses for propositions, boolean operators, and $\oplus_c$ are as for LTSs; the clauses for $\exists$ and $\forall$ are as follows:

$$[\![\exists\psi]\!]^{\mathrm{p}}(s) \;=\; \sup\{\mathrm{E}_s^\pi([\![\psi]\!]^{\mathrm{p}}) \mid \pi \in \Pi\}, \qquad [\![\forall\psi]\!]^{\mathrm{p}}(s) \;=\; \inf\{\mathrm{E}_s^\pi([\![\psi]\!]^{\mathrm{p}}) \mid \pi \in \Pi\}.$$

The fixpoint semantics. Given a valuation x: $S \to [0,1]$, we denote by $\exists\mathrm{Pre}(x)$: $S \to [0,1]$ the valuation defined by $\exists\mathrm{Pre}(x)(s) = \max_{a\in\tau(s)} \sum_{t\in S} x(t) \cdot a(t)$, and we denote by $\forall\mathrm{Pre}(x)$: $S \to [0,1]$ the valuation defined by $\forall\mathrm{Pre}(x)(s) = \min_{a\in\tau(s)} \sum_{t\in S} x(t)a(t)$. With this notation, the fixpoint semantics $[\![\cdot]\!]^{\mathrm{f}}$ is defined by the same clauses as for LTSs.

2.3 Properties of DCTL

Duality laws. For all state formulas ϕ_1, ϕ_2 over (Σ, A), all MDPs with propositions Σ, and all contractive parameter interpretations of A and $* \in \{\mathrm{p}, \mathrm{f}\}$, we have the following equivalences: $[\![\neg\exists\Diamond_c\phi]\!]^* = [\![\forall\Box_c\neg\phi]\!]^*$, $[\![\neg\exists\Box_c\phi]\!]^* = [\![\forall\Diamond_c\neg\phi]\!]^*$, and $[\![\neg\exists\triangle_c\phi]\!]^* = [\![\forall\triangle_c\neg\phi]\!]^*$. In particular, we see that $\triangle_c$ is self-dual and that a minimalist definition of DCTL will omit one of $\{\mathrm{T}, \mathrm{F}\}$, one of $\{\vee, \wedge\}$, and one of $\{\exists, \forall, \Diamond, \Box\}$.

Comparing both semantics. We show that the path and fixpoint semantics coincide over transition systems, and over Markov systems with boolean propositions (for non-nested formulas), but do not coincide in general over (non-boolean) Markov chains. This result is surprising, as it indicates that the standard connection between CTL and μ-calculus breaks down as soon as we consider *both* probabilistic systems and quantitative valuations. Since discounting plays no role in the proof of the theorem, an analogous result holds also for the logic without the operator $\triangle$ under no discounting. On the other hand, the theorem states that the two semantics always coincide for the $\triangle_c$ operator.

Theorem 1. *The following assertions hold:*

1. *For all LTSs with propositions Σ, all contractive parameter interpretations of A, and all* DCTL *formulas ϕ over (Σ, A), we have $[\![\phi]\!]^{\mathrm{p}} = [\![\phi]\!]^{\mathrm{f}}$.*
2. *For all boolean MDPs with propositions Σ, all contractive parameter interpretations of A, and all* DCTL *formulas ϕ over (Σ, A) that contain no nesting of path quantifiers, we have $[\![\phi]\!]^{\mathrm{p}} = [\![\phi]\!]^{\mathrm{f}}$.*
3. *There is a Markov chain $\mathcal{S}$ with propositions Σ, a contractive parameter interpretation A, and a* DCTL *formula ϕ over (Σ, A) such that $[\![\phi]\!]^{\mathrm{p}} \neq [\![\phi]\!]^{\mathrm{f}}$.*
4. *For all MDPs with propositions Σ, all contractive parameter interpretations of A, and all $r \in \Sigma$, we have $[\![\exists\triangle_c r]\!]^{\mathrm{p}} = [\![\exists\triangle_c r]\!]^{\mathrm{f}}$ and $[\![\forall\triangle_c r]\!]^{\mathrm{p}} = [\![\forall\triangle_c r]\!]^{\mathrm{f}}$.*

The example for part 3 of the theorem was given in the introduction.

Robustness. Consider two MDPs $\mathcal{S} = (S, \tau, \Sigma, [\cdot])$ and $\mathcal{S}' = (S, \tau', \Sigma, [\cdot]')$ with the same state space S and the same set Σ of propositions. We define $\|\mathcal{S}, \mathcal{S}'\| = \max_{s\in S}\{\max_{r\in\Sigma} |[r](s) - [r]'(s)|, \max_{a\in\tau(s)} \min_{b\in\tau'(s)} \sum_{s'\in S} |a(s') - b(s')|, \max_{b\in\tau'(s)} \min_{a\in\tau(s)} \sum_{s'\in S} |a(s') - b(s')|\}$. It is not difficult to see that $\|\cdot,\cdot\|$ is a metric on MDPs. For an MDP $\mathcal{S}$ and a parameter interpretation $\langle\cdot\rangle$, we write $[\![\cdot]\!]^{\mathrm{f}}_{\mathcal{S},\langle\cdot\rangle}$ and $[\![\cdot]\!]^{\mathrm{p}}_{\mathcal{S},\langle\cdot\rangle}$ to denote the two semantics functions defined on $\mathcal{S}$ with respect to $\langle\cdot\rangle$.

Theorem 2. *Let $\mathcal{S}$ and $\mathcal{S}'$ be two MDPs with state space S, and let $\langle\cdot\rangle$ be a contractive parameter interpretation.*

1. *For all $\epsilon > 0$, there is a $\delta > 0$ such that for all* DCTL *formulas ϕ and all states $s \in S$, if $\|\mathcal{S}, \mathcal{S}'\| \le \delta$, then $|[\![\phi]\!]^{\mathrm{f}}_{\mathcal{S},\langle\cdot\rangle}(s) - [\![\phi]\!]^{\mathrm{f}}_{\mathcal{S}',\langle\cdot\rangle}(s)| \le \epsilon$.*
2. *Let Φ be a set of* DCTL *formulas such that the maximum nesting depth of every formula in Φ is k. For all $\epsilon > 0$, there is a $\delta > 0$ such that for all formulas $\phi \in \Phi$ and all states $s \in S$, if $\|\mathcal{S}, \mathcal{S}'\| \le \delta$, then $|[\![\phi]\!]^{\mathrm{p}}_{\mathcal{S},\langle\cdot\rangle}(s) - [\![\phi]\!]^{\mathrm{p}}_{\mathcal{S}',\langle\cdot\rangle}(s)| \le \epsilon$.*

Notice that we get the continuity statement for the path semantics only for sets of formulas with bounded nesting depth. To see this, consider the three-state

Markov chain $\mathcal{S} = (\{s_0, s_1, s_2\}, \tau, \{r\}, [\cdot])$, where $\tau(s_0)$ is the distribution that chooses s_1 with probability $1 - \epsilon$ and chooses s_2 with probability ϵ, and $\tau(s_i)$ chooses s_i with probability 1 for $i = 1, 2$. Let $[r](s_0) = [r](s_1) = 0$ and $[r](s_2) = 1$. Consider the Markov chain $\mathcal{S}'$ that differs from $\mathcal{S}$ in that $\tau(s_0)$ chooses s_1 with probability 1. Then $\|\mathcal{S}, \mathcal{S}'\| = \epsilon$. Now consider the formulas $(\exists\Diamond_c)^n r$, for $n \geq 1$. Let $x_n = [\![(\exists\Diamond_c)^n r]\!]^{\mathrm{p}}_{\mathcal{S},\langle\cdot\rangle}(s_0)$. Then $x_{n+1} = (1-\epsilon) \cdot x_n + \langle c \rangle \cdot \epsilon$, and the limit as n goes to ∞ is $\langle c \rangle$. On the other hand, $[\![(\exists\Diamond_c)^n r]\!]^{\mathrm{p}}_{\mathcal{S}',\langle\cdot\rangle}(s_0) = 0$ for all n.

3 Model Checking DCTL

The model-checking problem of a DCTL formula ϕ over an LTS, a Markov chain, or an MDP with respect to one of the two semantics $* \in \{\mathrm{p}, \mathrm{f}\}$ consists in computing the value $[\![\phi]\!]^*(s)$ for all states s of the system under consideration. Similar to CTL model checking [4], we recursively consider one of the subformulas ψ of ϕ and compute the valuation $[\![\psi]\!]^*$. Then we replace ψ in ϕ by a new proposition p_ψ with $[p_\psi] = [\![\psi]\!]^*$. Because of the duality laws stated in Section 2.3, it suffices to focus on model checking formulas of the forms $\exists\Diamond_c r$, $\forall\Diamond_c r$, and $\forall\triangle_c r$, for a proposition $r \in \Sigma$. We will present the algorithms, for both semantics, over transition systems in Section 3.1, over Markov chains in Section 3.2, and over MDPs in Section 3.3.

Thoughout this section, we fix a parameter interpretation $\langle\cdot\rangle$, a set of propositions Σ, a proposition $r \in \Sigma$, and a parameter c and write $[r] = q$ and $\langle c \rangle = \alpha$. We omit the superscripts p and f and just write $[\![\cdot]\!]$ if the path and fixpoint semantics coincide. We restrict our attention to the case of finite-state systems and $\alpha < 1$; the case $\alpha = 1$ is treated in [6]. For complexity analyses, we assume that operations on reals (comparison, addition, and multiplication) can be performed in constant time; in other words, we provide the asymptotic complexity of each algorithm in terms of the number of arithmetic operations.

3.1 Model Checking DCTL over Transition Systems

We fix an LTS $\mathcal{S} = (S, \delta, \Sigma, [\cdot])$. As stated in Theorem 1, the two semantics of DCTL coincide over LTSs. Hence, we need only one algorithm to model check a formula in either semantics.

Model checking $\exists\Diamond$ and $\forall\Diamond$ in both semantics. The fixpoint semantics of DCTL suggests an iterative algorithm for evaluating formulas. In particular, $[\![\exists\Diamond_c r]\!]^{\mathrm{f}} = \lim_{n\to\infty} v_n$, where $v_0(s) = q(s)$, and $v_{n+1}(s) = q(s) \sqcup \alpha \cdot \max\{v_n(s') \mid s' \in \delta(s)\}$ for all $n \geq 0$. Over LTSs, the fixpoint is reached in a finite number of steps, namely, $[\![\exists\Diamond_c r]\!] = v_{|S|}$. To see this, observe that the value $[\![\exists\Diamond_c r]\!]^{\mathrm{f}}(s)$, the maximal (discounted) maximum over all paths from s, is obtained at a state in an acyclic prefix of some path from s. The argument that $[\![\forall\Diamond_c r]\!] = v_{|S|}$, where $v_{n+1}(s) = q(s) \sqcup \alpha \cdot \min\{v_n(s') \mid s' \in \delta(s)\}$, is slightly more involved. The value $[\![\forall\Diamond_c r]\!]^{\mathrm{f}}(s)$, the minimal (discounted) maximum over all paths from s, is again obtained at a state s' in an acyclic prefix of some path ρ from s. This is because if some state s'' were repeated on ρ before s', then the path ρ' that results from ρ by infinitely visiting s'' (and never visiting s') would achieve a smaller (discounted) maximum than ρ.

Model checking $\forall\triangle$ in both semantics. Computing $[\![\forall\triangle_c r]\!](s)$ consists in minimizing the (discounted) average $[\![\triangle_c r]\!]$ over the paths from s. As observed by [19] for the non-discounted case, the minimal discounted average is obtained on a path ρ' from s which, after some prefix ρ keeps repeating some simple cycle ℓ. Hence ℓ contains at most $|S|$ states. To find ρ', we use two steps. In the first phase, we find for each state s the simple cycle ℓ starting at s with the minimal discounted average. In the second phase, we find the best prefix-cycle combination $\rho\ell^\omega$.

Phase 1. We need to compute $L_\alpha(s) = \min\{[\![\triangle_c r]\!]^{\mathrm{p}}(\rho) \mid \rho \in \mathit{Traj}_s \text{ and } \rho = (s_0 s_1 s_2 \dots s_{n-1})^\omega \text{ and } n \le |S|\}$, where the value $[\![\triangle_c r]\!]^{\mathrm{p}}(\rho)$ is given by $\frac{1-\alpha}{1-\alpha^n} \cdot \sum_{i=0}^{n-1} \alpha^i \cdot q(s_i)$. Consider the recursion $v_0(s,s') = 0$ and $v_{n+1}(s,s') = q(s) + \alpha \cdot \min\{v_n(t,s') \mid t \in \delta(s)\}$. Then $v_n(s,s')$ minimizes $\sum_{i=0}^{n-1} \alpha^i \cdot q(s_i)$ over all finite paths $s_0 s_1 \dots s_n$ with $s_0 = s$ and $s_n = s'$. Hence

$$L_\alpha(s) = (1-\alpha) \cdot \min\left\{ \tfrac{v_1(s,s)}{1-\alpha^1}, \tfrac{v_2(s,s)}{1-\alpha^2}, \dots, \tfrac{v_{|S|-1}(s,s)}{1-\alpha^{|S|-1}} \right\}.$$

For a fixed state s', computing $\min\{v_n(t,s') \mid t \in \delta(s)\}$ for all $s \in S$ can be done in $O(|\delta|)$ time. Therefore, v_{n+1} is obtained from v_n in $O(|S|^2 + |S| \cdot |\delta|) = O(|S| \cdot |\delta|)$ time. Hence, the computation of $v_{|S|}$ and L_α requires $O(|S|^2 \cdot |\delta|)$ time.

Phase 2. After a prefix of length n, the cost $L_\alpha(s)$ of repeating a cycle at state s has to be discounted by α^n, which is exactly the factor by which we discount $q(s)$ after taking that prefix. Hence, we modify the original LTS $\mathcal{S}$ into an LTS $\mathcal{S}^+$, as follows. For every state $s \in S$, we add a copy $\hat{s}$ whose weight $w^+(\hat{s})$ we set to $L_\alpha(s)$; the weights $w^+(s)$ of states $s \in S$ remain $q(s)$. Moreover, for every $t \in S$ and $s \in \delta(t)$, we add $\hat{s}$ as a successor to t, that is, $\delta^+(t) = \delta(t) \cup \{\hat{s} \mid s \in \delta(t)\}$ and $\delta^+(\hat{s}) = \{\hat{s}\}$. Taking the transition from t to $\hat{s}$ corresponds to moving to s and repeating the optimal cycle from there. We find the value of the optimal prefix-cycle combination starting from s as the *discounted distance* from s to $\hat{S} = \{\hat{s} \mid s \in S\}$ in the modified graph $\mathcal{S}^+$ with weights w^+. Formally, given an LTS $\mathcal{S}$, a state s, a weight function $w\colon S \to \mathbb{R}^{\ge 0}$, a discount factor α, and a target set T, the minimal discounted distance from s to T is $d(s) = \min\{\sum_{i=0}^{n-1} \alpha^i \cdot w(s_i) \mid s_0 s_1 \dots s_{n-1} \in \mathit{FTraj}(s) \text{ and } s_{n-1} \in T\}$. The value of $d(s)$ for $s \in S$ is computed by the call $\mathit{DiscountedDistance}(\mathcal{S}^+, w^+, \alpha, \hat{S})$ to the algorithm below, which is a discounted version of the Bellman-Ford algorithm for finding shortest paths. Our algorithm performs backward computation from the set T, because discounted shortest paths (i.e., paths whose discounted distance is minimal among all paths with the same first and last state) are closed under suffixes, but not under prefixes.

```
function DiscountedDistance(S, w, α, T) :
   for each t ∈ S do
      if t ∈ T then d(t) := w(t) else d(t) := ∞;
   for i := 1 to |S| − 1 do
      for each s ∈ S and s' ∈ δ(s) do
         if d(s) > w(s) + α · d(s') then d(s) := w(s) + α · d(s');
   return d.
```

Like the standard version, discounted Bellman-Ford runs in $O(|S|\cdot|\delta|)$ time. Thus, the complexity of computing $[\![\forall\triangle_c r]\!]$ is dominated by the first phase.

Complexity of Dctl **model checking over LTSs.** The overall complexity of model checking a Dctl formula is polynomial in the size of the system and the size of the formula.

Theorem 3. *Given a* Dctl *formula ϕ, an LTS $\mathcal{S} = (S, \delta, P, [\cdot])$, and a parameter interpretation $\langle\cdot\rangle$, the problem of model checking ϕ over $\mathcal{S}$ with respect to $\langle\cdot\rangle$ can be solved in time $O(|S|^2\cdot|\delta|\cdot|\phi|)$.*

3.2 Model Checking Dctl over Markov Chains

As stated by Theorem 1, the path and fixpoint semantics over Markov chains coincide for the formula $\exists\triangle_c r$. Hence, we present below one algorithm for model checking this formula over Markov chains in either semantics. By contrast, the path and the fixpoint semantics over Markov chains may differ for the formulas $\exists\Diamond_c r$ and $\forall\Diamond_c r$. Hence, we need to provide algorithms for both semantics. Because of the absence of nondeterministic choice in a Markov chain, $[\![\exists\Diamond_c r]\!]^* = [\![\forall\Diamond_c r]\!]^*$ for $* \in \{\mathrm{f}, \mathrm{p}\}$; so giving algorithms for $\exists\Diamond_c$ suffices. This section gives the algorithm for model checking $\exists\Diamond_c r$ over a Markov chain with respect to the path semantics; the model-checking algorithm for $\exists\Diamond_c r$ in the fixpoint semantics is a special case of the algorithm for MDPs presented in Section 3.3. It is an open problem whether model checking $\exists\Diamond_c r$ in the fixpoint semantics can be improved over Markov chains. In the following, we consider a fixed Markov chain $(S, \tau, \Sigma, [\cdot])$ and its probability transition matrix P. We write I for the identity matrix.

Model checking $\exists\Diamond$ in the path semantics. When evaluating $[\![\exists\Diamond_c r]\!]^{\mathrm{p}}$ in a state s, we start with the initial estimate $q(s)$. If s is the state $s_{\max}$ with the maximum value of q, the initial estimate is the correct value. If s has the second largest value for q, the estimate can only be improved if $s_{\max}$ is hit within a certain number l of steps, namely, before the discount α^l becomes smaller than $\frac{q(s)}{q(s_{\max})}$. This argument is recursively applied to all states.

Let $s_1, \dots, s_n$ be an ordering of the states in S such that $q(s_1) \geq q(s_2) \geq \dots \geq q(s_n)$. We use integers as matrix indices, thus writing $P(i,j)$ for p_{s_i,s_j}. For all $1 \leq j < i \leq n$ such that $q(s_i) > 0$, let $k_{i,j} = \lfloor \log_\alpha \frac{q(s_i)}{q(s_j)} \rfloor$, with the convention that $\log_\alpha 0 = \infty$. Let $v(s_i) = [\![\exists\Diamond_\alpha r]\!]^{\mathrm{p}}(s_i)$. Then, $v(s_1) = q(s_1)$, and we can express the value of $v(s_i)$ in terms of the values $v(s_1), \dots, v(s_{i-1})$. Let $K = \max\{k_{i,j} \mid k_{i,j} < \infty\}$, and for all $l > 0$, let $B_l^i = \{s_j \mid 1 \leq j < i \text{ and } 1 \leq l \leq k_{i,j}\}$. Intuitively, B_l^i contains those states that, if hit in exactly l steps from s_i, can increase the value of $v(s_i)$. For the (arbitrary) state s_i, the following holds:

$$v(s_i) = q(s_i) \cdot \mathit{stay}^i + \sum_{j=1}^{i-1} v(s_j) \cdot \sum_{l=1}^{k_{i,j}} \alpha^l \cdot \mathit{go}_{j,l}^i,$$

where $stay^i = \Pr_{s_i}\left[\bigwedge_{l>0} Z_l \notin B_l^i\right]$ and $go_{j,l}^i = \Pr_{s_i}\left[Z_l = s_j \wedge \bigwedge_{m=1}^{l-1} Z_m \notin B_m^i\right]$. It is easy to check that $stay^i + \sum_{j=1}^{i-1}\sum_{l=1}^{k_{i,j}} go_{j,l}^i = 1$. We proceed in two phases. The first phase handles states s_i with $q(s_i) > 0$. Since the sequence $(B_l^i)_{l>0}$ is decreasing, it can have at most $|S|$ different values. It follows that there exist $m \leq |S|$ and $b_1^i \leq \cdots \leq b_{m+1}^i \in \mathbb{N}$ and sets $X_1^i, \ldots, X_m^i \subseteq S$ such that $b_1^i = 1$, $b_{m+1}^i = K+1$, and for all $k = 1, \ldots, m$ and all $b_k^i \leq l < b_{k+1}^i$, we have $B_l^i = X_k^i$. Let P_k^i be the substochastic matrix obtained from P by disabling all transitions leading to states in X_k^i, i.e., $P_k^i(j', j) = 0$ for all j', j with $s_j \in X_k^i$. Then, for given $b_k^i \leq l < b_{k+1}^i$, we have

$$go_{j,l}^i = \left((P_1^i)^{b_2^i - b_1^i} \cdot (P_2^i)^{b_3^i - b_2^i} \cdot \ldots \cdot (P_{k-1}^i)^{b_k^i - b_{k-1}^i} \cdot (P_k^i)^{l - b_k^i} \cdot P\right)(i, j).$$

Let $m_j^i = \max\{k \mid s_j \in X_k^i\}$ be the index of the last X_k^i containing s_j. We have

$$\sum_{l=1}^{k_{i,j}} \alpha^l \cdot go_{j,l}^i = \sum_{k=1}^{m_j^i} \sum_{l=b_k^i}^{b_{k+1}^i - 1} \alpha^l \cdot go_{j,l}^i =$$

$$\Big(\sum_{k=1}^{m_j^i} \alpha^{b_k^i} \cdot (P_1^i)^{b_2^i - b_1^i} \cdot (P_2^i)^{b_3^i - b_2^i} \cdot \ldots \cdot (P_{k-1}^i)^{b_k^i - b_{k-1}^i} \cdot \Big(\sum_{l=0}^{b_{k+1}^i - b_k^i - 1} \alpha^l \cdot (P_k^i)^l \cdot P\Big)\Big)(i, j) =$$

$$\Big(\sum_{k=1}^{m_j^i} \alpha^{b_k^i} \cdot (P_1^i)^{b_2^i - b_1^i} \cdot (P_2^i)^{b_3^i - b_2^i} \cdot \ldots \cdot (P_{k-1}^i)^{b_k^i - b_{k-1}^i} \cdot \Big(\frac{I - (\alpha P_k^i)^{b_{k+1}^i - b_k^i}}{I - \alpha P_k^i}\Big) \cdot P\Big)(i, j).$$

Each matrix $(P_k^i)^{b_{k+1}^i - b_k^i}$ can be computed by repeated squaring in time $O(|S|^3 \cdot \log b_k^i)$. Some further calculations show that, for a fixed i, both $\sum_{l=1}^{k_{i,j}} \alpha^l \cdot go_{j,l}^i$ and $\sum_{l=1}^{k_{i,j}} go_{j,l}^i$ can be computed in time $O(|S|^4 \cdot \log K)$. The value $stay^i$ is given by $1 - \sum_{j,l} go_{j,l}^i$. The total complexity of this phase is thus $O(|S|^5 \cdot \log K)$.

The second phase considers those states s_i with $q(s_i) = 0$. Let u be the smallest index i such that $q(s_i) = 0$. Now, $go_{j,l}^i$ is the probability of hitting s_j after exactly l steps, meanwhile avoiding all states with indices smaller than u. To compute $v(s_i)$ efficiently, we define a stochastic matrix P_0 from P by adding an absorbing state s_{n+1} and using s_{n+1} to turn all states s_j with $j < u$ into transient states (so, for all $j < u$, $P_0(j, n+1) = 1$ and $P_0(j, j') = 0$ for $j' \neq n+1$). Also, we set $\bar{v}$ to be the column vector with $\bar{v}_j = v(s_j)$ (computed in phase 1), if $j < u$, and $\bar{v}_j = 0$ otherwise. Then,

$$v(s_i) = \sum_{j=1}^{u-1} v(s_j) \cdot \sum_{l=1}^{\infty} \alpha^l \cdot (P_0)^l(i, j) = ((I - \alpha P_0)^{-1} \cdot \bar{v})(i).$$

Solving the system (3.2) takes time $O(|S|^3)$ using LUP decomposition. The time spent in the two phases amounts to $O(|S|^5 \cdot \log K)$. An alternative algorithm, which we omit for space constraints, takes time $O(|S|^3 \cdot K)$ and may thus perform better in practical applications.

Model checking $\forall\triangle$ in both semantics. Since the two semantics of the formula $\forall\triangle_c r$ coincide, a single model-checking algorithm suffices for both semantics and can be computed by the following classical equation [9]. If we let $[\![\exists\triangle_c r]\!]$ and q denote column vectors, we have $[\![\exists\triangle_c r]\!] = (1-\alpha)\cdot\sum_{i\geq 0}\alpha^i P^i q = (1-\alpha)\cdot(I-\alpha P)^{-1}\cdot q$. Thus, we can compute the value $[\![\exists\triangle_c r]\!](s)$ for each state $s \in S$ by solving a linear system with $|S|$ variables. This takes time $O(|S|^{\log_2 7})$ using Strassen's algorithm or $O(|S|^3)$ using LUP decomposition.

Complexity of DCTL model checking over Markov chains. The overall complexity is polynomial in the size of the system. With respect to the size of the formula, the complexity is polynomial for the fixpoint semantics, and exponential for the path semantics. The latter result is due to the fact that, in the path semantics, the number of arithmetic operations is polynomial in the size of the bit-wise encoding of the valuations, and these encodings grow exponentially with respect to the number of nestings of temporal operators.

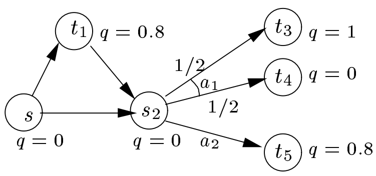
Theorem 4. *Given a* DCTL *formula ϕ, a Markov chain $\mathcal{S} = (S, \tau, P, [\cdot])$, and a parameter interpretation $\langle\cdot\rangle$, the problem of model checking ϕ over $\mathcal{S}$ with respect to $\langle\cdot\rangle$ can be solved in time polynomial in $|S|$, $|[\cdot]|_b$, and $|\langle\cdot\rangle|_b$. Furthermore, the problem of model checking ϕ over $\mathcal{S}$ with respect to $\langle\cdot\rangle$ can be solved in time polynomial in $|\phi|$ in the fixpoint semantics, and exponential in $|\phi|$ in the path semantics.*

3.3 Model Checking DCTL over Markov Decision Processes

As it is the case for Markov chains, also for MDPs the path and fixpoint semantics do not coincide for the formulas $\exists\diamond_c r$ and $\forall\diamond_c r$, so that separate algorithms are needed. The two semantics do coincide for the formula $\forall\triangle_c r$ on MDPs, hence one algorithm suffices. We consider a fixed MDP $\mathcal{S} = (S, \tau, \Sigma, [\cdot])$.

Model checking $\exists\diamond$ and $\forall\diamond$ in the path semantics. If $\alpha = 0$, then trivially $[\![\exists\diamond_c r]\!]^{\mathrm{p}}(s) = [\![\forall\diamond_c r]\!]^{\mathrm{p}}(s) = q(s)$ at all $s \in S$, so in the following we assume $0 < \alpha < 1$. The problem of computing $[\![\exists\diamond_c r]\!]^{\mathrm{p}}$ on an MDP can be viewed as an optimization problem, where the goal is to maximize the expected value of the sup of q over a path. As a preliminary step to solve the problem, we note that in general the optimal strategy is *history dependent,* that is, the choice of distribution at a state depends in general on the past sequence of states visited by the path.

Example 1. Consider the system depicted on the right and assume $\alpha = 1$. The optimal choice in state s_2 depends on whether t_1 was hit or not. If it was, the current sup is 0.8 and the best choice is a_1, because with probability $\frac{1}{2}$ the sup will increase to 1. If t_1 was not hit, the best choice is a_2, because it gives a certain gain of 0.8, rather than an expected gain of 0.5. The same argument holds if α is sufficiently close to 1.



While the above example indicates that the optimal strategy is in general history-dependent, it also suggests that all a strategy needs to remember is the sup value that has occurred so far along the path. For $\pi \in \Pi$, $s \in S$, and $x \in \mathbb{R}$, we define $\mathrm{Esup}^\pi(s,x) = \mathrm{E}^\pi_s[x \sqcup \sup_{i>0} \alpha^i q(Z_i)]$; the term x corresponds to the (appropriately discounted) sup value that has occurred so far in the past of a path. Obviously, $[\![\exists\Diamond_c r]\!]^{\mathrm{P}}(s) = \sup_{\pi\in\Pi} \mathrm{Esup}^\pi(s,q(s))$ and $[\![\forall\Diamond_c r]\!]^{\mathrm{P}}(s) = \inf_{\pi\in\Pi} \mathrm{Esup}^\pi(s,q(s))$. The optimization problem to compute these quantities is phrased in terms of the variables $v(s,x)$, representing the value of $\mathrm{Esup}(s,x)$. Since we are ultimately interested in the value of $\mathrm{Esup}(s,q(s))$ for $s \in S$, and since if $x \geq 1$ we have $\mathrm{Esup}^{\pi'}(t,x) = x$ for all $t \in S$ and $\pi' \in \Pi$, it suffices to consider values for x that belong to the finite set $X = \{q(s)/\alpha^k \mid s \in S \wedge k \in \mathbb{N} \wedge q(s)/\alpha^k < 1\}$. We set up the following set of equations in the variables $\{v(s,x) \mid s \in S \wedge x \in X\}$:

$$v(s,x) = \begin{cases} x & \text{if } x \geq \alpha \\ x \sqcup \alpha \cdot \max_{a\in\tau(s)} \sum_{t\in S} v(t, \frac{x}{\alpha} \sqcup q(t)) \cdot a(t) & \text{otherwise} \end{cases} \tag{1}$$

Intuitively, the equations (1) can be understood as follows. At a state $s = s_m$ of a path $s_0 s_1 \ldots$, the quantity $v(s_m,x)$ represents the maximum over all strategies of $\sup_{i>0} E_{s_m}[\alpha^i q(Z_i)]$ given that $\sup_{0\leq i\leq m} \alpha^{-i} q(s_{m-i}) = x$. The recursion (1) then relates $v(s,x)$ to $v(t,y)$ at the successors t of s, where at t we consider the new conditioning $y = x/\alpha \sqcup q(t)$, thus discounting x by α^{-1} (as s is one step before t), and taking into account the value $q(t)$ seen at t. The following lemma relates the least fixpoint of (1) to $[\![\exists\Diamond_c r]\!]^{\mathrm{P}}$.

Lemma 1. *Let $\{v^*(s,x) \mid s \in S \wedge x \in X\}$ be the least (pointwise) fixpoint of the set of equations (1). Then, we have $[\![\exists\Diamond_c r]\!]^{\mathrm{P}}(s) = v^*(s,q(s))$ for all $s \in S$.*

Proof. Let $X' = \{q(s)/\alpha^k \mid s \in S \wedge k \in \mathbb{N}\}$ be the set that, compared to X, also includes elements greater than 1. We consider an iterative evaluation of the least fixpoint (1), given for all $s \in S$, $x \in X'$, and $k \geq 0$, by $v_0(s,x) = x$ and

$$v_{n+1}(s,x) = \alpha \cdot \max_{a\in\tau(s)} \sum_{t\in S} v_n(t, \frac{x}{\alpha} \sqcup q(t)) \cdot a(t). \tag{2}$$

The proof consists of two parts: (i) showing that for all $s \in S$ and $x \in X'$, there is a strategy $\pi^* \in \Pi$ such that $\mathrm{E}^{\pi^*}_s[x \sqcup \sup_{0<i\leq n} \alpha^i \cdot q(Z_i)] = v_n(s,x)$, and (ii) showing that for all $\pi \in \Pi$, $s \in S$, and $x \in X'$, we have $\mathrm{E}^\pi_s[x \sqcup \sup_{0<i\leq n} \alpha^i \cdot q(Z_i)] \leq v_n(s,x)$. Once (i) and (ii) are proved, the result follows from

$$\lim_{n\to\infty} v_n = v^*, \qquad \lim_{n\to\infty} \mathrm{E}^\pi_s[x \sqcup \sup_{0\leq i\leq n} \alpha^i \cdot q(Z_i)] = \mathrm{E}^\pi_s[x \sqcup \sup_{i\geq 0} \alpha^i \cdot q(Z_i)].$$

We prove only (i), since the proof of (ii) is similar. The strategy π^* is in general a function of $\langle s,x\rangle \in S \times X'$. We define it inductively: π^*_0 is arbitrary; for $n \geq 0$, π^*_{n+1} first chooses a distribution $a \in \tau(s)$ that realizes the maximum in (2), and then upon a transition from s to some $t \in S$, proceeds as π^*_n from $\langle t, \frac{x}{\alpha} \sqcup q(t)\rangle$. Part (i) follows by induction, noting that for all $n \geq 0$, all $s \in S$ and all $x \in X'$ we have:

$$v_{n+1}(s,x) = \alpha \cdot \max_{a \in \tau(s)} \sum_{t \in S} v_n(t, \frac{x}{\alpha} \sqcup q(t)) \cdot a(t)$$

$$= \alpha \cdot \max_{a \in \tau(s)} \sum_{t \in S} \mathrm{E}_t^{\pi^*_n} \Big[\frac{x}{\alpha} \sqcup q(t) \sqcup \sup_{0<i\leq n} \alpha^i \cdot q(Z_i)\Big] \cdot a(t) = \mathrm{E}_s^{\pi^*_{n+1}} \Big[x \sqcup \sup_{0<i\leq n} \alpha^i \cdot q(Z_i)\Big] \blacksquare$$

To compute $[\![\forall\Diamond_c r]\!]^{\mathrm{p}}$, we simply replace $\max_{a\in\tau(s)}$ with $\min_{a\in\tau(s)}$ in (1), and again consider the least fixpoint. The least fixpoints for $[\![\exists\Diamond_c r]\!]^{\mathrm{p}}$ and $[\![\forall\Diamond_c r]\!]^{\mathrm{p}}$ can be computed by linear programming, following a standard approach. For example, to compute $[\![\exists\Diamond_c r]\!]^{\mathrm{p}}$ we consider the following linear-programming problem in the set $\{v(s,x) \mid s \in S \wedge x \in X\}$ of variables: minimize $\sum_{s\in S}\sum_{x\in X} v(s,x)$ subject to

$$v(s,x) \geq x, \qquad v(s,x) \geq \alpha \cdot \sum_{t\in S} \tilde{v}(t, \frac{x}{\alpha} \sqcup q(t)) \cdot a(t)$$

for all $s \in S$, all $x \in X$, and all $a \in \tau(s)$, where $\tilde{v}(t,x)$ is 1 if $x \geq 1$, and is $v(t,x)$ otherwise. Denoting by $\{\hat{v}(s,x) \mid s \in S \wedge x \in X\}$ an optimal solution, we have $\hat{v}(s,q(s)) = v^*(s,q(s)) = [\![\exists\Diamond_c r]\!]^{\mathrm{p}}(s)$. This linear program contains at most $2\cdot|S|\cdot|X|$ variables. We have $|X| = -|S|\cdot\log_\alpha q_{min}$, where $q_{min} = \min\{q(s) \mid s \in S \wedge q(s) > 0\}$. Hence, if q-values are encoded in binary notation, the number of variables in the encoding is linear in the size of the input encoding of the MDP.

Model checking $\exists\Diamond$ and $\forall\Diamond$ in the fixpoint semantics. The computation of $[\![\exists\Diamond_c r]\!]^{\mathrm{f}}$ and $[\![\forall\Diamond_c r]\!]^{\mathrm{f}}$ on an MDP can be performed by transforming the fixpoints into linear-programming problems, following a standard approach. For example, for $[\![\forall\Diamond_c r]\!]^{\mathrm{f}}$ we consider the following linear programming problem in the set $\{v(s), u(s) \mid s \in S\}$ of variables: minimize $\sum_{s\in S}(v(s) - u(s))$ subject to $v(s) \geq q(s)$, $v(s) \geq u(s)$, and $u(s) \leq \alpha \cdot \sum_{t\in S} v(t)\cdot a(t)$, for all $s \in S$ and $a \in \tau(s)$. Denoting by $\{v^*(s), u^*(s) \in \mathbb{R} \mid s \in S\}$ an optimal solution, we have $[\![\forall\Diamond_c r]\!]^{\mathrm{f}}(s) = v^*(s)$ at all $s \in S$. Again, this can be solved in time polynomial in $|\mathcal{S}|_b$ and $|\alpha|_b$.

Model checking $\forall\triangle$ in both semantics. With the two semantics for $\forall\triangle_c r$ coinciding, a single algorithm suffices for model checking $\forall\triangle$ in both semantics. The fixpoint semantics of this formula immediately suggests an algorithm based on standard methods used for discounted long-run average problems [2].

Lemma 2. *Consider the following linear-programming problem in the set $\{v(s) \mid s \in S\}$ of variables: maximize $\sum_{s\in S} v(s)$ subject to $v(s) \leq (1-\alpha)\cdot q(s) + \alpha \cdot \sum_{t\in S} v(t)\cdot a(t)$, for all $s \in S$ and $a \in \tau(s)$. Denoting by $\{v^*(s) \mid s \in S\}$ an optimal solution, we have $[\![\forall\triangle_c r]\!](s) = v^*(s)$ for all states $s \in S$.*

Complexity of DCTL model checking over MDPs. As for Markov chains, also for MDPs the model checking problem can be solved in time polynomial in the size of the system, and in the fixpoint (resp. path) semantics, in time polynomial (resp. exponential) in the size of the DCTL formula.

Theorem 5. *Given a* DCTL *formula* ϕ, *an MDP* $\mathcal{S} = (S, \tau, P, [\cdot])$, *and a parameter interpretation* $\langle\cdot\rangle$, *the problem of model checking* ϕ *over* $\mathcal{S}$ *with respect to* $\langle\cdot\rangle$ *can be solved in time polynomial in* $|S|$, $|[\cdot]|_b$, *and* $|\langle\cdot\rangle|_b$. *Furthermore, the problem of model checking* ϕ *over* $\mathcal{S}$ *with respect to* $\langle\cdot\rangle$ *can be solved in time polynomial in* $|\phi|$ *in the fixpoint semantics, and exponential in* $|\phi|$ *in the path semantics.*

References

1. C. Baier, B.R. Haverkort, H. Hermanns, and J.-P. Katoen. Model checking continuous-time Markov chains by transient analysis. In *Computer-Aided Verification*, LNCS 1855, pages 358–372. Springer, 2000.
2. D.P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 1995. Volumes I and II.
3. A. Bianco and L. de Alfaro. Model checking of probabilistic and nondeterministic systems. In *Foundations of Software Technology and Theoretical Computer Science*, LNCS 1026, pages 499–513. Springer, 1995.
4. E.M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 1999.
5. L. de Alfaro. *Formal Verification of Probabilistic Systems*. PhD thesis. Technical Report STAN-CS-TR-98-1601, Stanford University, 1997.
6. L. de Alfaro, M. Faella, T.A. Henzinger, R. Majumdar, and M. Stoelinga. Model checking discounted temporal properties. Technical Report UCSC-CRL-03-12, University of California, Santa Cruz, 2003.
7. L. de Alfaro, T.A. Henzinger, and R. Majumdar. Discounting the future in systems theory. In *Automata, Languages, and Programming*, LNCS 2719, pages 1022–1037. Springer, 2003.
8. J. Desharnais, A. Edalat, and P. Panangaden. Bisimulation for labeled Markov processes. *Information and Computation*, 179:163–193, 2002.
9. J. Filar and K. Vrieze. *Competitive Markov Decision Processes*. Springer, 1997.
10. H. Hansson. *Time and Probabilities in Formal Design of Distributed Systems*. Elsevier, 1994.
11. M. Huth and M.Z. Kwiatkowska. Quantitative analysis and model checking. In *Proc. Logic in Computer Science*, pages 111–122. IEEE, 1997.
12. J.G. Kemeny, J.L. Snell, and A.W. Knapp. *Denumerable Markov Chains*. Van Nostrand, 1966.
13. D. Kozen. A probabilistic PDL. In *Proc. Theory of Computing*, pages 291–297. ACM, 1983.
14. M.Z. Kwiatkowska. Model checking for probability and time: From theory to practice. In *Proc. Logic in Computer Science*, pages 351–360. IEEE, 2003.
15. Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer, 1991.
16. A. McIver. Reasoning about efficiency within a probabilistic μ-calculus. In *Proc. Probabilistic Methods in Verification*, pages 45–58. Technical Report CSR-98-4, University of Birmingham, 1998.
17. A. McIver and C. Morgan. Games, probability, and the quantitative μ-calculus. In *Logic Programming, Artificial Intelligence, and Reasoning*, LNCS 2514, pages 292–310. Springer, 2002.
18. D. Williams. *Probability with Martingales*. Cambridge University Press, 1991.
19. U. Zwick and M.S. Paterson. The complexity of mean-payoff games on graphs. *Theoretical Computer Science*, 158:343–359, 1996.