Available online at www.sciencedirect.com
SCIENCE DIRECT®

Electronic Notes in Theoretical Computer Science 130 (2005) 23–37

Electronic Notes in Theoretical Computer Science
www.elsevier.com/locate/entcs

Memoryless Strategies for Stochastic Games via Domain Theory

Carroll Morgan[1],[2]

Dept. of Computer Science and Engineering
University of New South Wales
Sydney 2052 Australia

Annabelle McIver[1],[3]

Department of Computer Science
Macquarie University
Sydney 2109 Australia

Abstract

In Formal Methods we use mathematical structures to model real systems we want to build, or to reason about; in this paper we are concerned principally with *game-based* models.
In an earlier work [16] we presented an approach to stochastic *minimising* games based on our logic for probabilistic programs [17,11]. The contribution here is to extend our logical approach to *maximising* games. That maximising and minimising are not (simply) duals is due to the use of least fixed-points in both cases, as is normal for iterating programs. (For duality, we would have to use greatest fixed-points in the maximising case.)
The significance of that extension is that it makes a further link between the program-logical approach and the more general methods for finding *control strategies* that "solve" games of this kind, in the sense of giving the player a "recipe" which, if followed, will deliver the theoretically optimal result.

Keywords: Probabilistic semantics, stochastic games, Markov decision processes.

[1] We are grateful for the support of the *Australian Research Council* (ARC) under its Discovery Grant DP0345457.

[2] Email: carrollm@cse.unsw.edu.au

[3] Email: anabel@ics.mq.edu.au

doi:10.1016/j.entcs.2005.03.003

1 Introduction

Techniques such as Z [19] and B [1] model a real-world system by postulating a mathematical model of its state, and formulating abstracted operations that change that state in a way that is supposed to correspond to the actual operations that can be observed or carried out on the real system.

In this work we are concerned with more general systems than the above, those that suggest (to their observers) the process of playing a game, possibly involving demonic nondeterminism (the environment's trying to frustrate the user), angelic nondeterminism (the user's attempts to succeed in spite of that) and probabilistic nondeterminism (out of control of either "player", but at least quantifiable).

A recent trend, especially for systems like these, is to characterise correctness (and refinement) not as simply "achieving a postcondition", the traditional view, but as maximising a reward (of which the traditional view is a special case: get \$1 if you achieve the postcondition; get \$0 if you don't).

The "classical" approach to such problems is via linear-programming techniques and *Markov Decision Processes* (MDP's) [5] rather than programming logic; but we have found that the approach can be enhanced by using our experience with program logic in other areas. Similarly, we find that the classical work done already can inform our design of the programming logics and even in some cases suggest techniques for proving properties of them [4, for example].

In particular, we have shown how two-player turn-based demonic /angelic /probabilistic games correspond to expressions in our *quantitative μ-calculus* $qM\mu$ [14,15,10], which is itself a superset of a quantitative temporal logic qTL.

Thus we are following a more general approach than is typical of *e.g.* Z or B, where we observe a system, describe its behaviour as a state-with-operations system (necessarily an informal step), codify that description as a mathematical model, and then use mathematical tools (theorems) to manipulate the formal text (refinement *etc.*) Finally, we translate our results back into the real world, as predictions about the system's behaviour or as code that will implement it.

Here, we observe a system, describe its behaviour as a game (necessarily an informal step), codify that description as a mathematical model (a game tree), and then use mathematical tools (theorems) to manipulate the formal text (via quantitative logic corresponding to the tree). And finally, again, we translate our results back into the real world, as predictions about the system's behaviour or as code that will implement it.

The specific technical result of this paper is to show how the program-logic approach can *predict* that the result of “playing the game” has a certain optimal bound, and can *prove* that there is a memoryless player’s strategy that will realise it.

2 Existing theory, and our contribution

In the mathematical style, *Markov Decision Processes* address realistic problems related to resource contention within a probabilistic context. A (*non-competitive*) *MDP*, in its most abstract form, describes the actions of a solitary gambler and the consequences of his selected strategy in placing bets.

The use of *MDP*’s brings with it a long-established body of algorithmic techniques, based on linear programming, for quantifying certain well-defined properties; however, even simple arguments can be intricate and difficult to follow — largely because of the need to include all the details of each *MDP*. There is no built-in facility of abstraction.

Our approach to this specification and analysis problem is to abstract the properties of a probabilistic system using a *quantitative logic*, in which the expressions include real values that reflect the probabilistic transitions in the underlying system. As with program logic generally, it leads to simpler specification and analysis techniques; and the basis for the logic is a domain-theoretic model of the underlying system. The abstraction is included naturally.

Abstraction is of course the essence of logic; and in this paper we concentrate on the domain-theoretic models over which such quantitative logics [14,15,11] are built. That domain-theoretic “spin” allows simpler proofs of some of the key existence theorems.

In the Computer-Science style, early treatments containing both probability and nondeterminism sought to introduce *probabilistic models* of distributed systems, which went a long way towards clarifying the many issues related to how probability and nondeterminism should interact.[4] A major contribution of the work was to provide specification languages and verification techniques for probabilistic properties in general [18] and temporal properties in particular [20,2,6]. Examples of such properties include “a given set of system states will be reached with probability one” and “a given property is invariant with such-and-such a probability”. Later work extended the application areas to include average expected times between events and shortest-path problems [3].

[4] When we write “nondeterminism” on its own, we mean demonic rather than probabilistic.

A key component of those treatments was to provide actual algorithms for computing the properties specified, and many model-checking systems are based on the ideas put forward in that early work.

In this paper we treat mathematical-style problems but with Computer-Science-style tools, in particular domain theory: we address the problem of showing that a memoryless strategy exists for a maximising player within an iterating game whose value for infinite behaviour is zero (thus a least fixed-point).

We use left-associating dot "." for functional application, and write quantifications in the order *bound variable* then *constraint* then *term*, with scoping always explicit, given either by parentheses $(\cdots)$ or set-comprehension braces $\{\cdots\}$: thus the set of positive cubes is $\{i{:}\,\mathbb{Z} \mid i > 0 \cdot i^3\}$.

3 The classical perspective: Markov Decision Processes

3.1 The basics

In this section we set out the basic theory of *MDP*'s, following closely the approach taken by Filar and Vrieze [5], although we do not use their notation exactly.

Let S be a *finite state space*, let L be a finite set of *labels* and let $\overline{S}$ be the set of *discrete probability distributions* over S.[5]

Definition 3.1 A finite-state *Markov Decision Process* over S is a pair (ρ, ω), where ρ is a transition system of type $L \times S \to \overline{S}$ and ω is a reward function of type $L \times S \to \mathbb{R}$. The transition system ρ maps every label l in L and state s in S to a probability distribution $\rho.l.s$ over final states, while ω specifies an immediate reward $\omega.l.s$ following the selection of label l in state s.

The following metaphor, which complements the above definition, provides a useful intuition based on a gambler's playing a game of chance:

- From any position s he may choose from a selection of $\#S$-sided dice each with the elements of S written on it; each die has some label l and is biased according to the distribution $\rho.l.s$.
- Each choice of die incurs an immediate cost or reward of $\omega.l.s$.

[5] Later we will extend $\overline{S}$ to include *sub*-distributions, those that sum to no more than one (rather than exactly to one).

- His total expected winnings are then determined with respect to a *payoff function* $\phi: S \to \mathbb{R}$ by his immediate reward and the final state s' he reaches after rolling his chosen die: in total, he wins $\phi.s'$ plus the reward.

Thus if in the *MDP* (ρ, ω) the gambler plays from initial state s, and chooses to roll the die labelled l, then his total *expected one-step winnings* is given by

$$(1) \qquad M.l.s.\phi \quad := \quad \omega.l.s + \int_{\rho.l.s} \phi \quad ,$$

where we are thus letting M stand for the function of l, s and ϕ that (ρ, ω) represents as above. In general we write $\int_{\Delta} \phi$ for the *expected value* of random variable ϕ over distribution Δ. In the finite case, as here, the integral in (1) is simply $\sum_{s':S}(\rho.l.s.s' \times \phi.s')$, since the distribution Δ produced by the *MDP* is just $\rho.l.s$ and finite expected values are just weighted sums.

A gambler's *greatest* expected one-step winnings (1) from initial state s is the best he can do over all choices of l (*i.e.* by selecting the die most biased in his favour, given ω and the payoff function ϕ); it is given by the function $M.\sqcup$, defined as follows: [6]

Definition 3.2 For *MDP* $M = (\rho, \omega)$ we define

$$M.\sqcup.\phi.s \quad := \quad (\sqcup\, l: L \cdot M.l.\phi.s) \quad = \quad (\sqcup\, l: L \cdot \omega.l.s + \int_{\rho.l.s} \phi \quad) \; .$$

We illustrate the above definitions with an example due to Filar, reproduced here in Fig. 1 (but slightly modified). Let the *MDP* in Fig. 1 be M, and consider a payoff function ϕ given by

$$\phi.s_1 := 10, \quad \phi.s_2 := 20, \quad \phi.s_3 := 30 \; .$$

Then the gambler's greatest expected winnings from (initial) s_2 are:

$$\begin{array}{ll} & M.\sqcup.\phi.s_2 \\ = & (10 + 1\times 20) \;\sqcup\; (5 + 0.8\times 10 + 0.2\times 20) \\ = & 30 \sqcup 17 \\ = & 30 \; . \end{array}$$

[6] Strictly speaking the symbols "$M.\sqcup$" are the name of the function we are defining, based on M, rather than indicating some label "$\sqcup$" passed to M as its first argument. This unorthodox but suggestive notation will be useful later.

	State s_1	*State* s_2	*State* s_3
Label l_1	0 $(0.9, 0, 0.1)$	10 $(0, 1, 0)$	
Label l_2	15 $(0, 1, 0)$	5 $(0.8, 0.2, 0)$	25 $(0.9, 0.1, 0)$

In this representation a box portrays an action in a state and its reward/transition (upper-left/lower-right) consequences. For example, in the above the choice of Label l_1 results in a reward of $\omega.l_1.s_1 = 0$ from State s_1 and transition probabilities $\rho.l_1.s_1.s_1 = 0.9$, $\rho.l_1.s_1.s_2 = 0$ and $\rho.l_1.s_1.s_3 = 0.1$.

Where there are differing numbers of alternatives (*e.g.* only one label in state s_3), we regard the missing labels as giving "copies" of the effects of labels already there for that state. In that way in the theory we can assume a fixed number of labels throughout.

Fig. 1. A summable Markov Decision Process [5, Example 2.1.1 p10]

3.2 Discounted-cost games

Suppose now that the gambler plays repeatedly, accumulating his rewards as he goes, to give an overall "multi-step" winnings. At each state he chooses a label determining his next move; let his *strategy* be $\sigma \colon S \to L$ representing "in advance" the choice of label he would make in state s. The gambler's problem now is to evaluate the effectiveness of his strategy, were he to follow it, in terms of his overall winnings. A strategy σ, transition system ρ and an initial state s together determine a well-defined probability distribution over sequences of states in S that will be observed by the gambler as he plays the game. From that distribution the expected overall winnings can be computed, although for some games it could be undefined. One kind of game that guarantees definedness is the *discounted-cost* game, which we now describe.

Definition 3.3 A *discounted MDP* is a tuple (ρ, ω, β) where (ρ, ω) is an (undiscounted) *MDP* and the extra component β with $0 \leq \beta < 1$ is a *discount factor*. Play is according to some strategy σ as above, but the rewards are "discounted" according to when they occur: a reward after n steps is worth only β^n of its nominal value as given by ω.

The discount has the effect that the later a reward is received the less it is worth: thus its value declines at a fixed percentage. Suppose the gambler starts playing the *MDP*given by (ρ, ω, β) from initial state s_0 at Step 0: the discount is ineffective for his first move (it is β^0), so that his first reward is

$\omega.(\sigma.s_0).s_0$, as in the single-step case.

For his second move, from some state s_1 chosen probabilistically from the distribution $\rho.(\sigma.s_0).s_0$, he will receive (only) the discounted value $\beta \times \omega.(\sigma.s_1).s_1$; therefore his *expected* overall winnings when he has reached State s_1 is

$$\omega.(\sigma.s_0).s_0 \quad + \quad \beta \times \int_{\rho.(\sigma.s_0).s_0} \omega.(\sigma.s_1).s_1 \; \mathrm{d}s_1 \; . \qquad [7]$$

Similar —but increasingly complex— expressions give the expected total reward for later steps. Note that here we are not using a final payoff function ϕ, since the game is unending.

In general for discounted $M = (\rho, \omega, \beta)$ we write $M^*.\sigma.n.s$ for the overall expected winnings up to Step n, with strategy σ.[8]

The gambler's overall expected winnings, were he to play according to strategy σ, is thus the limit of the above as n increases, that is $\lim_{n\to\infty} M^*.\sigma.n.s$, which accordingly we write $M^*.\sigma.\infty.s$.

It turns out that for discounted-cost games, the σ-like strategies are sufficient to realise the the maximum possible overall expected winnings; with the next theorem we sketch the traditional proof. (That is, we do not need more powerful strategies of the kind sketched in Sec. 3.3 below.)

Theorem 3.4 *A gambler playing the discounted game set out in Def. 3.3 can achieve the greatest possible expected winnings by following a σ-like strategy.*

Proof. The standard proof uses linear-programming techniques; for details refer to Filar [5]. □

In the sections to come, we look at a domain-theoretic proof of that same result, and then we extend it to show that the result applies even when the game is not discounted.

3.3 On more general strategies

The "σ-like" strategies we considered above are called *pure*, because they are not themselves probabilistic, *stationary* because although they depend on the state they do not depend on the "time" at which that state is reached and

[7] In the usual way, we read "$\mathrm{d}s_1$" as indicating a functional abstraction of the expression $\omega.(\sigma.s_1).s_1$ over its parameter s_1. Thus the argument of the integral sign is a function over S, as it should be.

[8] Continuing our abuse of notation, we use the superscript in M^* to remind us that in this case we are iterating the single-step M. The σ in place of the label indicates that the labels are chosen according to that strategy; and we write (integer) n in the place formerly occupied by payoff ϕ, since both are to do with the stopping of the *MDP*.

memoryless because they depend only on the current state and not on any previous states the iterated process M^* may have passed through.

There are more general kinds of strategies, however; and the more information a strategy can take into account, the more "powerful" it is in the sense of being able to force the *MDP* to display certain behaviours. The most interesting behaviour, for us, is the one which realises the greatest expected payoff $M^*.\sqcup.\infty$. We give a domain-theoretic style proof below that in fact the σ-like strategies —pure, stationary and memoryless— are sufficient for that, provided we restrict ourselves to non-negative rewards.

The further significance of such strategies is that they place the overall problem back in the context of finite state spaces, since for a finite state space S, and finitely many labels, there are only finitely many strategies of that type. (On the other hand, even generalising slightly, to "impure" or probabilistic strategies, makes the number of possible strategies infinite.) And it means that gamblers who can achieve their optimal reward by playing such strategies can evaluate their optimal expected payoffs using linear-programming techniques.

4 The Computer-Science perspective: domain theory

We are studying "infinite event-horizon games", where there is no definite time horizon which will assure convergence: the more general framework provided by a domain-theoretic perspective clarifies the conditions under which convergence is guaranteed.

We begin by recalling some elementary facts of domain theory.

4.1 A domain of quantitative functions

A *partially ordered set* is a pair $(C, \sqsubseteq)$ where C is a set, and $\sqsubseteq$ is a reflexive, anti-symmetric and transitive order on C. The order is said to be *complete* if any chain in C has a least upper bound in C, where a *chain* is a totally-ordered subset. For us here, a *domain* is a complete partially-ordered set with a least element.

Given a finite state space S we construct $\mathcal{E}S$, the set of $\mathbb{R}_{\geq}^{\infty}$-valued functions over S, where $\mathbb{R}_{\geq}^{\infty}$ is the set of non-negative reals completed with a top-element ∞; they are called *payoffs*. The only use of ∞ we make here is to ensure $(\mathbb{R}_{\geq}^{\infty}, \leq)$ is complete (thus a domain), *i.e.* the only property of ∞ that we use is that it dominates all proper reals. Next we lift $\leq$ on $\mathbb{R}_{\geq}^{\infty}$ to $\mathcal{E}S$ pointwise, defining for $\phi, \phi' \colon \mathcal{E}S$ that

$$\phi \sqsubseteq \phi' \quad \textit{iff} \quad (\forall s : S \cdot \phi.s \leq \phi'.s) \ .$$

We use a different symbol ("$\sqsubseteq$" rather than "$\leq$") to remind us that the lifted order is only partial.

For real k we write $\underline{k}$ for the constant-valued payoff with value k for all states. The least element of $\mathcal{E}S$ is thus $\underline{0}$, and the greatest is $\underline{\infty}$, and we note the following:

Lemma 4.1 *The space* $(\mathcal{E}S, \sqsubseteq)$ *is a domain.*

We will see that the optimal value of an *MDP* can be found as a "fixed-point" in the domain of payoffs, though in the standard theory the existence of those fixed-points is proved by appealing to a "contraction-mapping theorem" which requires the single-step game to define a contraction mapping over a suitable metric.

In the domain-theoretic setting however the existence of fixed-points is more straightforward. In general, functions over a domain are *guaranteed* to have fixed-points if they are "monotone", where (in general) a function $F\colon C \to C$ is said to be *monotone* if $c \sqsubseteq c'$ implies $F.c \sqsubseteq F.c'$. We write $\mu.F$ for the *least fixed-point* of F in C — it is the $\sqsubseteq$-smallest element c such that $F.c = c$.

Happily, discounted accumulation games, like the ones $M^*.\sqcup.\infty$ we studied above, have monotone single-step functions $M.\sqcup$ over the order $\sqsubseteq$ on $\mathcal{E}S$, and thus we have immediately that their least fixed-points exist. Of course in some cases that least fixed-point may be infinite (as could be the case in accumulation games if no discount were applied).

Taking the least fixed-point may seem an arbitrary decision, but in fact it agrees with the operational specification that a player receives only the rewards explicitly determined by his appeal to ω as he "passes through", and a payoff of zero is waiting for him "at infinity". (Note also that $\underline{\infty}$ is the greatest fixed-point of any M, making that an unlikely choice.) Indeed the properties of least fixed-points justify precisely the constraints used in the linear-programming techniques that are commonly employed to compute the greatest overall value. Translated into the *MDP* accumulation-game notation of Sec. 3.1, the *lfp-induction* property says that the the least fixed-point of $M.\sqcup$ is the least payoff ϕ that satisfies

$$M.\sqcup.\phi \quad \sqsubseteq \quad \phi\ ,$$

which inequation, being a comparison of functions, can be interpreted as a set of inequalities, one for each element of the state space.

The connection with accumulation games is given by the following lemma:

Lemma 4.2 *Let M be an MDP with non-negative reward function ω. Then the greatest expected overall winnings of the iterated game $M^*.\sqcup.\infty$ is the least* *fixed-point of the payoff-to-payoff function $M.\sqcup$. Similarly, for strategy σ, the expected winnings of the iterated game $M^*.\sigma.\infty$ is the least fixed-point of the function $M.\sigma$.*

Proof. The value of either form of game — either $M^*.\sqcup.\infty$ or $M^*.\sqcap.\infty$ — is given as the limit of an infinite sum; exactly the same sum occurs in the iterated calculation of the corresponding fixed point. □

The value of the game is the basis on which a player determines how he should play — if he plays a bad strategy then his ultimate payoff will be far from the value of the game. On the other hand if he uses a good strategy then his expected payoff will be close to the value of the game.

Linear programming techniques [5] can be used to compute his optimal strategy; unfortunately for many sizeable systems the enormous state space makes linear-programming solutions infeasible, so that alternative computation strategies need to be employed. Here the domain-theoretic approach will suggest using *iteration*, and this is indeed the approach taken in practice (for example *PRISM* [8,7]).

Determining the optimal strategy remains a problem, however, and we turn our attention to that topic next.

4.2 Memoryless strategies for discounted games

We begin first by defining a discounted game in domain-theoretic terms, so that we can study its properties. First we define the "norm" $\|f\|$ of any function f from the state space into the reals (including negatives).

Definition 4.3 The *norm* of a function $f\colon S \to \mathbb{R}$ is the largest absolute value in its codomain: we have

$$\|\phi\| \quad := \quad (\sqcup s\colon S \cdot |X.s|) \ ,$$

where $|\cdot|$ denotes absolute value.

Clearly $\|f\|$ is always finite for finite S.

We now define "contraction mapping".

Definition 4.4 Function F from $\mathcal{E}S$ to $\mathcal{E}S$ is a *contraction* mapping with *factor* β if $0 \le \beta < 1$ and

$$\|F.\phi - F.\phi'\| \quad \le \quad \beta \times \|\phi - \phi'\|$$

for all payoffs ϕ, ϕ'.

We now show how we can regard a discounted *MDP* as a contraction mapping. Given an *undiscounted* (ρ, ω), and a discount factor β, form the (again undiscounted) "β-scaled" *MDP* $(\beta \times \rho, \omega)$ by defining $(\beta \times \rho).l.s.s' := \beta \times (\rho.l.s.s')$, thus making the "distributions" $(\beta \times \rho).l.s$ sum to β rather than to 1. Since $\beta < 1$ we call them *sub-distributions*; regarded as a matrix $(\beta \times \rho).l$ for any l would be called *sub-stochastic.* When M is (ρ, ω) we write $M_{\times\beta}$ for $(\beta \times \rho, \omega)$; note that only the transition probabilities, and not the reward, are scaled by β.

Note that we have the following property immediately.

Lemma 4.5 *The β-scaled $M_{\times\beta}$ formed from an undiscounted MDP M is dominated by M itself: for all payoffs ϕ we have*

$$M_{\times\beta}. \sqcup .\phi \quad \sqsubseteq \quad M.\sqcup.\phi \ .$$

For this we can write $M_{\times\beta}. \sqcup \ \sqsubseteq \ M.\sqcup$.

We observe now that $M_{\times\beta}.\sqcup$ is a contraction mapping with factor β.

Lemma 4.6 *For any MDP M, payoffs ϕ, ϕ' and real $0 \leq \beta$ we have that*

$$\|M_{\times\beta}. \sqcup .\phi \ - \ M_{\times\beta}. \sqcup .\phi'\| \quad \leq \quad \beta \times \|\phi - \phi'\| \ .$$

Thus whenever $\beta < 1$ we have that $M_{\times\beta}$ is a contraction mapping with respect to the metric $\|\cdot\|$.

Proof (sketch) Suppose the contrary, for a contradiction; let $\|\phi - \phi'\|$ be some real δ. Then for some state s we must have *wlog*

$$M_{\times\beta}. \sqcup .\phi.s \ + \ \beta \times \delta \quad < \quad M_{\times\beta}. \sqcup .\phi'.s \ .$$

Thus for some labels l, l' we have $M_{\times\beta}.l.\phi.s + \beta \times \delta \ < \ M_{\times\beta}.l'.\phi'.s$ and, furthermore, that $M_{\times\beta}.l.\phi.s \ \geq \ M_{\times\beta}.l'.\phi.s$. Thus

$$M_{\times\beta}.l'.\phi.s \ + \ \beta \times \delta \quad < \quad M_{\times\beta}.l'.\phi'.s \ ,$$

and the result now follows by arithmetic since $\sum(\beta \times \rho).l' \ \leq \ \beta$. □

Putting together Lemmas 4.2 and 4.6 gives us the following.

Lemma 4.7 *For any MDP M with scaling factor β we have that $M^*_{\times\beta}.\sqcup.\infty$ is the unique fixed-point of $M_{\times\beta}.\sqcup$.*

Proof (sketch) We have that $M_{\times\beta}.\sqcup$ is a contraction mapping over $\mathcal{E}S$; it is analytically continuous by construction (recall that L is finite); and all continuous contraction mappings over an analytically closed set, which $\mathcal{E}S$ is, have a unique fixed point. □

Finally, we observe that when we have unique fixed points the strategies are particularly easy to calculate.

Lemma 4.8 *Suppose that some MDP M is such that all its fixed-strategy instantiations $M.\sigma$ have unique fixed points. (This would be the case for example* *if M were of the form $M'_{\times\beta}$ for some M', since then all instantiations $M'_{\times\beta}.\sigma$ would be contraction mappings.)*

Define $\phi := \mu(M.\sqcup)$ and construct a strategy σ such that $M.\sigma.\phi = \phi$, easily done since we are considering a strategy for just one step of M with respect to a known payoff function ϕ.[9]

Then we have that σ realises the optimal value of the iterated MDP, that is it achieves ϕ: we have

$$M^*.\sqcup.\infty \quad = \quad \mu.(M.\sqcup) \quad = \quad \phi \quad = \quad \mu.(M.\sigma) \quad = \quad M^*.\sigma.\infty \ .$$

Proof. By construction of σ we know that ϕ is a fixed-point of $M.\sigma$, and by assumption $M.\sigma$ has only one fixed-point: thus in fact ϕ must be its least fixed point. □

That is, Lem. 4.8 tells us immediately how to compute the optimal strategy provided that the value of the game is known: it can be found by computing the strategy σ that solves the equation

$$M.\sigma.\phi \quad = \quad \phi \ ,$$

where ϕ is $\mu.(M.\sqcup)$ and, as we noted above, can be computed by iteration beforehand (*i.e.* without knowing σ).

4.3 Memoryless strategies for non-discounted *games*

We can prove the existence of stationary strategies for non-discounted games by approximating them with discounted games, as we now show.

For a general non-discounted *MDP* given by $M = (\rho, \omega)$ we can form a discounted *MDP* given by $M_{\times\beta}$ as before, and we noted then in Lem. 4.5 that $M_{\times\beta}.\sqcup \ \sqsubseteq \ M.\sqcup$. In fact we have the following stronger property.

Lemma 4.9 *For any (non-discounted) MDP M we have*

$$M.\sqcup \quad = \quad (\sqcup\beta \mid \beta < 1 \cdot M_{\times\beta}.\sqcup) \ .$$

Proof. We calculate directly, as follows: for any payoff ϕ,

$$\begin{aligned} & (\sqcup\beta \mid \beta < 1 \cdot M_{\times\beta}.\sqcup).\phi \\ = \ & (\sqcup\beta \mid \beta < 1 \cdot M_{\times\beta}.\sqcup.\phi) \\ = \ & (\sqcup\beta \mid \beta < 1 \cdot M.\sqcup.(\beta \times \phi)) \\ = \ & M.\sqcup.(\sqcup\beta \mid \beta < 1 \cdot \beta \times \phi) \\ = \ & M.\sqcup.\phi \ , \end{aligned}$$

[9] Actually ϕ is known "in principle", since $\mu.(M.\sqcup)$ itself must first be calculated.

where in the second-last step we rely on the continuity of $M.\sqcup$. □

We can now deduce quite easily the existence of memoryless strategies for non-discounted maximising *MDP*'s.

Lemma 4.10 *Let M be a (non-discounted) MDP; then $M^*.\sqcup$ has an optimal memoryless strategy.*

Proof. For any $\beta < 1$ we have from Lem. 4.6 that $M_{\times\beta}.\sqcup$ is a contraction mapping, and thus by Lem. 4.8 that $M^*_{\times\beta}.\sqcup$ has a memoryless optimal strategy σ_β, say, that is one that satisfies $M^*_{\times\beta}.\sqcup.\infty = M^*_{\times\beta}.\sigma_\beta.\infty$. We observe the relationships

$$M_{\times\beta}.\sigma_\beta \quad \sqsubseteq \quad M.\sigma_\beta \quad \sqsubseteq \quad M.\sqcup \ ,$$

where the first inequality relies on Lem. 4.5 (or Lem. 4.9), and the second holds because in fact $M.\sigma \sqsubseteq M.\sqcup$ for any σ.

By monotonicity of the least-fixed-point operator, we can immediately deduce a similar relationship between the corresponding least fixed-points, giving us

$$\mu.(M_{\times\beta}.\sigma_\beta) \quad \sqsubseteq \quad \mu.(M.\sigma_\beta) \quad \sqsubseteq \quad \mu.(M.\sqcup) \ .$$

Because of the way we chose σ_β, however, that immediately simplifies to

$$\mu.(M_{\times\beta}.\sqcup) \quad \sqsubseteq \quad \mu.(M.\sigma_\beta) \quad \sqsubseteq \quad \mu.(M.\sqcup) \ . \tag{2}$$

The result now follows from the following observations. Let β approach 1 in (2); we know from Lem. 4.9 that $M_{\times\beta}.\sqcup$ approaches $M.\sqcup$; by continuity of μ itself for this kind of limit,[10] we therefore know that $\mu.(M_{\times\beta}.\sqcup)$ approaches $\mu.(M.\sqcup)$, which is the right-hand side; thus the middle term $\mu.(M.\sigma_\beta)$, squashed between left and right, must approach $\mu.(M.\sqcup)$ in the limit as well. *But there are only finitely many σ's in total*, and so one of the σ_β's must actually attain the limit: let it be $\widehat{\sigma}$.

Thus $\mu.(M.\widehat{\sigma}) = \mu.(M.\sqcup)$, that is $M^*.\widehat{\sigma}.\infty = M^*.\sqcup.\infty$, and we are done. □

5 Conclusions

The direct connection between Markov processes and probabilistic-program semantics is only to be expected: both model a stochastic activity evolving in a series of discrete steps. Going "forwards", the Markov approach multiplies distributions by matrices, while the program-semantics approach (Kleisli-) composes state-to-distribution functions; going "backwards", the former multiplies payoffs by matrices, and the latter (functionally-) composes payoff transformers.

[10] This continuity of μ requires a separate proof.

In fact these four approaches correspond so well that there might not be much to choose for simplicity between them. When nondeterminism is introduced, however, the last — payoff transformers — is singled out because *it does not need to be extended.*

Markov processes must be extended to Markov *decision* processes, with their labels and strategies; and state-to-distribution functions become state-to-distribution relations. But payoff transformers retain their "shape", merely broadening the class transformers that are considered well-formed;[11] thus many of the simpler notions applicable to the deterministic case can be carried over unchanged.

An example of that — but not our topic in this report — is stationarity: it is well know that a long-term "stationary" distribution exists for any Markov process whose transition matrix satisfies certain conditions; but if those conditions are not met, the stationary distribution does not exist. McIver shows [9] that when recast as an payoff transformer the process has a stationary distribution in all cases, and considerable simplification results: the traditional conditions merely ensure that the distribution is "deterministic".

Our concern here has been the more general simplification that might result from treating the nondeterminism of *MDP*'s in the transformer style. In particular, the whole apparatus of state sequences, strategies and labels is "sucked up" into the provisions that transformers make automatically for nondeterminism. Rather than seek extremal strategies — which of necessity requires that strategies themselves be modelled explicitly — we look instead at extremal fixed-points of functions. The price we pay for that is the explicit modelling of the program lattice.

References

[1] Abrial, J.-R., "The B Book: Assigning Programs to Meanings," Cambridge University Press, 1996.

[2] Aziz, A., V. Singhal, F. B. R. Brayton and A. Sangiovanni-Vincentelli, *It usually works: The temporal logic of stochastic systems*, in: *Computer-Aided Verification, 7th Intl. Workshop*, LNCS **939** (1995), pp. 155–65.

[3] de Alfaro, L., *Computing minimum and maximum reachability times in probabilistic systems*, in: *Proceedings of CONCUR '99*, LNCS **1664** (1999), pp. 66–81.

[4] Everett, H., *Recursive games*, in: *Contributions to the Theory of Games III*, Ann. Math. Stud. **39**, Princeton University Press, 1957 pp. 47–78.

[5] Filar, J. and O. Vrieze, "Competitive Markov Decision Processes: Theory, Algorithms, and Applications," Springer Verlag, 1996.

[11] In standard programming the transformers drop their "disjunctivity" when nondeterminism is introduced, retaining conjunctivity; and deterministic payoff transformers are linear, becoming (only) sublinear when nondeterminism is introduced.

[6] Hansson, H. and B. Jonsson, *A logic for reasoning about time and reliability*, Formal Aspects of Computing **6** (1994), pp. 512–35.

[7] Kwiatkowska, M., *Model checking for probability and time: from theory to practice*, in: P. G. Kolaitis, editor, *Proc. Eighteenth Annual IEEE Symp. on Logic in Computer Science, LICS 2003* (2003), pp. 351–360.

[8] Kwiatkowska, M., G. Norman and D. Parker, *Probabilistic symbolic model checking with PRISM: A hybrid approach*, International Journal on Software Tools for Technology Transfer (STTT) **6** (2004), pp. 128–142.

[9] McIver, A., *A generalisation of stationary distributions, and probabilistic program algebra*, Electronic Notes in Theo. Comp. Sci. **45**, Elsevier, 2001 .

[10] McIver, A. and C. Morgan, *Games, probability and the quantitative μ-calculus qMu*, in: *Proc. LPAR*, LNAI **2514** (2002), pp. 292–310, revised and expanded at [12].

[11] McIver, A. and C. Morgan, "Abstraction, Refinement and Proof for Probabilistic Systems," Technical Monographs in Computer Science, Springer Verlag, New York, 2004.

[12] McIver, A. and C. Morgan, *Results on the quantitative μ-calculus qMμ* (2004), to appear in *ACM TOCL.*

[13] McIver, A., C. Morgan, J. Sanders and K. Seidel, *Probabilistic Systems Group: Collected reports*, `web.comlab.ox.ac.uk/oucl/research/areas/probs`.

[14] Morgan, C. and A. McIver, *A probabilistic temporal calculus based on expectations*, in: L. Groves and S. Reeves, editors, *Proceedings*, Discrete Mathematics and Computer Science (1997), pp. 4–22, available at [13, key PTL96].

[15] Morgan, C. and A. McIver, *An expectation-based model for probabilistic temporal logic*, Logic Journal of the IGPL **7** (1999), pp. 779–804, available at [13, key MM97].

[16] Morgan, C. and A. McIver, *Cost analysis of games using program logic*, in: *Proc. of the 8th Asia-Pacific Software Engineering Conference (APSEC 2001)*, 2001, abstract only: full text available at [13, key MDP01].

[17] Morgan, C., A. McIver and K. Seidel, *Probabilistic predicate transformers*, ACM Transactions on Programming Languages and Systems **18** (1996), pp. 325–53, `doi.acm.org/10.1145/229542.229547`.

[18] Segala, R., "Modeling and Verification of Randomized Distributed Real-Time Systems," Ph.D. thesis, MIT (1995).

[19] Spivey, J., "Understanding Z: a Specification Language and its Formal Semantics," Cambridge University Press, 1988.

[20] Vardi, M., *A temporal fixpoint calculus*, in: *Proc. 15th Ann. ACM Symp. on Principles of Programming Languages* (1988), pp. 250–9, extended abstract.