

# Chance and Mass Interpretations of Probabilities in Markov Decision Processes


Yun Chen Tsai ✉ 

National Institute of Informatics, Tokyo, Japan

SOKENDAI (The Graduate University for Advanced Studies), Kanagawa, Japan

Kittiphon Phalakarn ✉ 

National Institute of Informatics, Tokyo, Japan

S. Akshay ✉ 

Department of CSE, Indian Institute of Technology Bombay, Mumbai, India

Ichiro Hasuo ✉ 

National Institute of Informatics, Tokyo, Japan

SOKENDAI (The Graduate University for Advanced Studies), Kanagawa, Japan

Imiron Co., Ltd., Tokyo, Japan

---

## Abstract

Markov decision processes (MDPs) are a popular model for decision-making in the presence of uncertainty. The conventional view of MDPs in verification treats them as state transformers with probabilities defined over sequences of states and with schedulers making random choices. An alternative view, especially well-suited for modeling dynamical systems, defines MDPs as distribution transformers with schedulers distributing probability masses. Our main contribution is a unified semantical framework that accommodates these two views and two new ones. These four semantics of MDPs arise naturally through identifying different sources of randomness in an MDP (namely *schedulers*, *configurations*, and *transitions*) and providing different ways of interpreting these probabilities (called the *chance* and *mass* interpretations). These semantics are systematically unified through a mathematical construct called *chance-mass (CM) classifier*. As another main contribution, we study a reachability problem in each of the two new semantics, demonstrating their hardness and providing two algorithms for solving them.

**2012 ACM Subject Classification** Theory of computation → Random walks and Markov chains; Theory of computation → Verification by model checking

**Keywords and phrases** MDP, distribution transformer, antichain, template-based synthesis

**Digital Object Identifier** 10.4230/LIPIcs.CONCUR.2025.33

**Related Version** *Full Version*: <https://arxiv.org/abs/2506.10377>

**Funding** The authors are supported by ASPIRE Grant No. JPMJAP2301, and ERATO HASUO Metamathematics for Systems Design Project (No. JPMJER1603), JST. S.A. is supported by the SBI Foundation Hub for Data Science & Analytics, IIT Bombay. I.H. is supported by CREST ZT-IoT Project (No. JPMJCR21M3), JST.

## 1 Introduction

**MDPs and Conventional Semantics.** Markov decision processes (MDPs) are a classical model combining nondeterminism and randomness, with a rich theory and practical uses in a wide variety of applications [26]. MDPs are often viewed as state transformers where a move from one state to another is defined as follows: from a given state, an action is chosen by a scheduler out of multiple possible actions and then a probabilistic transition is made leading to a resulting state. In other words, a scheduler resolves the nondeterminism in the MDP to give rise to a Markov chain (MC), a purely probabilistic model. In general, a scheduler can also be randomized (also called a mixed scheduler), and thus the actions are chosen according to the probabilities it defines. See [9] for more discussion.



© Yun Chen Tsai, Kittiphon Phalakarn, S. Akshay, and Ichiro Hasuo;  
licensed under Creative Commons License CC-BY 4.0

36th International Conference on Concurrency Theory (CONCUR 2025).

Editors: Patricia Bouyer and Jaco van de Pol; Article No. 33; pp. 33:1–33:19

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

**MDPs as Distribution Transformers.** A different semantics of MDPs (and Markov chains), inspired by dynamical systems, considers them as *distribution transformers*. In this view, we start with a probability distribution or mass over the set of all states of the MDP, and according to the action chosen by a scheduler, this mass gets re-distributed over the states. Therefore, a one-step transition results in a new probability distribution over the set of states, in which sense the MDP is a linear transformer of distributions. This semantics is studied in multiple recent works [1, 2, 4–6, 10, 11, 19, 21] with applications ranging from AI, where swarms of robots are modeled, to biological systems, where concentrations of biochemical reagents or populations of cells are modeled. This semantics is also common, e.g., in statistics and the theory of dynamical systems, where mathematicians are often interested in *stationary distributions*, which are essentially fixed points of the distribution transformer.

**Reachability Problems in MDPs.** The *reachability problem* is fundamental in model checking. In the context of MDPs, it asks whether there exists a scheduler such that the MDP, following this scheduler from a given state, reaches a given *target state* with a certain probability. It is known that the reachability problem for MDPs can be solved in polynomial time, and well-established probabilistic model checkers such as PRISM [20] and Storm [17] can handle large instances efficiently.

However, it has been observed that when the reachability problem on MDPs is “lifted” and is asked under the distribution transformer semantics, it becomes significantly harder. The “lifted” problem asks for the reachability probability to a *target distribution*, instead of to a target state. The problem is significantly harder even for Markov chains; in particular, it is as hard as the Skolem problem, a long-standing open problem on linear recurrences [3, 23, 24].

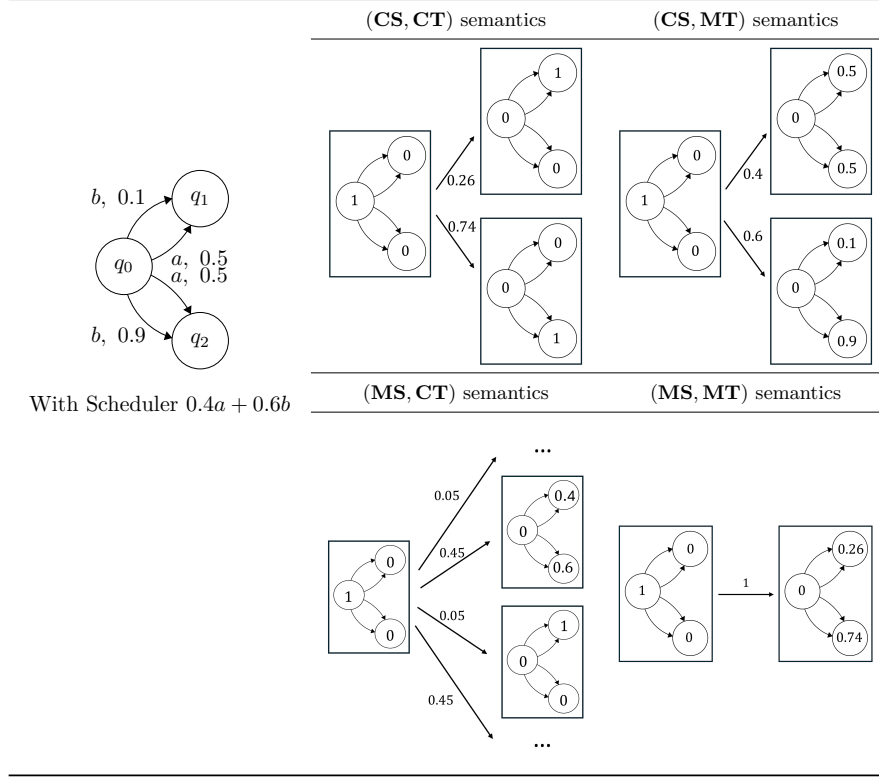
Motivated from these observations, we aim to develop a systematic framework that describes the two semantics and the “lifted” reachability problem. This leads to our main contribution: a unified framework for MDPs semantics.

**Four Semantics for MDPs: a Unified Framework.** We start by observing that there are two sources of probabilities in MDPs: those arising from randomized/mixed schedulers (such as 0.4 and 0.6 in Table 1) and the inherent transition probabilities (such as 0.1 and 0.9 in Table 1). On the other hand, these probabilities can be interpreted in two ways: the conventional manner, which we call the *chance interpretation*, models probabilities as random events like coin tosses; and the “distribution transformer” manner, which we call the *mass interpretation*, models the probabilities as the rate of splitting a mass of particles.

It is then natural to consider four variations of MDP semantics, where each of these two classes of probabilities – coming from the above two sources – is interpreted under either the *chance* or *mass* interpretation (cf. Table 1). For example, the conventional semantics corresponds to applying the *chance* interpretation to both scheduler and transition probabilities, as both schedulers and transitions are modeled as random events (like coin tosses). Similarly, the distribution transformer semantics corresponds to applying the *mass* interpretation to both scheduler and transition probabilities.

Let us explain the illustration in Table 1 in more detail. The left side of Table 1 shows a simple MDP with three states and two possible actions from  $q_0$ , where each action leads to a probabilistic transition over states  $q_1$  and  $q_2$ . A mixed scheduler is fixed, which chooses action  $a$  with probability 0.4 and action  $b$  with probability 0.6. Each of the four Markov chains (MCs) shown on the right side of Table 1, corresponding to each semantics, is called the *config MC* for the MDP and the specified scheduler. The intuition here is that these config MCs are Markov chains where each state represents a mass distribution over the state space of the MDP on the left, and they model how mass distributions over the MDP states evolve under the specified interpretation of probabilities.

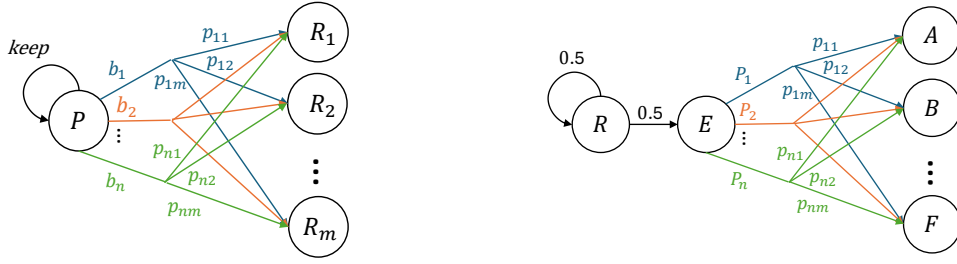
■ **Table 1** Examples of the four semantics of MDPs.



Let's take the (CS, MT), an abbreviation for **C**hance-interpreted **S**cheduler, **M**ass-interpreted **T**ransitions, semantics as an example. As suggested by its naming, this semantics interprets the probability from the scheduler as a random event, and the probability from the transition as a rate of splitting mass. In one step of the MDP, an (unfair) coin is first tossed to decide the action to be taken by the scheduler: a 0.4 chance of choosing action  $a$  and a 0.6 chance of choosing action  $b$ . This corresponds to the two transitions of the config MC as illustrated in Table 1. Once the action is determined, the mass of 1 at  $q_0$  is then split according to the transition probabilities, which yields a distribution of 0.5 at  $q_1$  and 0.5 at  $q_2$  in the case of action  $a$  (the 0.4 branch). The process for the case of action  $b$  is similar. These two distributions are the only distributions that can be reached by the MDP after one step if the probabilities are interpreted as specified.

Table 1 also clearly demonstrates that the reachability problem has different answers under different semantics. For example, the mass distribution “0.26 at  $q_1$ , 0.74 at  $q_2$ ” is reached only under the (MS, MT) semantics; the mass distribution “1 at  $q_1$ ” is reached with the probability 0.27 under the (CS, CT) semantics, while it is with the probability 0.05 under the (MS, CT) semantics.

**Examples.** All four semantics are not only theoretically natural, but also lead to different application scenarios. For the purely random (CS, CT) semantics, examples are abundant in the CS literature [9], and similarly for the purely distributional (MS, MT) semantics, examples are found in the dynamical systems and AI literature [5].



■ **Figure 1** *Casino* MDP for (MS, CT) sem. (left); *exam* MDP for (CS, MT) sem. (right).

**Casino.** An example suited for the (MS, CT) semantics is the *casino* MDP in Fig. 1. Here,  $P$  is the initial state, from which actions  $b_i$  ( $i \in \{1, \dots, n\}$ , “bet at Game  $i$ ”) and *keep* (“skip this round”) are offered. Each bet  $b_i$  can lead to various rewards – represented by states  $R_j$  – and the probabilities  $p_{ij}$  are known. What is notable about the (MS, CT) semantics is that a mixed scheduler  $r_{\text{keep}}\text{keep} + r_1b_1 + \dots + r_nb_n$  nicely models a gambler’s strategy to split the fund for different games (betting the  $r_1$  portion of the fund at Game 1, the  $r_2$  portion at Game 2, etc.). In contrast, the transition probabilities  $p_{ij}$  should be interpreted in the chance, not mass, manner. This is because the rewards from each game are random events and only one outcome is produced at each game  $i$  in each round (for example you either win or lose with a slot machine, instead of getting a half-and-half outcome). Another point illustrated here is that the scheduler can choose its action depending on the current mass. For example, a possible strategy is to spare half the original fund for the second round (using a strategy  $\frac{1}{2}\text{keep} + \dots$ ), and to decide how to use it depending on the outcome of the first round (i.e., the mass over the states  $R_1, \dots, R_m$  after one transition).

**Exam.** As an example suited for the (CS, MT) semantics, we present the *exam* MDP in Fig. 1. Here, one chooses a problem set  $P_i$  from a given repertoire  $P_1, \dots, P_n$ , as commonly done in language tests such as TOEFL and IELTS. Difficulties can differ among problem sets: with the problem set  $P_i$ , the  $p_{ij}$  portion of the examinees gets the grade  $j \in \{A, B, \dots, F\}$ , and this number  $p_{ij}$  is known. Exams are repeated, and different problem sets can be used at each time. The state  $R$  with its transitions models that the mass of potential examinees decays exponentially over time. Here, a mixed scheduler  $r_1P_1 + \dots + r_nP_n$  models the strategy that the problem set  $P_i$  is chosen with the probability  $r_i$ . Each time, one uses the same problem set for all examinees, so  $r_i$  should be chance-interpreted. In contrast, the transition probabilities  $p_{ij}$  should be mass-interpreted: every individual has a chance of getting a certain grade while the overall distribution over grades should follow the distribution indicated by the paper’s difficulties.

**Contributions.** In this paper, we are interested in 1) formulating a unified framework that captures all four semantics and 2) solving the *threshold reachability* problem under the new semantics we introduce. Given an MDP  $\mathcal{M}$  and a scheduler  $\sigma$ , choosing a semantics (out of the four) induces the so-called *config MC*  $\mathcal{M}_\sigma$ ; then the threshold reachability problem asks if a target set  $H$  can be reached with probability greater than a given threshold  $\xi$ . As mentioned earlier, this problem is poly-time solvable for the (CS, CT) semantics, while being Skolem hard for the (MS, MT) semantics. Hence, it is natural to ask how hard it is for the new (CS, MT) and (MS, CT) semantics, and if we can develop approaches to solve it.

Our contributions (as summarized in Table 2) are as follows.

1. A unified framework where different semantics for MDPs are systematically derived by appropriately specifying the so-called *Chance-Mass (or CM) classifier*.

■ **Table 2** Summary of semantics/results: Blue are our contributions, Grey were previously known.

		Transition	
		Chance-interpreted	Mass-interpreted
Scheduler	Chance-interpreted	( <b>CS</b> , <b>CT</b> ) semantics Ptime	( <b>CS</b> , <b>MT</b> ) semantics Undecidable, antichain algo
	Mass-interpreted	( <b>MS</b> , <b>CT</b> ) semantics #P-hard, template algo	( <b>MS</b> , <b>MT</b> ) semantics Skolem-hard

- For the (**CS**, **MT**) semantics, we show that this problem is undecidable under a global scheduler and we build an antichain-based backward reachability algorithm for finite union of finitely-generated monotone sets that is sound but necessarily incomplete.
- For the (**MS**, **CT**) semantics, we show the problem is #P-hard – the counting counterpart of NP, see e.g. [7] – even for a single action in the MDP, and we develop a template-based submartingale which provides a sound algorithm for reachability.
- We implemented both these algorithms in a prototype. This illustrates the working of our algorithms on a small set of examples including the motivating examples above.

**Related Work.** Beyond the related work already discussed, here we add some more context and mention a few others. First, the use of distributional (**MS**, **MT**) semantics for MDPs has been long studied, but mostly from a theoretical point of view. See e.g., [19]. Recently, a template-based approach is implemented in [4] for safety constraints and [5] for reach-avoidance, where the authors use Farkas Lemma [12] and Handelman’s [16] theorem to develop sound but incomplete algorithms via synthesizing ranking functions. For the (**CS**, **MT**) semantics, we use  $\gamma$ -scaled submartingales for reachability as done e.g., in [27]. Antichain algorithms have been developed for LTL synthesis e.g., in [13], but to the best of our knowledge, we are the first to use them in the context of MDPs.

## 2 Preliminaries

We let  $\mathbb{N}, \mathbb{Q}, \mathbb{R}$  denote the sets of natural numbers, rational numbers and real numbers, respectively. The unit interval is denoted by  $[0, 1]$ , and we let  $[n] = \{1, 2, \dots, n\}$ . For sets  $A$  and  $B$ ,  $2^A$  denotes the powerset of  $A$ , and  $B^A$  denotes the set of functions  $f : A \rightarrow B$ . The collection of all finite (or infinite) sequences over  $A$  is denoted by  $A^*$  (or  $A^\omega$ , respectively).  $A^+ \subseteq A^*$  is the set of nonempty finite sequences.

**Probability Distributions.** For any set  $X$ , we let  $\mathcal{D}(X) = \{d : X \rightarrow [0, 1] \mid \sum_x d(x) = 1, \text{supp}(d) \text{ is finite}\}$  denote the set of *finitely-supported discrete probability distributions* over  $X$ . Here,  $\text{supp}(d) = \{x \in X \mid d(x) > 0\}$  is the *support* of  $d$ . In this paper, we call  $d \in \mathcal{D}(X)$  simply a *distribution* over  $X$ .

An  $n$ -dimensional *stochastic matrix* is a  $n \times n$  matrix  $M \in [0, 1]^{n \times n}$  such that  $M\mathbf{1} = \mathbf{1}$ . We let  $\text{Stoc}(n)$  denote the set of  $n$ -dimensional stochastic matrices.

**Markov Decision Processes.** A *Markov decision process* (MDP)  $\mathcal{M} = (Q, \text{Act}, \delta)$  consists of a finite set  $Q$  of *states*, a finite set  $\text{Act}$  of *actions*, and a *transition function*  $\delta : Q \times \text{Act} \rightarrow \mathcal{D}(Q)$ . We write  $M_a$  for the *transition matrix* of an MDP  $\mathcal{M}$  for an action  $a$ ; concretely  $(M_a)_{ij} = \delta(q_i, a)(q_j)$ . Using this, we also write  $\mathcal{M} = (Q, \text{Act}, (M_a)_{a \in \text{Act}})$  for the same MDP.

A distribution over  $Q$ , i.e.  $d \in \mathcal{D}(Q)$ , shall be called a *configuration* of the MDP  $\mathcal{M}$ . This terminology is because of their use in Sec. 3.1.

**Antichains and Monotone Sets.** For a partially-ordered set  $(X, \leq)$ , a subset  $S \subseteq X$  is an *antichain* if for any  $x, y \in S$ ,  $x \not\leq y$  and  $y \not\leq x$ . A subset  $S \subseteq X$  is *upward closed* if for any  $x \in S$  and  $y \in X$ ,  $x \leq y$  implies  $y \in S$ . For an upward-closed set  $S \subseteq X$ , the *bottom* of  $S$  is defined by  $\lfloor S \rfloor = \{x \in S \mid \forall y \in S, y \leq x \implies x = y\}$ ; this collects all  $\leq$ -minimal elements of  $S$  and is clearly an antichain. Given an arbitrary subset  $T \subseteq X$ , its upward closure  $\uparrow T = \{x \in X \mid \exists y \in T, y \leq x\}$  is upward-closed. We say an upward-closed set  $S \subseteq X$  is *finitely generated* if  $S = \uparrow T$  for some finite set  $T$ . We do not lose generality if we restrict this  $T$  to be an antichain. In this case, we have  $\lfloor S \rfloor = T$ . The dual notions, namely of *downward-closed set* and its *top*, are defined similarly. A set is *monotone* if it is either upward-closed or downward-closed.

### 3 A Unified Framework for Mass and Chance Interpretations

Our unified framework translates an MDP  $\mathcal{M}$  and a scheduler  $\sigma$  to the so-called *configuration Markov chain (config MC)  $\mathcal{M}_\sigma$* ; the last is the semantic basis on which we can ask problems such as reachability. The transition function of the config MC  $\mathcal{M}_\sigma$  is defined systematically as the composition  $\mathbb{X} \circ \delta_\sigma$  of a *chance-mass classifier (CM classifier)  $\mathbb{X}$*  and a function  $\delta_\sigma$ ; see Def. 9. There is a notable separation of concerns here: a CM classifier  $\mathbb{X}$  specifies which semantics we use (cf. Table 2) but is independent of  $\mathcal{M}$  or  $\sigma$  (Def. 7); the function  $\delta_\sigma$  (Def. 6), in contrast, is defined by  $\mathcal{M}$  (whose transition function is  $\delta$ ) and a scheduler  $\sigma$ , but is independent of the choice of the semantics.

The following basic constructs and notations will be used throughout the construction. The notation  $\mathcal{D}g$  – inspired by category theory – may be nonstandard but is useful in Sec. 4.

► **Notation 1** (currying and pushforward). *Let  $X, Y, Z$  be sets. The currying of  $f: X \times Y \rightarrow Z$  is denoted by  $f^\wedge: X \rightarrow Z^Y$ . For  $g: X \rightarrow Y$ , the function  $\mathcal{D}g: \mathcal{D}(X) \rightarrow \mathcal{D}(Y)$  carries a (discrete) distribution  $d \in \mathcal{D}(X)$  to the pushforward measure  $(\mathcal{D}g(d))(y)$  given by  $(\mathcal{D}g(d))(y) = \sum_{x \in X, g(x)=y} d(x)$ .*

► **Notation 2** (ket notation). *For distributions, we use the following ket notation: for  $d \in \mathcal{D}(X)$ , we write  $d = \sum_x a_x |x\rangle$  with  $a_x = d(x) \in [0, 1]$ . Intuitively, if we consider  $d = \sum_x a_x |x\rangle$  as a random variable,  $a_x$  is the probability of obtaining the value  $x$ . More generally, we may use an arbitrary index set  $I$  and a function  $f: I \rightarrow X$  and write  $\sum_{i \in I} a_i |f(i)\rangle$ . This distribution assigns to each  $x \in X$  the probability  $\sum_{i \in I, f(i)=x} a_i$ .*

We note two special cases of the usages of the ket notation. Firstly, when  $I = X' \subseteq X$  is a subset of  $X$ , the distribution  $d = \sum_{x \in X'} a_x |x\rangle$  assigns 0 by case distinction:  $d(x) = a_x$  if  $x \in X'$ , and  $d(x) = 0$  otherwise. Secondly, the pushforward measure  $\mathcal{D}f(d)$ , with  $d = \sum_{x \in X} a_x |x\rangle$  and  $f: X \rightarrow Y$ , can be conveniently denoted by  $\mathcal{D}f(d) = \sum_{x \in X} a_x |f(x)\rangle$ .

#### 3.1 CM Classifiers, Pre-configurations, and Config Markov Chains

We introduce the central constructs of our framework, under two different views:

- the *concrete* and *element-level* view, describing the constructs as functions; and
- the *abstract* and *type-level* view, providing high-level intuitions of the nature of each construct using its type.

■ **Table 3** Combinatorial presentation of CM classifiers  $\mathbb{X}$ .

$\mathbb{X}(\text{CS}, \text{CT})$	$\mathbb{X}(\text{CS}, \text{MT})$	$\mathbb{X}(\text{MS}, \text{CT})$	$\mathbb{X}(\text{MS}, \text{MT})$

The latter view is in fact that of *category theory* [8, 18, 22], but its precise treatment is beyond the scope of this paper.

We highlight the role of the type-level view through a preview of one of our main constructs, namely a *CM classifier*. It is formulated as a function  $\mathbb{X}_Q: \mathcal{D}(\mathcal{D}(\mathcal{D}(Q))) \rightarrow \mathcal{D}(\mathcal{D}(Q))$ . The intuition of a CM classifier  $\mathbb{X}_Q$ , in the abstract type-level view, is as in the following preview. (The preview focuses on type-level intuitions and is not meant to be understood at the first look – it is really a *preview*. Readers can skim through it and come back.)

- In our framework for MDP semantics, there are three sources of probability distributions (namely *schedulers*, *configurations*, and *transitions*); they are represented by the three occurrences of the distribution operator  $\mathcal{D}$  in the domain of  $\mathbb{X}_Q$ .
- Each of the three kinds of probabilities are interpreted either in a *chance* or *mass* manner; these two manners are represented by the two occurrences of  $\mathcal{D}$  in the codomain of  $\mathbb{X}_Q$ .
- For intuition, we often annotate  $\mathcal{D}$ 's with their intentions, namely  $\mathcal{D}_{\text{sched}}, \mathcal{D}_{\text{conf}}, \mathcal{D}_{\text{trans}}$  (for probabilities of different sources) and  $\mathcal{D}_{\text{chan}}, \mathcal{D}_{\text{mass}}$  (for how they are interpreted). In particular, following the above intuitions, a CM classifier  $\mathbb{X}_Q$  will have the type

$$\mathbb{X}_Q: \mathcal{D}_{\text{sched}}(\mathcal{D}_{\text{conf}}(\mathcal{D}_{\text{trans}}(Q))) \longrightarrow \mathcal{D}_{\text{chan}}(\mathcal{D}_{\text{mass}}(Q)). \quad (1)$$

Note that these  $\mathcal{D}$ 's have the same mathematical content regardless of the annotations.

- The function  $\mathbb{X}_Q$  specifies which kind of probability (on the left) is interpreted in which manner (on the right). One combinatorial way of presenting such a specification  $\mathbb{X}_Q$  is as a mapping from the three  $\mathcal{D}$ 's in the domain to the two  $\mathcal{D}$ 's in the codomain.
- Furthermore, such a mapping can be described by 1) swapping the order of  $\mathcal{D}$ 's, 2) suppressing two  $\mathcal{D}$ 's into one, and 3) inserting one  $\mathcal{D}$  where there was none. See Table 3. We introduce functions  $\lambda, \mu, \eta$  for conducting these operations (Def. 11); examples of  $\mathbb{X}_Q$  are thus described as combinations of  $\lambda, \mu, \eta$ .

We move on to the precise technical development, which we mostly do in the concrete, element-level view, with some type-level intuitions supplementing it. More examples of each component introduced in this section is given in [28].

In our general framework, a configuration of an MDP is a distribution over states.

► **Definition 3** (configuration, scheduler). *A configuration of an MDP  $\mathcal{M} = (Q, \text{Act}, \delta)$  is a distribution  $d \in \mathcal{D}(Q)$ . A (configuration-based) scheduler for  $\mathcal{M}$  is a function  $\sigma: \mathcal{D}(Q)^+ \rightarrow \mathcal{D}(\text{Act})$  that maps a sequence of configurations to a distribution over actions. A scheduler is memoryless if its value only depends on the last element of its input, that is for any sequence  $d_1 d_2 \dots d_n \in \mathcal{D}(Q)^+$ ,  $\sigma(d_1 d_2 \dots d_n) = \sigma(d_n)$ . Therefore, a memoryless scheduler is a function of the type  $\sigma: \mathcal{D}(Q) \rightarrow \mathcal{D}(\text{Act})$ . A scheduler is pure if it always gives a Dirac distribution, that is for any  $p \in \mathcal{D}(Q)^+$ ,  $\sigma(p)(a) = 1$  for some  $a \in \text{Act}$ . Otherwise, the scheduler is mixed.*



► **Remark 4.** Note that the scheduler notion in Def. 3 uses *global* actions: it chooses an action  $a$  is picked from the distribution  $\sigma(d_1 d_2 \dots d_n) \in \mathcal{D}(\text{Act})$  and *this action  $a$  is used at every state  $q \in Q$* . The more common notion of *local* action – allowing different actions at different states – can be encoded as global actions by bloating the action set.

A CM classifier (as presented in (1)) takes, as input, three-fold probability distributions, and its layers correspond to the three sources of probability in MDPs. We shall call such data a pre-configuration.

► **Definition 5** (pre-configuration). *Let  $\mathcal{M} = (Q, \text{Act}, \delta)$  be an MDP. A pre-configuration of  $\mathcal{M}$  is an element  $t \in \mathcal{D}(\mathcal{D}(\mathcal{D}(Q)))$ , or  $t \in \mathcal{D}_{\text{sched}}(\mathcal{D}_{\text{conf}}(\mathcal{D}_{\text{trans}}(Q)))$  with annotations.*

The following function  $\delta_\sigma$ , as sketched in the beginning of the section, extracts a pre-configuration from an MDP  $\mathcal{M}$  and a scheduler  $\sigma$ , exposing the three sources of probabilities.

► **Definition 6** ( $\delta_\sigma$ ). *Let  $\mathcal{M} = (Q, \text{Act}, \delta)$  be an MDP and  $\sigma$  be a scheduler. We define a function  $\delta_\sigma: \mathcal{D}(Q)^+ \rightarrow \mathcal{D}(\mathcal{D}(\mathcal{D}(Q)))$  as follows: for any  $d_0 d_1 \dots d_k \in \mathcal{D}(Q)^+$ ,*

$$\delta_\sigma(d_0 d_1 \dots d_k) = \sum_{a \in \text{Act}} \sigma(d_0 d_1 \dots d_k)(a) \left| \sum_{q \in Q} d_k(q) \mid \delta(q, a) \right\rangle. \quad (2)$$

The annotations in both (1) and Def. 5 can now be explained using (2).

- The innermost distribution  $\delta(q, a)$  on the right-hand side of (2) comes from the transition function  $\delta: Q \times \text{Act} \rightarrow \mathcal{D}(Q)$  of an MDP; so we write  $\delta(q, a) \in \mathcal{D}_{\text{trans}}(Q)$ .
- The distribution  $\left| \sum_{q \in Q} d_k(q) \mid \delta(q, a) \right\rangle$ , that is one-level higher, has probabilities  $d_k(q)$  that come from a “configuration”  $d_k$  (this terminology is justified by the memoryless case of Def. 9 later). Thus we say it belongs to  $\mathcal{D}_{\text{conf}}(\mathcal{D}_{\text{trans}}(Q))$ .
- Finally, the whole right-hand side of each of (2) is a probability distribution over elements of  $\mathcal{D}_{\text{conf}}(\mathcal{D}_{\text{trans}}(Q))$ , where the probabilities (such as  $\sigma(d)(a)$ ) come from the scheduler  $\sigma$ . Therefore, the right-hand side is an element of  $\mathcal{D}_{\text{sched}}(\mathcal{D}_{\text{conf}}(\mathcal{D}_{\text{trans}}(Q)))$ .

The following defines CM classifiers, whose intuitions have been discussed earlier in Sec. 3.1.

► **Definition 7** (chance-mass (CM) classifier). *A chance-mass (CM) classifier is a family  $\mathbb{X} = (\mathbb{X}_Q)_Q$  of functions indexed by set  $Q$ ; its  $Q$ -component has type  $\mathbb{X}_Q: \mathcal{D}(\mathcal{D}(\mathcal{D}(Q))) \rightarrow \mathcal{D}(\mathcal{D}(Q))$ . When the index  $Q$  is obvious we drop it, writing  $\mathbb{X}: \mathcal{D}(\mathcal{D}(\mathcal{D}(Q))) \rightarrow \mathcal{D}(\mathcal{D}(Q))$ .*

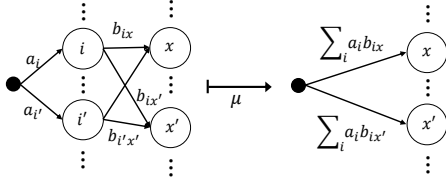
► **Remark 8.** For our theoretical development, in Def. 7, it is enough to fix an MDP  $\mathcal{M} = (Q, \text{Act}, \delta)$  and only consider the  $Q$ -component  $\mathbb{X}_Q: \mathcal{D}(\mathcal{D}(\mathcal{D}(Q))) \rightarrow \mathcal{D}(\mathcal{D}(Q))$  of  $\mathbb{X}$ . We made the above general definition – it is independent of the choice of  $Q$  – to emphasize the *uniformity* of  $\mathbb{X}$ . This uniformity should be formalized as the *naturality in  $Q$* , a central notion from category theory [22]. We will do so in our future work.

We are ready to define a config MC  $\mathcal{M}_{\mathbb{X}, \sigma}$ , a semantic basis on which we formalize e.g. reachability problems. As announced, its transition function is a composition  $\mathbb{X} \circ \delta_\sigma$  of 1) a CM classifier  $\mathbb{X}$  that chooses the type of semantics (cf. Table 1) and 2) the function  $\delta_\sigma$  that exposes three sources of probability in an MDP and forms a pre-configuration.

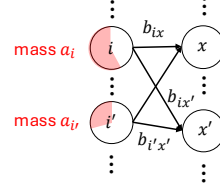
► **Definition 9** (config MC  $\mathcal{M}_{\mathbb{X}, \sigma}$ ). *Let  $\mathcal{M} = (Q, \text{Act}, \delta)$  be an MDP,  $\mathbb{X}$  be a CM classifier and  $\sigma$  be a scheduler. The configuration Markov chain (config MC)  $\mathcal{M}_{\mathbb{X}, \sigma} = (\mathcal{D}(Q)^+, \delta_{\mathbb{X}, \sigma})$  is a Markov chain carried by the set  $\mathcal{D}(Q)^+$ . Its transition function  $\delta_{\mathbb{X}, \sigma}: \mathcal{D}(Q)^+ \rightarrow \mathcal{D}(\mathcal{D}(Q)^+)$  is defined as follows: for  $d_0 d_1 \dots d_k \in \mathcal{D}(Q)^+$ ,*

$$\delta_{\mathbb{X}, \sigma}(d_0 d_1 \dots d_k) = \sum_{d_{k+1} \in \mathcal{D}(Q)} \left( \mathbb{X}(\delta_\sigma(d_0 d_1 \dots d_k))(d_{k+1}) \right) |d_0 d_1 \dots d_k d_{k+1}\rangle,$$





■ **Figure 2** The suppression operator  $\mu$ .



■ **Figure 3** The MC2CM operator  $\lambda$ , its input.

that is, a config MC state  $d_0 d_1 \dots d_k$  evolves into  $d_0 d_1 \dots d_k d_{k+1}$  with the probability  $(\mathbb{X}(\delta_\sigma(d_0 d_1 \dots d_k)))(d_{k+1})$ . Note that, in this memoryful setting, config MC states record the whole history as sequences and get extended along transitions.

For a memoryless scheduler  $\sigma: \mathcal{D}(Q) \rightarrow \mathcal{D}(\text{Act})$ , the config MC  $\mathcal{M}_{\mathbb{X}, \sigma} = (\mathcal{D}(Q), \delta_{\mathbb{X}, \sigma})$  is carried by the set  $\mathcal{D}(Q)$  of configurations (Def. 3). The function  $\delta_{\mathbb{X}, \sigma}: \mathcal{D}(Q) \rightarrow \mathcal{D}(\mathcal{D}(Q))$  is

$$\delta_{\mathbb{X}, \sigma}(d) = \sum_{d' \in \mathcal{D}(Q)} \left( \mathbb{X}(\delta_\sigma(d))(d') \right) |d'\rangle, \quad (3)$$

that is, a configuration  $d \in \mathcal{D}(Q)$  evolves into  $d' \in \mathcal{D}(Q)$  with the probability  $(\mathbb{X}(\delta_\sigma(d)))(d')$ . Note that by the definition of  $\mathcal{D}$ , the CM classifier  $\mathbb{X}(\delta_\sigma(d_0 d_1 \dots d_k))$  and  $\mathbb{X}(\delta_\sigma(d))$  are finitely-supported, hence the use of summation notation here is well-defined.

We formally state the reachability problem. More details are in [28].

► **Problem 10** (threshold reachability problem). Assume the setting of Def. 9. Let  $d_0 \in \mathcal{D}(Q)$  and  $H \subseteq \mathcal{D}(Q)$ , be an initial configuration and a target set, respectively. We let  $\text{Reach}_{\mathbb{X}, \sigma}(d_0, H)$  denote the set of paths that start from  $d_0$  and eventually reach  $H$ . The threshold reachability problem for  $\xi$  asks if there exists a scheduler  $\sigma$  such that  $\mathbb{P}_{\mathbb{X}, \sigma}(\text{Reach}_{\mathbb{X}, \sigma}(d_0, H)) \geq \xi$ .

## 4 Instances of the Unified Framework and Four Semantics

Now we instantiate the framework and derive the four semantics of MDPs in Tables 1 and 2. We first introduce the three operators  $\lambda, \mu, \eta$  for constructing the CM classifiers  $\mathbb{X}$ .

► **Definition 11** (MC2CM  $\lambda$ , suppression  $\mu$ , Dirac  $\eta$ ). Let  $X$  be a set.

■ The MC2CM operator  $\lambda_X: \mathcal{D}(\mathcal{D}(X)) \rightarrow \mathcal{D}(\mathcal{D}(X))$  on  $X$  is defined as follows:

$$\lambda_X \left( \sum_{i \in I} a_i \left| \sum_{x \in X} b_{ix} |x\rangle \right. \right) = \sum_{f \in X^I} \left( \prod_{i \in I} b_{i, f(i)} \right) \left| \sum_{i \in I} a_i |f(i)\rangle \right. \rangle. \quad (4)$$

■ The suppression operator  $\mu_X: \mathcal{D}(\mathcal{D}(X)) \rightarrow \mathcal{D}(X)$  on  $X$  is defined as follows:

$$\mu_X \left( \sum_{i \in I} a_i \left| \sum_{x \in X} b_{ix} |x\rangle \right. \right) = \sum_{x \in X} \left( \sum_{i \in I} a_i b_{ix} \right) |x\rangle. \quad (5)$$

■ The Dirac operator  $\eta_X: X \rightarrow \mathcal{D}(X)$  is defined by  $\eta_X(x) = 1 |x\rangle$ , that is,  $\eta_X(x)(x') = 1$  if  $x = x'$  and 0 otherwise.

Much like in Def. 7, we often drop the subscript  $X$  and simply write  $\lambda, \mu, \eta$ .

Here are some intuitions (see [28] for more). The suppression operator  $\mu$ , as in Fig. 2, suppresses two successive probabilistic branching into one: *tossing coins altogether, instead of one-by-one*. The Dirac operator  $\eta$  turns an element  $x$  into a trivial distribution.

The MC2CM operator  $\lambda$  is much less known. Its type with annotations is  $\lambda: \mathcal{D}_{\text{mass}}(\mathcal{D}_{\text{chan}}(X)) \rightarrow \mathcal{D}_{\text{chan}}(\mathcal{D}_{\text{mass}}(X))$ ; hence the name (“mass-chance to chance-mass”). See Fig. 3. This is how we interpret the input  $\sum_{i \in I} a_i \left| \sum_{x \in X} b_{ix} |x\rangle \right\rangle$  of  $\lambda$ ; more specifically, we are looking at the mass  $a_i$  at each  $i \in I$ , and each  $i \in I$  has the transition probability  $b_{ix}$  to each  $x \in X$ .

Here is the intention of  $\lambda$ : the destination of the mass  $a_i$  at each  $i$  is chosen by chance (coin toss), and the whole mass  $a_i$  goes to the chosen destination. This happens at each  $i \in I$ , and as a result, each  $x \in X$  will acquire the sum of the masses whose destination is  $x$ .

This whole process can be modeled as follows.

- Using functions  $f: I \rightarrow X$ , we can enumerate all combinations of the destinations  $f(i)$  of the mass at each  $i \in I$ . That is,  $f$  is the event “the destination of  $i$  is chosen as  $f(i)$ .”
- For each such event  $f \in X^I$ , the probability for  $f$  is  $\prod_{i \in I} b_{i,f(i)}$ , where  $b_{i,f(i)}$  is the probability of  $f(i)$  being chosen as the destination of  $i$ .
- Once an event  $f$  is chosen, the mass at each  $i \in I$  is moved accordingly. The resulting mass at  $x \in X$  is  $\sum_{i \in I \text{ s.t. } f(i)=x} a_i$ , and the induced distribution of masses is conveniently represented in the ket notation by  $\sum_{i \in I} a_i |f(i)\rangle$ .

This explains the right-hand side of (4): it is the distribution over the mass distributions  $\sum_{i \in I} a_i |f(i)\rangle$  (where  $f$  varies), and the probability for each  $f$  given by  $\prod_{i \in I} b_{i,f(i)}$ .

#### 4.1 Concrete Instances

Now we present, in the concrete level of elements, the four CM classifiers outlined in Table 2. We also present explicit formulas for the config MCs that are derived using them.

► **Definition 12** (CM classifiers, concrete instances). *We define four CM classifiers  $\mathbb{X}^{(\text{CS}, \text{CT})}, \mathbb{X}^{(\text{MS}, \text{CT})}, \mathbb{X}^{(\text{CS}, \text{MT})}, \mathbb{X}^{(\text{MS}, \text{MT})}: \mathcal{D}_{\text{sched}}(\mathcal{D}_{\text{conf}}(\mathcal{D}_{\text{trans}}(Q))) \rightarrow \mathcal{D}_{\text{chan}}(\mathcal{D}_{\text{mass}}(Q))$ :*

$$\begin{aligned}
\mathbb{X}^{(\text{CS}, \text{CT})} &= \mu_{\mathcal{D}(Q)} \circ \mathcal{D}\lambda_Q, \quad \text{that is concretely,} \\
\mathbb{X}^{(\text{CS}, \text{CT})} \left( \sum_{i \in I} a_i \left| \sum_{j \in J} b_{i,j} \left| \sum_{q \in Q} p_{i,j,q} |q\rangle \right\rangle \right\rangle \right) &= \sum_{i \in I} \sum_{f \in Q^J} \left( a_i \prod_{j \in J} p_{i,j,f(j)} \right) \left| \sum_{j \in J} b_{i,j} |f(j)\rangle \right\rangle; \\
\mathbb{X}^{(\text{MS}, \text{CT})} &= \mathcal{D}\mu_{(Q)} \circ \lambda_{\mathcal{D}(Q)} \circ \mathcal{D}\lambda_Q, \quad \text{that is concretely,} \\
\mathbb{X}^{(\text{MS}, \text{CT})} \left( \sum_{i \in I} a_i \left| \sum_{j \in J} b_{i,j} \left| \sum_{q \in Q} p_{i,j,q} |q\rangle \right\rangle \right\rangle \right) &= \sum_{f \in Q^{I \times J}} \prod_{(i,j) \in I \times J} p_{i,j,f(i,j)} \left| \sum_{(i,j) \in I \times J} a_i b_{i,j} |f(i,j)\rangle \right\rangle; \\
\mathbb{X}^{(\text{CS}, \text{MT})} &= \mathcal{D}\mu_{(Q)}, \quad \text{that is concretely,} \\
\mathbb{X}^{(\text{CS}, \text{MT})} \left( \sum_{i \in I} a_i \left| \sum_{j \in J} b_{i,j} \left| \sum_{q \in Q} p_{i,j,q} |q\rangle \right\rangle \right\rangle \right) &= \sum_{i \in I} a_i \left| \sum_{j \in J} \sum_{q \in Q} b_{i,j} p_{i,j,q} |q\rangle \right\rangle; \text{ and} \\
\mathbb{X}^{(\text{MS}, \text{MT})} &= \eta_{\mathcal{D}(Q)} \circ \mathcal{D}\mu_Q \circ \mu_{\mathcal{D}(Q)}, \quad \text{that is concretely,} \\
\mathbb{X}^{(\text{MS}, \text{MT})} \left( \sum_{i \in I} a_i \left| \sum_{j \in J} b_{i,j} \left| \sum_{q \in Q} p_{i,j,q} |q\rangle \right\rangle \right\rangle \right) &= \left| \sum_{i \in I} \sum_{j \in J} \sum_{q \in Q} a_i b_{i,j} p_{i,j,q} |q\rangle \right\rangle.
\end{aligned}$$

Here is the illustration of the definition of the **(CS, CT)** case (the other cases are similar). As demonstrated in Table 3, the CM classifier  $\mathbb{X}^{(\text{CS}, \text{CT})}$  first applies  $\mathcal{D}\lambda_Q$  to the pre-configuration to rearrange the order of the distribution operators and obtain an element

in  $\mathcal{D}_{\text{sched}}(\mathcal{D}_{\text{trans}}(\mathcal{D}_{\text{conf}}(Q)))$ . Since both the scheduler and the transition distributions are interpreted in the chance manner, we apply the suppression operator  $\mu_{\mathcal{D}(Q)}$  to suppress  $\mathcal{D}_{\text{sched}}$  and  $\mathcal{D}_{\text{trans}}$  to  $\mathcal{D}_{\text{chan}}$  and leave  $\mathcal{D}_{\text{conf}}$  to  $\mathcal{D}_{\text{mass}}$ .

Those concrete CM classifiers in Def. 12 instantiate the general definition of config MC (Def. 9) as follows. To avoid complex super- and subscripts, a config MC  $\mathcal{M}_{\mathbb{X}(\text{CS}, \text{CT}), \sigma}$  shall be denoted by  $\mathcal{M}_{\sigma}^{(\text{CS}, \text{CT})}$ , and similarly for other semantics. Here we focus on memoryless schedulers for readability (cf. (3)); the general case is presented in [28].

► **Definition 13** (config MC under concrete semantics, memoryless). *Let  $\mathcal{M} = (Q, \text{Act}, \delta)$  be an MDP, and  $\sigma: \mathcal{D}(Q) \rightarrow \mathcal{D}(\text{Act})$  be a (memoryless) scheduler. The four config MCs induced by  $\mathcal{M}$  and  $\sigma$ , under the four CM classifiers in Def. 12, are concretely described as follows.*

- Those config MCs are denoted by  $\mathcal{M}_{\sigma}^{(\text{CS}, \text{CT})}, \mathcal{M}_{\sigma}^{(\text{MS}, \text{CT})}, \mathcal{M}_{\sigma}^{(\text{CS}, \text{MT})}, \mathcal{M}_{\sigma}^{(\text{MS}, \text{MT})}$ .
- They all have  $\mathcal{D}(Q)$  as a state space.
- Their transition functions, all of type  $\mathcal{D}(Q) \rightarrow \mathcal{D}(\mathcal{D}(Q))$ , are given as follows.

$$\begin{aligned}
 \delta_{\mathbb{X}(\text{CS}, \text{CT}), \sigma}(d)(d') &= \sum_{a \in \text{Act}} \sum_{f \in Q^Q \text{ s.t. } \mathcal{D}f(d)=d'} \sigma(d)(a) \cdot \left( \prod_{q \in Q} \delta(q, a)(f(q)) \right), \\
 \delta_{\mathbb{X}(\text{MS}, \text{CT}), \sigma}(d)(d') &= \sum_{f \in Q^Q \times \text{Act} \text{ s.t. } d' = \mathcal{D}f(d \otimes \sigma(d))} \prod_{(q, a) \in Q \times \text{Act}} \delta(q, a)(f(q, a)), \\
 \delta_{\mathbb{X}(\text{CS}, \text{MT}), \sigma}(d)(d') &= \sum_{a \in \text{Act} \text{ s.t. } d' = \mu_Q(\mathcal{D}(\delta^{\wedge})(a))(d)} \sigma(d)(a), \\
 \delta_{\mathbb{X}(\text{MS}, \text{MT}), \sigma}(d)(d') &= \begin{cases} 1 & \text{if } d' = \sum_{a \in \text{Act}} \sum_{q' \in Q} \sum_{q \in Q} \sigma(d)(a) d(q) \delta(q, a)(q') |q'\rangle, \\ 0 & \text{otherwise.} \end{cases}
 \end{aligned} \tag{6}$$

In the second line,  $d \otimes \sigma(d) \in \mathcal{D}(Q \times \text{Act})$  denotes the product, which is a coupling of the configuration  $d \in \mathcal{D}(Q)$  and the distribution  $\sigma(d) \in \mathcal{D}(\text{Act})$  over actions, that is  $(d \otimes \sigma(d))(q, a) = d(q) \cdot \sigma(d)(a)$ .

► **Remark 14.** As we discussed in Sec. 1, the **(CS, CT)** semantics should coincide with the conventional semantics of MDPs, but Def. 13 may not look that way. We can show that 1) the transition function  $\delta_{\mathbb{X}(\text{CS}, \text{CT}), \sigma}$  preserves *Dirac-ness* (i.e. if  $d$  is Dirac, then  $\delta_{\mathbb{X}(\text{CS}, \text{CT}), \sigma}(d)(d') > 0$  implies  $d'$  is Dirac as well), and 2) the definition in Def. 13 indeed is the usual semantics when  $d$  and  $d'$  are Dirac (meaning that these configurations are single states with mass 1). A proof is in [28].

► **Remark 15** (combinations other than the four). We have presented four instantiations of the CM classifier framework, corresponding Table 2. Combinatorially, there are  $2^3 = 8$  possible ways of assigning  $\mathcal{D}_{\text{sched}}$ ,  $\mathcal{D}_{\text{conf}}$  and  $\mathcal{D}_{\text{trans}}$  to  $\mathcal{D}_{\text{chan}}$  and  $\mathcal{D}_{\text{mass}}$ , but we have only four since we always assign  $\mathcal{D}_{\text{conf}}$  to  $\mathcal{D}_{\text{mass}}$ . The other options (where  $\mathcal{D}_{\text{conf}}$  is assigned to  $\mathcal{D}_{\text{chan}}$ ) are not considered as they do not conceptually fit the general framework in Sec. 3, where a “state” of a config MC induced by an MDP should be a distribution over the state space of the MDP.

## 5 Algorithms and Complexity

Now we present several complexity results and two algorithms for Problem 10, with  $\mathbb{X}^{(\text{CS}, \text{MT})}$  and  $\mathbb{X}^{(\text{MS}, \text{CT})}$  as CM classifiers (the other two have been studied before). Due to space constraints, all proofs are in Appendix of [28]. For completeness, we start by recalling known results when  $\mathbb{X}^{(\text{CS}, \text{CT})}$  and  $\mathbb{X}^{(\text{MS}, \text{MT})}$  are provided as the input for the CM classifier. In this section, we refer to Problem 10 as the *threshold S-reachability problem* with the CM classifier provided as  $\mathbb{X}^{\mathbf{S}}$ , where  $\mathbf{S}$  can be **(CS, CT)**, **(MS, CT)**, **(CS, MT)** or **(MS, MT)**.

**Known results on the (CS, CT) and the (MS, MT) semantics.** The (CS, CT) semantics models the conventional MDP semantics (Rem. 14). This implies that the threshold (CS, CT)-reachability problem is in Ptime when the initial configuration  $d_0$  is Dirac. The proof is a simple adaption of those for the conventional semantics and can be found in [9].

The (MS, MT) semantics is about distribution transformers in [1, 2, 4–6, 10, 11, 19, 21]. There, the threshold reachability is shown to be positivity-hard (see [3]), implying that the threshold (MS, MT)-reachability is positivity-hard. (The result in [3] uses local actions but it is easily translated to our formalization with global actions. See Rem. 4.)

### 5.1 Complexity and Algorithm for the (MS, CT) Semantics

We first a lower bound for the threshold (MS, CT)-reachability. It is proven by a reduction from the counting variant of the subset sum problem, which is known to be  $\sharp$ P-complete [25].

► **Theorem 16.** *The threshold (MS, CT)-reachability problem is  $\sharp$ P-hard.*

We show, however, that given an explicit scheduler, the problem becomes positivity-hard. Unlike in conventional (CS, CT) or distributional (MS, MT) semantics, here fixing a scheduler does not ease the problem. Thm. 18 is proved by reduction from the reachability problem for Markov chains under the (MS, MT) semantics (shown positivity-hard in [3]).

► **Problem 17** (threshold (MS, CT)-reachability with explicit scheduler). *Given an MDP  $\mathcal{M} = (Q, \text{Act}, \delta)$ , an initial configuration  $d_0 \in \mathcal{D}(Q)$ , a scheduler  $\sigma$ , a set of configuration  $H \subseteq \mathcal{D}(Q)$  and a threshold  $\xi \in [0, 1]$ , decide if  $\mathbb{P}_{\mathbb{X}(\text{MS}, \text{CT}), \sigma}(\text{Reach}_{\mathbb{X}(\text{MS}, \text{CT}), \sigma}(d_0, H)) \geq \xi$ .*

► **Theorem 18** (positivity-hardness when scheduler given). *Problem 17 is Positivity-hard.*

► **Remark 19.** The proof of the above results are presented in [28]. Both of the proof does not require the scheduler to be memoryful, hence the hardness results holds even when the problem is restricted to memoryless schedulers.

### A Template-based Synthesis Algorithm for Finitely-Generated Monotone Set

Now we present a template-based synthesis algorithm for solving the threshold (MS, CT)-reachability problem, under the assumption that the target set is provided as a finitely-generated monotone set.

We saw that the threshold reachability problem is hard (Thms 16 and 18) – this is because MDPs under the (MS, CT) semantics can branch into exponentially many configurations which are expensive to track. Therefore, instead of tracking reachable config's, we directly search for a reachability certificate by template-based synthesis and constraint solving.

Our result is an extension of the template-based algorithm in [5] with  $\gamma$ -scaled submartingales introduced in [27]. We also make several changes on the assumption of the target set, namely using upward- or downward-closed sets, which involves some technical novelties. We present a version with upward-closed sets; dealing with downward-closed sets is similar.

We start by defining the  $\gamma$ -scaled configuration submartingale, adapted from [27].

► **Definition 20** ( $\gamma$ -scaled configuration submartingale). *Given an MDP  $\mathcal{M}$ , a memoryless scheduler  $\sigma$ , a target set  $H$  and a real number  $\gamma \in (0, 1)$ , a  $\gamma$ -scaled configuration submartingale is a function  $R : \mathcal{D}(Q) \rightarrow [0, 1]$  such that the following holds: for each  $d \in \mathcal{D}(Q) \setminus H$ ,*

$$R(d) \leq \gamma \cdot \sum_{d' \in \mathcal{D}(Q)} R(d') \cdot \delta_{\mathbb{X}(\text{MS}, \text{CT}), \sigma}(d)(d'). \quad (7)$$

Note that, in (7), the term  $\delta_{\mathbb{X}(\text{MS}, \text{CT}), \sigma}(d)$  is of type  $\mathcal{D}(\mathcal{D}(Q))$  which gives a distribution over successor configurations  $d'$  in the config MC under the scheduler  $\sigma$ . Hence, evaluating it with  $d'$  gives the probability of reaching  $d'$  from  $d$ . The sum in (7) can be understood as the expected value of the submartingale  $R$  after one-step under the scheduler  $\sigma$ . Note that the sum is well-defined since all distributions in this paper are countably supported.

The next theorem is proved much like in [27].

► **Theorem 21** ( $\gamma$ -scaled submartingale witness). *Assume that the input of Problem 10 is given, with  $\mathbb{X}(\text{MS}, \text{CT})$  as the CM classifier. Let  $R$  be a  $\gamma$ -scaled submartingale  $R$  for some  $\gamma \in (0, 1)$ . Then  $R(d_0) \geq \xi$  implies there exists a memoryless scheduler  $\sigma$  such that  $\mathbb{P}(\text{Reach}_{\mathbb{X}(\text{MS}, \text{CT}), \sigma}(d_0, H)) \geq \xi$ . If  $R(d_0) \geq \xi$  holds, we say that  $R$  is a witness of reachability.*

As usual in template-based synthesis, our algorithm collects constraints – ours are given as a system of inequalities over reals – and solves them. The construction is three steps.

- In the first step, templates for a scheduler  $\sigma$  and a  $\gamma$ -scaled configuration submartingale  $R$  are constructed.
- In the second step, constraints are constructed to ensure that a feasible solution of the constraints corresponds to a valid scheduler and a valid  $\gamma$ -scaled submartingale that is a witness of reachability.
- In the final step, the whole constraints are passed to an off-the-shelf polynomial inequality solver to compute a feasible solution (we use Z3). If a feasible solution is found, the algorithm concludes the target set is reachable and return the submartingale as a witness.

Below we briefly describe each step.

**Step 1: Setup of Template.** The first step is to set up templates for a scheduler and a  $\gamma$ -scaled submartingale. This step is standard and done similarly to other template-based synthesis algorithms; thus we only outline the templates here. For details, see [5] and the references therein.

- **Scheduler:** two sets of template variables  $\{\theta_q^a \mid a \in \text{Act}, q \in Q \cup \{0\}\}$  and  $\{s_q \mid q \in Q \cup \{0\}\}$  are fixed, such that the target scheduler  $\sigma$  is as follows: for each configuration  $d \in \mathcal{D}(Q)$  and action  $a \in \text{Act}$ ,  $\sigma(d)(a) = \frac{\theta_0^a + \sum_{q \in Q} \theta_q^a \cdot d(q)}{s_0 + \sum_{q \in Q} s_q \cdot d(q)}$ .
- **Submartingale:** a set of template variables  $\{r_q \mid q \in Q \cup \{0\}\}$  is fixed so that the target submartingale  $R: \mathcal{D}(Q) \rightarrow [0, 1]$  is  $R(d) = r_0 + \sum_{q \in Q} r_q \cdot d(q)$ .

**Step 2: Constraint Collection.** We now describe how the constraints for the templates are constructed to enforce the validity and witness property of the templates.

**Constraints for Schedulers.** To ensure that for any configuration  $d \in \mathcal{D}(Q)$ ,  $\sigma(d)$  is a distribution over actions, we require it to satisfy 1) for any action  $a \in \text{Act}$ ,  $\sigma(d)(a) \geq 0$  and 2)  $\sum_{a \in \text{Act}} \sigma(d)(a) = 1$ . Concretely, using the template variables,

$$\Phi_{\text{Schedule}} : \begin{cases} d \in \mathcal{D}(Q) & \implies \bigwedge_{a \in \text{Act}} \left[ \theta_0^a + \sum_{q \in Q} \theta_q^a \cdot d(q) \geq 0 \right] \\ d \in \mathcal{D}(Q) & \implies s_0 + \sum_{q \in Q} s_q \cdot d(q) \geq 1 \\ d \in \mathcal{D}(Q) & \implies \sum_{a \in \text{Act}} \theta_0^a + \sum_{q \in Q} \theta_q^a \cdot d(q) = s_0 + \sum_{q \in Q} s_q \cdot d(q). \end{cases} \quad (8)$$

**Constraints for Submartingales.** For submartingales, we have two constraints: 1) for any configuration  $d \in \mathcal{D}(Q)$ ,  $R(d) \in [0, 1]$  and 2) for any configuration  $d \in \mathcal{D}(Q) \setminus H$ ,  $R(d)$  satisfies the submartingale condition in (7). Concretely, using the template variables,

$$\Phi_{\text{Bound}} : d \in \mathcal{D}(Q) \implies 0 \leq r_0 + \sum_{q \in Q} r_q \cdot d(q) \leq 1, \quad (9)$$

$$\Phi_{\text{Inductive}} : d \in \mathcal{D}(Q) \setminus H \implies r_0 + \sum_{q \in Q} r_q \cdot d(q) \leq \gamma \sum_{a \in \text{Act}} \sigma(q)(a) R(M_a^T d). \quad (10)$$

**Constraints for Reachability Certificates.** We also impose a constraint so that  $R$  is a witness of reachability, i.e.  $R(d_0) \geq \xi$ . Concretely,

$$\Phi_{\text{Reachable}} : R(d_0) = r_0 + \sum_{q \in Q} r_q d_0(q) \geq \xi. \quad (11)$$

**Step 2.5: Quantifier Elimination.** It is worth noting that all the constraints we obtained in the system are either a simple inequality or a quantified formula in the form of  $\forall \mathbf{x} \in \mathbb{R}^n, \Phi(\mathbf{x}) \geq 0 \implies \psi(\mathbf{x}) \geq 0$ , where  $\Phi$  is a linear function and  $\psi$  is a polynomial. Therefore, the well-known Farkas's lemma [12] and Handelman's theorem [16] can be applied to perform quantifier elimination, which reduce the problem to solving a system of polynomial inequalities over reals. Further details are presented in [28].

**Step 3: Constraint Solving.** Once all constraints are translated to an existential problem over reals, the system is then passed to a polynomial inequality solver for finding a feasible solution. If a feasible solution is found, then we claim that the target set  $H$  is reachable.

► **Theorem 22.** *The algorithm is sound, with a runtime in 2EXPTIME.*

The soundness proof follows directly from Thm. 21. For the time complexity, it takes exponential time (in the number of generators of  $H$ ) to construct all the constraints, and since existential theory of reals is in PSPACE, hence we obtain 2EXPTIME in total.

## 5.2 Complexity and Algorithm for the (CS, MT) Semantics

**Complexity Results.** In the (CS, MT) semantics, similarly to the conventional (CS, CT) semantics, one can replace a mixed scheduler with a pure scheduler without decreasing the reachability probability. Intuitively, this is achieved by inductively choosing the action with the largest probability of reaching the target set.

► **Lemma 23.** *Assume the setting of Problem 10. If there is a mixed scheduler  $\sigma: \mathcal{D}(Q)^+ \rightarrow \mathcal{D}(\text{Act})$  such that  $\mathbb{P}(\text{Reach}_{\mathbb{X}(\text{CS}, \text{MT}), \sigma}(d_0, H)) \geq \xi$ , then there is a pure scheduler  $\sigma': \mathcal{D}(Q)^+ \rightarrow \text{Act}$  (interpreted as a scheduler that only consists of Dirac distributions over Act in our framework), such that  $\mathbb{P}(\text{Reach}_{\mathbb{X}(\text{CS}, \text{MT}), \sigma'}(d_0, H)) \geq \xi$ .*

Therefore we can limit our search to pure (instead of mixed) schedulers. Furthermore we notice the following: in the concrete description of (6), if  $\sigma$  is a pure scheduler, the probability amplitude  $\sigma(d)(a)$  is either 0 or 1. All these reduce, in the case of the (CS, MT) semantics, the original problem (Problem 10) to the following *deterministic* problem.

► **Problem 24 (deterministic (CS, MT)-reachability).** *In the setting of Problem 10, for a finite sequence  $\bar{a} = a_1 a_2 \dots a_n \in \text{Act}^*$  of actions, let  $d_{\bar{a}, 1}, \dots, d_{\bar{a}, n} \in \mathcal{D}(Q)^*$  be a sequence of configurations and  $\sigma_{\bar{a}}$  be a pure scheduler such that for any  $j \in [n]$ , the following is satisfied:  $\sigma_{\bar{a}}(d_0 d_{\bar{a}, 1} \dots d_{\bar{a}, j-1})(a_j) = 1$  and  $\delta_{\mathbb{X}(\text{CS}, \text{MT}), \sigma_{\bar{a}}}(d_0 d_{\bar{a}, 1} \dots d_{\bar{a}, j-1})(d_{\bar{a}, j}) = 1$ . Note that the scheduler is potentially memoryful.*

*The deterministic (CS, MT)-reachability problem asks if there exists a finite sequence  $\bar{a} = a_1 a_2 \dots a_n \in \text{Act}^*$  of actions and a natural number  $n \in \mathbb{N}$  such that  $d_{\bar{a}, n} \in H$ .*



Now the following result is proved by reduction from the emptiness problem of probabilistic finite automaton (PFA). The latter is undecidable; see e.g. [14]. Further, we also show that the problem remains undecidable even when the target set  $H$  is finitely generated and monotone. Both results only hold for searching a memoryful scheduler.

► **Theorem 25** (undecidability of (CS,MT)-reachability). *The threshold (CS,MT)-reachability is undecidable even when the target set  $H$  is restricted to a finitely-generated monotone set.*

### An Antichain-based Algorithm for Finitely-generated Monotone Set

Despite the complexity, here we attempt to provide an algorithm for solving Problem 24 with a target set  $H$  being finitely generated and upward-closed (similarly to Sec. 5.1, the adaption to downward-closed sets is straightforward). We motivate the use of antichain – used e.g. in [13] – by a simple observation that all stochastic matrices are monotone with respect to the element-wise order. That is, let  $M \in \text{Stoc}(n)$  be a stochastic matrix, then for any  $\mathbf{x}, \mathbf{y} \in [0, 1]^n$ ,  $\mathbf{x} \preceq \mathbf{y}$  implies  $M^T \mathbf{x} \preceq M^T \mathbf{y}$ . We come to the following simple lemma.

► **Lemma 26** (pullback of upward-closed set). *Let  $H$  be an upward-closed set and  $M \in \text{Stoc}(n)$  be a stochastic matrix. Then the pullback of  $H$  with respect to  $M$ , denoted as  $M^\sharp H$  and defined as  $M^\sharp H = \{\mathbf{x} \in [0, 1]^n \mid M^T \mathbf{x} \in H\}$ , is upward closed.*

Hence inductively, the set of all initial configurations that are reachable to  $H$  is also upward-closed, which motivates the usage of antichains since they are a canonical representation of upward-closed sets. Formally, we connect this observation to Problem 24 with the following theorem, which characterizes the set of configurations reaching  $H$  as the least fixed point of an operator that preserves upward-closedness.

► **Theorem 27** (reachable configuration as a least fixed point). *Given an MDP  $\mathcal{M} = (Q, \text{Act}, \text{Mat})$ , an initial configuration  $d_0 \in \mathcal{D}(Q)$  and a finitely-generated upward-closed set  $H$ , let  $B : 2^{[0,1]^Q} \rightarrow 2^{[0,1]^Q}$  be a function defined by  $B(S) = H \cup \bigcup_{M \in \text{Mat}} M^\sharp S$ . Then the sequence  $(B^j(\perp))_{j \in \mathbb{N}}$  converges to its least fixed point  $\mu B$  and  $H$  is reachable from  $d_0$  if and only if  $d_0 \in \mu B$ .*

The (CS,MT)-reachability problem is undecidable in general (Thm. 25); thus computing the least fixed point of  $B$  is not always possible. Nevertheless, we can approximate it and provide a sound but necessarily incomplete algorithm. Since  $\perp$  is by definition upward closed, it follows that all approximants  $B^k(\perp)$  are upward closed. Hence algorithmically, it suffices for us to keep track of the minimal elements of each  $B^k(\perp)$ , which can be maintained efficiently through the antichain data structure.

The only challenge that remains is to compute the bottom (the subset of  $\preceq$ -minimal elements) of an upward-closed set. Specifically, in the current setting, we need to compute  $\lfloor M^\sharp S \rfloor = \lfloor \{\mathbf{y} \in [0, 1]^Q \mid M^T \mathbf{y} \succeq \mathbf{x} \text{ for some } \mathbf{x} \in S\} \rfloor$  for  $M \in \text{Mat}$ . Since we assume an antichain structure is always maintained in the algorithm, this problem can be reduced to computing the set  $\lfloor \{\mathbf{y} \in [0, 1]^Q \mid M^T \mathbf{y} \succeq \mathbf{x}\} \rfloor$  for each  $\mathbf{x} \in \lfloor S \rfloor$  and merging them.

Our algorithm solves the last problem (for each  $\mathbf{x} \in \lfloor S \rfloor$ ) by iteratively solving the following sequence of minimization problems, increasing  $k$  over time: we randomly sample  $(\mathbf{q}_j)_{j \in [k-1]} \in [Q]^{k-1}$  (recall the notation  $[n]$  from Sec. 2), and solve

$$\begin{aligned} \mathbf{y}_k^* = \underset{\mathbf{y}}{\operatorname{argmin}} \mathbf{1}^T \mathbf{y} \quad \text{subject to} \quad & M^T \mathbf{y} \succeq \mathbf{x}, \quad \mathbf{1}^T \mathbf{y} \leq 1, \\ & \mathbf{y} \succeq \mathbf{0}, \quad \text{and} \quad \forall j < k, (\mathbf{y})_{\mathbf{q}_j} < (\mathbf{y}_j^*)_{\mathbf{q}_j}. \end{aligned} \tag{12}$$



The intuition is as follows. Our goal is to iteratively obtain elements  $\mathbf{y}_1^*, \mathbf{y}_2^*, \dots$  of the set  $S = \{\mathbf{y} \in [0, 1]^{|Q|} \mid M^T \mathbf{y} \succeq \mathbf{x}\}$ . For the first element  $\mathbf{y}_1^*$ , the last constraint in (12) is vacuous, and we obtain an element of  $S$  that is minimal wrt.  $\preceq$  (here it is important that  $\mathbf{y} \preceq \mathbf{y}'$  implies  $\mathbf{1}^T \mathbf{y} \leq \mathbf{1}^T \mathbf{y}'$ ). For the step case with  $k > 1$ , the last constraint in (12) enforces that  $\mathbf{y}_j^* \not\preceq \mathbf{y}_k^*$  for each  $j \in [k-1]$ , where  $\mathbf{q}_j \in |Q|$  is the “guessed” index that witnesses  $\mathbf{y}_j^* \not\preceq \mathbf{y}_k^*$ . That  $\mathbf{y}_k^* \preceq \mathbf{y}_j^*$  is prohibited, too, for each  $j < k$ , since otherwise  $\mathbf{y}_k^*$  must have been obtained in the  $j$ -th round instead of  $\mathbf{y}_j^*$ .

This problem is an instance of linear programming which can be solved by any off-the-shelf LP solver in PTime. If the solver does not return a solution, another  $\mathbf{q}$  will be sampled and used to solve for  $\mathbf{y}_k^*$ . The process terminates when either  $K$  solutions  $\mathbf{y}_1^*, \dots, \mathbf{y}_K^*$  are found, or the solver fails to find  $\mathbf{y}_k^*$  for some  $k$  after  $L$  different samples of  $(\mathbf{q}_j)_j$ 's ( $K$  and  $L$  are hyperparameters). The set of solutions  $\{\mathbf{y}_k^* \mid k \in [K]\}$  is then used to approximate the set  $\{\mathbf{y} \in [0, 1]^{|Q|} \mid M^T \mathbf{y} \succeq \mathbf{x}\}$ .

► **Lemma 28.** *Given a sub-distribution  $\mathbf{x} \in [0, 1]^n$ , a stochastic matrix  $M \in \text{Stoc}(n)$ , and parameters  $K, L \in \mathbb{N}$ , let  $\mathcal{Y}_{K,L}(\mathbf{x})$  denotes the set of solutions obtained by the above described procedure labelled by parameters  $K$  and  $L$ , then  $\mathcal{Y}_{K,L}(\mathbf{x}) \subseteq \{\mathbf{y} \in [0, 1]^{|Q|} \mid M^T \mathbf{y} \succeq \mathbf{x}\}$ . Moreover, if the set  $\{\mathbf{y} \in [0, 1]^{|Q|} \mid M \mathbf{y} \succeq \mathbf{x}\}$  is finitely generated, then we have  $\mathcal{Y}_{K,L}(\mathbf{x}) = \{\mathbf{y} \in [0, 1]^{|Q|} \mid M \mathbf{y} \succeq \mathbf{x}\}$  almost surely for large enough  $K$  and  $L$ .*

■ **Algorithm 1** Backward projection algorithm.

---

**Input:** an MDP  $\mathcal{M} = (Q, \text{Act}, \text{Mat})$ , an initial configuration  $d_0 \in \mathcal{D}(Q)$ , a target set antichain  $\lfloor H \rfloor \subset [0, 1]^{|Q|}$ , parameters  $K, L \in \mathbb{N}$

**Output:** True if  $d_0$  is reachable to  $H$

```

1  $S \leftarrow \{\}$ 
2  $S' \leftarrow \lfloor H \rfloor$ 
3 do
4    $S \leftarrow S'$ 
5    $S' \leftarrow \lfloor H \rfloor$ 
6   for  $M \in \text{Mat}$  do
7     for  $\mathbf{x} \in S$  do
8        $\{\mathbf{y}_k^* \mid k \in [K]\} \leftarrow$  Solve the problem in (12) against  $M, \mathbf{x}$  with parameters  $K, L$ 
9        $S' \leftarrow \lfloor S' \cup \{\mathbf{y}_k^* \mid k \in [K]\} \rfloor$ 
10  if  $d_0 \in S'$  then return True
11 while  $S \neq S'$ 
12 if  $d_0 \in S'$  then return True

```

---

Finally, we formalize the algorithm and its correctness. The algorithm takes as input an MDP  $\mathcal{M} = (Q, \text{Act}, \text{Mat})$ , an initial configuration  $d_0 \in \mathcal{D}(Q)$ , a target set  $H$  provided as  $\lfloor H \rfloor$  and two parameters  $K, L \in \mathbb{N}$ , and returns True if  $d_0$  is reachable to  $H$ . It starts with a set  $S = \lfloor H \rfloor$  and iteratively updates  $S$  by computing  $\lfloor B(S) \rfloor$  through solving the optimization problem described above. The algorithm only terminates when it witnesses a reachability or the set of configurations reachable to  $H$  stabilizes, that is,  $d_0 \in S$  or  $S = S'$ . The full pseudocode is provided in Algorithm 1. The termination is necessarily not guaranteed since the problem is undecidable (Thm. 25). But from Lem. 28 we can conclude that the algorithm always computes an under-approximation of the set of configurations reaching  $H$ , hence as a direct consequence of Thm. 27, we have the following theorem.

► **Theorem 29.** *Algorithm 1 is a sound algorithm for solving the (CS, MT)-reachability problem with finitely-generated monotone target set  $H$ .*

## 6 Prototype Implementation and Experiments

While the primary focus of this paper is the theoretical development of the unifying framework and algorithms in the two new semantics, we also implemented prototypes of the algorithms in Sections 5.1 and 5.2, as a proof of concept. Details of the experiment setup and benchmarks are presented in [28]. We tested the implementations against various models from [4, 10] and also our motivating examples *Casino* and *Exam* from Sec. 1.

Despite the hardness of the problem, the antichain algorithm in Sec. 5.2 on average solves a problem with three to five states within a few seconds. Most runs terminate before reaching a loop limit and prove the reachability. The template-based algorithm in Sec. 5.1 on average solves a problem with 3 to 5 states within the range of a few seconds to a hundred seconds. We were able to find a certificate that witnesses reachability in all our examples by considering only polynomials up to degree 4. Complete results are provided in [28]. In summary, our implementation indicates that the algorithms work on small models, at least, and have the potential to be further developed for real-world applications.

## 7 Conclusion

In this paper, we presented a unified framework for defining semantics for MDPs that distinguishes randomness arising from schedulers from randomness in transitions. By fixing which get interpreted distributionally (*mass*) and which probabilistically (*chance*), we obtain four semantics of which two are new. We proved hardness results for the reachability problem with monotone target states for the two new semantics and developed sound algorithms using completely different techniques. As future work, it would be interesting to use the framework defined here, e.g., for obtaining the semantics for *quantum* MCs and MDPs that have been recently considered in the verification community [15, 29].

---

## References

- 1 Rajab Aghamov, Christel Baier, Toghrul Karimov, Joris Nieuwveld, Joël Ouaknine, Jakob Piribauer, and Mihir Vahanwala. Model checking markov chains as distribution transformers. In Nils Jansen, Sebastian Junges, Benjamin Lucien Kaminski, Christoph Matheja, Thomas Noll, Tim Quatmann, Mariëlle Stoelinga, and Matthias Volk, editors, *Principles of Verification: Cycling the Probabilistic Landscape - Essays Dedicated to Joost-Pieter Katoen on the Occasion of His 60th Birthday, Part II*, volume 15261 of *Lecture Notes in Computer Science*, pages 293–313. Springer, 2024. doi:10.1007/978-3-031-75775-4\_13.
- 2 Manindra Agrawal, S. Akshay, Blaise Genest, and P. S. Thiagarajan. Approximate verification of the symbolic dynamics of markov chains. *J. ACM*, 62(1):2:1–2:34, 2015. doi:10.1145/2629417.
- 3 S. Akshay, Timos Antonopoulos, Joël Ouaknine, and James Worrell. Reachability problems for markov chains. *Information Processing Letters*, 115(2):155–158, 2015. doi:10.1016/j.ipl.2014.08.013.
- 4 S. Akshay, Krishnendu Chatterjee, Tobias Meggendorfer, and Đorđe Žikelić. Mdps as distribution transformers: Affine invariant synthesis for safety objectives. In *Computer Aided Verification: 35th International Conference, CAV 2023, Paris, France, July 17–22, 2023, Proceedings, Part III*, pages 86–112, Berlin, Heidelberg, 2023. Springer-Verlag. doi:10.1007/978-3-031-37709-9\_5.
- 5 S. Akshay, Krishnendu Chatterjee, Tobias Meggendorfer, and Đorđe Žikelić. Certified policy verification and synthesis for mdps under distributional reach-avoidance properties. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI '24*, 2024. doi:10.24963/ijcai.2024/1.

- 6 S. Akshay, Blaise Genest, and Nikhil Vyas. Distribution-based objectives for markov decision processes. In *LICS*, pages 36–45. ACM, 2018. doi:10.1145/3209108.3209185.
- 7 Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009. URL: <http://www.cambridge.org/catalogue/catalogue.asp?isbn=9780521424264>.
- 8 Steve Awodey. *Category Theory*. Oxford Logic Guides. Oxford Univ. Press, 2006.
- 9 Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT press, 2008.
- 10 Rohit Chadha, Vijay Anand Korthikanti, Mahesh Viswanathan, Gul Agha, and YoungMin Kwon. Model checking mdps with a unique compact invariant set of distributions. In *Eighth International Conference on Quantitative Evaluation of Systems, QEST 2011, Aachen, Germany, 5-8 September, 2011*, pages 121–130. IEEE Computer Society, 2011. doi:10.1109/QEST.2011.22.
- 11 Laurent Doyen, Thierry Massart, and Mahsa Shirmohammadi. Limit synchronization in markov decision processes. In *FoSSaCS*, volume 8412 of *Lecture Notes in Computer Science*, pages 58–72. Springer, 2014. doi:10.1007/978-3-642-54830-7\_4.
- 12 Julius Farkas. Theorie der einfachen ungleichungen. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 1902(124):1–27, 1902.
- 13 Emmanuel Filiot, Naiyong Jin, and Jean-François Raskin. Antichains and compositional algorithms for ltl synthesis. *Form. Methods Syst. Des.*, 39(3):261–296, December 2011. doi:10.1007/s10703-011-0115-3.
- 14 Hugo Gimbert and Youssouf Oualhadj. Probabilistic automata on finite words: Decidable and undecidable problems. In Samson Abramsky, Cyril Gavaille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part II*, volume 6199 of *Lecture Notes in Computer Science*, pages 527–538. Springer, 2010. doi:10.1007/978-3-642-14162-1\_44.
- 15 Ji Guan, Yuan Feng, Andrea Turrini, and Mingsheng Ying. Measurement-based verification of quantum markov chains. In Arie Gurfinkel and Vijay Ganesh, editors, *Computer Aided Verification - 36th International Conference, CAV 2024, Montreal, QC, Canada, July 24-27, 2024, Proceedings, Part III*, volume 14683 of *Lecture Notes in Computer Science*, pages 533–554. Springer, 2024. doi:10.1007/978-3-031-65633-0\_24.
- 16 David Handelman. Representing polynomials by positive linear functions on compact convex polyhedra. *Pacific Journal of Mathematics*, 132(1):35–62, 1988.
- 17 Christian Hensel, Sebastian Junges, Joost-Pieter Katoen, Tim Quatmann, and Matthias Volk. The probabilistic model checker storm. *Int. J. Softw. Tools Technol. Transf.*, 24(4):589–610, 2022. doi:10.1007/S10009-021-00633-Z.
- 18 Bart Jacobs. *Introduction to Coalgebra: Towards Mathematics of States and Observation*, volume 59 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2016. doi:10.1017/CB09781316823187.
- 19 Vijay Anand Korthikanti, Mahesh Viswanathan, Gul Agha, and YoungMin Kwon. Reasoning about mdps as transformers of probability distributions. In *QEST*, pages 199–208. IEEE Computer Society, 2010. doi:10.1109/QEST.2010.35.
- 20 Marta Z. Kwiatkowska, Gethin Norman, and David Parker. The PRISM benchmark suite. In *Ninth International Conference on Quantitative Evaluation of Systems, QEST 2012, London, United Kingdom, September 17-20, 2012*, pages 203–204. IEEE Computer Society, 2012. doi:10.1109/QEST.2012.14.
- 21 YoungMin Kwon and Gul A. Agha. Verifying the evolution of probability distributions governed by a DTMC. *IEEE Trans. Software Eng.*, 37(1):126–141, 2011. doi:10.1109/TSE.2010.80.
- 22 S. Mac Lane. *Categories for the Working Mathematician*. Springer, Berlin, 2nd edition, 1998.
- 23 Joël Ouaknine and James Worrell. Decision problems for linear recurrence sequences. In *RP*, volume 7550 of *Lecture Notes in Computer Science*, pages 21–28. Springer, 2012. doi:10.1007/978-3-642-33512-9\_3.

- 24 Joël Ouaknine and James Worrell. On linear recurrence sequences and loop termination. *ACM SIGLOG News*, 2(2):4–13, 2015. doi:10.1145/2766189.2766191.
- 25 Christos H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.
- 26 Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics. Wiley, 1994. doi:10.1002/9780470316887.
- 27 Toru Takisaka, Yuichiro Oyabu, Natsuki Urabe, and Ichiro Hasuo. Ranking and repulsing supermartingales for reachability in randomized programs. *ACM Trans. Program. Lang. Syst.*, 43(2), June 2021. doi:10.1145/3450967.
- 28 Yun Chen Tsai, Kittiphon Phalakarn, S. Akshay, and Ichiro Hasuo. Chance and mass interpretations of probabilities in markov decision processes (extended version). *arXiv preprint*, 2025.
- 29 Mingsheng Ying, Yuan Feng, and Shenggang Ying. Optimal policies for quantum markov decision processes. *Int. J. Autom. Comput.*, 18(3):410–421, 2021. doi:10.1007/S11633-021-1278-Z.