Trading Performance for Stability in Markov Decision Processes

Tomáš Brázdil*, Krishnendu Chatterjee†, Vojtěch Forejt‡*, and Antonín Kučera*
*Faculty of Informatics, Masaryk University ({xbrazdil,kucera}@fi.muni.cz)
†IST Austria (krish.chat@gmail.com)
‡Department of Computer Science, University of Oxford (vojfor@cs.ox.ac.uk)

***Abstract*—We study the complexity of central controller synthesis problems for finite-state Markov decision processes, where the objective is to optimize *both* the expected mean-payoff performance of the system and its stability. We argue that the basic theoretical notion of expressing the stability in terms of the variance of the mean-payoff (called *global* variance in our paper) is not always sufficient, since it ignores possible instabilities on respective runs. For this reason we propose alernative definitions of stability, which we call *local* and *hybrid* variance, and which express how rewards on each run deviate from the run's own mean-payoff and from the expected mean-payoff, respectively.**

We show that a strategy ensuring both the expected mean-payoff and the variance below given bounds requires randomization and memory, under all the above semantics of variance. We then look at the problem of determining whether there is a such a strategy. For the global variance, we show that the problem is in PSPACE, and that the answer can be approximated in pseudo-polynomial time. For the hybrid variance, the analogous decision problem is in NP, and a polynomial-time approximating algorithm also exists. For local variance, we show that the decision problem is in NP. Since the overall performance can be traded for stability (and vice versa), we also present algorithms for approximating the associated Pareto curve in all the three cases.

Finally, we study a special case of the decision problems, where we require a given expected mean-payoff together with *zero* variance. Here we show that the problems can be all solved in polynomial time.

I. Introduction

Markov decision processes (MDPs) are a standard model for stochastic dynamic optimization. Roughly speaking, an MDP consists of a finite set of states, where in each state, one of the finitely many actions can be chosen by a controller. For every action, there is a fixed probability distribution over the states. The execution begins in some initial state where the controller selects an outgoing action, and the system evolves into another state according to the distribution associated with the chosen action. Then, another action is chosen by the controller, and so on. A *strategy* is a recipe for choosing actions. In general, a strategy may depend on the execution history (i.e., actions may be chosen differently when revisiting the same state) and the choice of actions can be randomized (i.e., the strategy specifies a probability distribution over the available actions). Fixing a strategy for the controller makes the behaviour of a given MDP fully probabilistic and determines the usual probability space over its *runs*, i.e., infinite sequences of states and actions.

A fundamental concept of performance and dependability analysis based on MDP models is *mean-payoff*. Let us assume that every action is assigned some rational *reward*, which corresponds to some costs (or gains) caused by the action. The mean-payoff of a given run is then defined as the long-run average reward per executed action, i.e., the limit of partial averages computed for longer and longer prefixes of a given run. For every strategy σ, the overall performance (or throughput) of the system controlled by σ then corresponds to the expected value of mean-payoff, i.e., the *expected mean-payoff*. It is well known (see, e.g., [15]) that optimal strategies for minimizing/maximizing the expected mean-payoff are positional (i.e., deterministic and independent of execution history), and can be computed in polynomial time. However, the quality of services provided by a given system often depends not only on its overall performance, but also on its *stability*. For example, an optimal controller for a live video streaming system may achieve the expected throughput of approximately 2 MBits/sec. That is, if a user connects to the server many times, he gets 2 Mbits/sec connection *on average*. If an acceptable video quality requires at least 1.8 Mbits/sec, the user is also interested in the likelihood that he gets at least 1.8 Mbits/sec. That is, he requires a certain level of *overall stability* in service quality, which can be measured by the *variance* of mean-payoff, called *global variance* in this paper. The basic computational question is *"given rationals u and v, is there a strategy that achieves the expected mean-payoff u (or better) and variance v (or better)?"*. Since the expected mean-payoff can be "traded" for smaller global variance, we are also interested in approximating the associated *Pareto curve* consisting of all points (u, v) such that (1) there is a strategy achieving the expected mean-payoff u and global variance v; and (2) no strategy can improve u or v without worsening the other parameter.

The global variance says how much the actual mean-payoff of a run tends to deviate from the expected mean-payoff. However, it does not say *anything* about the stability of individual runs. To see this, consider again the video streaming system example, where we now assume that although the connection is guaranteed to be fast on average, the amount of data delivered per second may change substantially along the executed run for example due to a faulty network infrastructure. For simplicity, let us suppose that performing one action in the underlying MDP model takes one second, and the reward assigned to a given action corresponds to the amount of transferred data. The above scenario can be modeled

by saying that 6 Mbits are downloaded every third action, and 0 Mbits are downloaded in other time frames. Then the user gets 2 Mbits/sec connection almost surely, but since the individual runs are apparently "unstable", he may still see a lot of stuttering in the video stream. As an appropriate measure for the stability of individual runs, we propose *local variance*, which is defined as the long-run average of $(r_i(\omega) - mp(\omega))^2$, where $r_i(\omega)$ is the reward of the i-th action executed in a run ω and $mp(\omega)$ is the mean-payoff of ω. Hence, local variance says how much the rewards of the actions executed along a given run deviate from the mean-payoff of the run on average. For example, if the mean-payoff of a run is 2 Mbits/sec and all of the executed actions deliver 2 Mbits, then the run is "absolutely smooth" and its local variance is zero. The level of "local stability" of the whole system (under a given strategy) then corresponds to the *expected local variance*. The basic algorithmic problem for local variance is similar to the one for global variance, i.e., *"given rationals u and v, is there a strategy that achieves the expected mean-payoff u (or better) and the expected local variance v (or better)?"*. We are also interested in the underlying Pareto curve.

Observe that the global variance and the expected local variance capture different and to a large extent *independent* forms of systems' (in)stability. Even if the global variance is small, the expected local variance may be large, and vice versa. In certain situations, we might wish to minimize *both* of them at the same. Therefore, we propose another notion of *hybrid variance* as a measure for "combined" stability of a given system. Technically, the hybrid variance of a given run ω is defined as the long-run average of $(r_i(\omega) - \mathbb{E}[mp])^2$, where $\mathbb{E}[mp]$ is the expected mean-payoff. That is, hybrid variance says how much the rewards of individual actions executed along a given run deviate from the expected mean-payoff on average. The combined stability of the system then corresponds to the *expected hybrid variance*. One of the most crucial properties motivating the definition of hybrid variance is that the expected hybrid variance is small iff both the global variance and the expected local variance are small (in particular, for a prominent class of strategies the expected hybrid variance is a sum of expected local and global variances). The studied algorithmic problems for hybrid variance are analogous to the ones for global and local variance.

The Results. Our results are as follows:

1) *(Global variance).* The global variance problem was considered before but only under the restriction of memoryless strategies [18]. We first show that in general randomized memoryless strategies are not sufficient for Pareto optimal points for global variance (Example 1). We then establish that 2-memory strategies are sufficient. We show that the basic algorithmic problem for global variance is in PSPACE, and the approximate version can be solved in pseudo-polynomial time.
2) *(Local variance).* The local variance problem comes with new conceptual challenges. For example, for unichain MDPs, deterministic memoryless strategies are sufficient for global variance, whereas we show (Example 2) that even for unichain MDPs both randomization and memory is required for local variance. We establish that 3-memory strategies are sufficient for Pareto optimality for local variance. We show that the basic algorithmic problem (and hence also the approximate version) is in NP.
3) *(Hybrid variance).* After defining hybrid variance, we establish that for Pareto optimality 2-memory strategies are sufficient, and in general randomized memoryless strategies are not. We show the basic algorithmic problem for hybrid variance is in NP, and the approximate version can be solved in polynomial time.
4) *(Zero variance).* Finally, we consider the problem where the variance is optimized to zero (as opposed to a given non-negative number in the general case). In this case, we present polynomial-time algorithms to compute the optimal mean-payoff that can be ensured with zero variance (if zero variance can be ensured) for all the three cases. The polynomial-time algorithms for zero variance for mean-payoff objectives is in sharp contrast to the NP-hardness for cumulative reward MDPs [13].

To prove the above results, one has to overcome various obstacles. For example, although at multiple places we build on the techniques of [10] and [2] which allow us to deal with maximal end components of an MDP separately, we often need to extend these techniques, since unlike the above works which study multiple "independent" objectives, in the case of global and hybrid variance any change of value in the expected mean payoff implies a change of value of the variance. Also, since we do not impose any restrictions on the structure of the strategies, we cannot even assume that the limits defining the mean-payoff and the respective variances exist; this becomes apparent in the case of local and hybrid variance, where we need to rely on delicate techniques of selecting runs from which the limits can be extracted. Another complication is that while most of the work on multi-objective verification deals with linear objective functions, our objective functions are inherently quadratic due to the definition of variance.

The summary of our results is presented in Table I. A simple consequence of our results is that the Pareto curves can be approximated in pseudo-polynomial time in the case of global and hybrid variance, and in exponential time for local variance.

Related Work. Studying the trade-off between multiple objectives in an MDP has attracted significant attention in the recent years (see [1] for overview). In the verification area, MDPs with multiple mean-payoff objectives [2], discounted objectives [7], cumulative reward objectives [12], and multiple ω-regular objectives [10] have been studied. As for the stability of a system, the variance penalized mean-payoff problem (where the mean-payoff is penalized by a constant times the variance) under memoryless (stationary) strategies was studied in [11]. The mean-payoff variance trade-off problem for unichain MDPs was considered in [8], where a solution using quadratic programming was designed; under memoryless

	Memory size	Complexity	Approx. complexity	Zero-var. complexity
Global	2-memory LB: Example 1, UB: Theorem 1	PSPACE (Theorem 1)	Pseudo-polynomial (Theorem 1)	PTIME (Theorem 4)
Local	LB: 2-memory (Example 2) UB: 3-memory (Theorem 2)	NP (Theorem 2)	NP	PTIME (Theorem 4)
Hybrid	2-memory LB: Example 4, UB: Theorem 3	NP (Theorem 3)	PTIME (Theorem 3)	Quadratic (Theorem 4)

TABLE I
SUMMARY OF THE RESULTS, WHERE LB AND UB DENOTES LOWER- AND UPPER-BOUND, RESPECTIVELY.

(stationary) strategies the problem was considered in [18]. All the above works for mean-payoff variance trade-off consider the global variance, and are restricted to memoryless strategies. The problem for general strategies and global variance was not solved before. Although restrictions to unichains or memoryless strategies are feasible in some areas, many systems modelled as MDPs might require more general approach. For example, a decision of a strategy to shut the system down might make it impossible to return the running state again, yielding in a non-unichain MDP. Similarly, it is natural to synthesise strategies that change their decisions over time.

As regards other types of objectives, no work considers the local and hybrid variance problems. The variance problem for *discounted* reward MDPs was studied in [17]. The trade-off of expected value and variance of *cumulative* reward in MDPs was studied in [13], showing the zero variance problem to be NP-hard. This contrasts with our results, since in our setting we present polynomial-time algorithms for zero variance.

II. Preliminaries

We use $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, and $\mathbb{R}$ to denote the sets of positive integers, integers, rational numbers, and real numbers, respectively. We assume familiarity with basic notions of probability theory, e.g., *probability space*, *random variable*, or *expected value*. As usual, a *probability distribution* over a finite or countable set X is a function $f : X \to [0,1]$ such that $\sum_{x\in X} f(x) = 1$. We call f *positive* if $f(x) > 0$ for every $x \in X$, *rational* if $f(x) \in \mathbb{Q}$ for every $x \in X$, and *Dirac* if $f(x) = 1$ for some $x \in X$. The set of all distributions over X is denoted by $dist(X)$.

For our purposes, a *Markov chain* is a triple $M = (L, \to, \mu)$ where L is a finite or countably infinite set of *locations*, $\to \subseteq L \times (0,1] \times L$ is a *transition relation* such that for each fixed $\ell \in L$, $\sum_{\ell \xrightarrow{x} \ell'} x = 1$, and μ is the *initial probability distribution* on L. A *run* in M is an infinite sequence $\omega = \ell_1 \ell_2 \ldots$ of locations such that $\ell_i \xrightarrow{x} \ell_{i+1}$ for every $i \in \mathbb{N}$. A *finite path* in M is a finite prefix of a run. Each finite path w in M determines the set $Cone(w)$ consisting of all runs that start with w. To M we associate the probability space $(Runs_M, \mathcal{F}, \mathbb{P})$, where $Runs_M$ is the set of all runs in M, $\mathcal{F}$ is the σ-field generated by all $Cone(w)$ for finite paths w, and $\mathbb{P}$ is the unique probability measure such that $\mathbb{P}(Cone(\ell_1, \ldots, \ell_k)) = \mu(\ell_1) \cdot \prod_{i=1}^{k-1} x_i$, where $\ell_i \xrightarrow{x_i} \ell_{i+1}$ for all $1 \le i < k$ (the empty product is equal to 1).

Markov decision processes. A *Markov decision process* (MDP) is a tuple $G = (S, A, Act, \delta)$ where S is a *finite* set of states, A is a *finite* set of actions, $Act : S \to 2^A \setminus \{\emptyset\}$ is an action enabledness function that assigns to each state s the set $Act(s)$ of actions enabled at s, and $\delta : S \times A \to dist(S)$ is a probabilistic transition function that given a state s and an action $a \in Act(s)$ enabled at s gives a probability distribution over the successor states. For simplicity, we assume that every action is enabled in exactly one state, and we denote this state $Src(a)$. Thus, henceforth we will assume that $\delta : A \to dist(S)$.

A *run* in G is an infinite alternating sequence of states and actions $\omega = s_1 a_1 s_2 a_2 \ldots$ such that for all $i \ge 1$, $Src(a)_i = s_i$ and $\delta(a_i)(s_{i+1}) > 0$. We denote by $Runs_G$ the set of all runs in G. A *finite path* of length k in G is a finite prefix $w = s_1 a_1 \ldots a_{k-1} s_k$ of a run, and we use $last(w) = s_k$ for the last state of w. Given a run $\omega \in Runs_G$, we denote by $A_i(\omega)$ the i-th action a_i of ω.

A pair (T, B) with $\emptyset \neq T \subseteq S$ and $B \subseteq \bigcup_{t\in T} Act(t)$ is an *end component* of G if (1) for all $a \in B$, if $\delta(a)(s') > 0$ then $s' \in T$; and (2) for all $s, t \in T$ there is a finite path $w = s_1 a_1 \ldots a_{k-1} s_k$ such that $s_1 = s$, $s_k = t$, and all states and actions that appear in w belong to T and B, respectively. An end component (T, B) is a *maximal end component (MEC)* if it is maximal wrt. pointwise subset ordering. The set of all MECs of G is denoted by $MEC(G)$. Given an end component $C = (T, B)$, we sometimes abuse notation by considering C as the disjoint union of T and B (e.g. we write $S \cap C$ to denote the set T). For a given $C \in MEC(G)$, we use R_C to denote the set of all runs $\omega = s_1 a_1 s_2 a_2 \ldots$ that eventually *stay* in C, i.e., there is $k \in \mathbb{N}$ such that for all $k' \ge k$ we have that $s_{k'}, a_{k'} \in C$.

Strategies and plays. Intuitively, a strategy in an MDP G is a "recipe" to choose actions. Usually, a strategy is formally defined as a function $\sigma : (SA)^* S \to dist(A)$ that given a finite path w, representing the execution history, gives a probability distribution over the actions enabled in $last(w)$. In this paper we adopt a definition which is equivalent to the standard one, but more convenient for our purpose. Let M be a finite or countably infinite set of *memory elements*. A *strategy* is a triple $\sigma = (\sigma_u, \sigma_n, \alpha)$, where $\sigma_u : A \times S \times \mathsf{M} \to dist(\mathsf{M})$ and $\sigma_n : S \times \mathsf{M} \to dist(A)$ are *memory update* and *next move* functions, respectively, and α is an initial distribution on memory elements. We require that for all $(s, m) \in S \times \mathsf{M}$, the distribution $\sigma_n(s, m)$ assigns a positive value only to actions enabled at s. The set of all strategies is denoted by Σ (the underlying MDP G will be always clear from the context).

A *play* of G determined by an initial state $s \in S$ and a strategy σ is a Markov chain G_s^σ (or G^σ if s is clear from the context) where the set of locations is $S \times \mathsf{M} \times A$, the initial distribution μ is positive only on (some) elements of $\{s\} \times \mathsf{M} \times A$ where $\mu(s, m, a) = \alpha(m) \cdot \sigma_n(s, m)(a)$, and $(t, m, a) \xrightarrow{x} (t', m', a')$ iff $x = \delta(a)(t') \cdot \sigma_u(a, t', m)(m') \cdot \sigma_n(t', m')(a') > 0$. Hence, G_s^σ

starts in a location chosen randomly according to α and σ_n. In a current location (t, m, a), the next action to be performed is a, hence the probability of entering t' is $\delta(a)(t')$. The probability of updating the memory to m' is $\sigma_u(a, t', m)(m')$, and the probability of selecting a' as the next action is $\sigma_n(t', m')(a')$. Since these choices are independent (in the probability theory sense), we obtain the product above.

Note that every run in G_s^σ determines a unique run in G. Hence, every notion originally defined for the runs in G can also be used for the runs in G_s^σ, and we use this fact implicitly at many places in this paper. For example, we use the symbol R_C to denote the set of all runs in G_s^σ that eventually stay in C, certain functions originally defined over $Runs_G$ are interpreted as random variables over the runs in G_s^σ, etc.

Strategy types. In general, a strategy may use infinite memory, and both σ_u and σ_n may randomize. A strategy is *pure* (or *deterministic*) if α is Dirac and both the memory update and the next move functions give a Dirac distribution for every argument, and *stochastic-update* if α, σ_u, and σ_n are unrestricted. Note that every pure strategy is stochastic-update. A *randomized* strategy is a strategy which is not necessarily pure. We also classify the strategies according to the size of memory they use. Important subclasses are *memoryless* strategies, in which M is a singleton, *n-memory* strategies, in which M has exactly n elements, and *finite-memory* strategies, in which M is finite.

For a finite-memory strategy σ, a *bottom strongly connected component* (BSCC) of G_s^σ is a subset of locations $W \subseteq S \times \mathsf{M} \times A$ such that for all $\ell_1 \in W$ and $\ell_2 \in S \times \mathsf{M} \times A$ we have that (i) if ℓ_2 is reachable from ℓ_1, then $\ell_2 \in W$, and (ii) for all $\ell_1, \ell_2 \in W$ we have that ℓ_2 is reachable from ℓ_1. Every BSCC W determines a unique end component $(\{s \mid (s, m, a) \in W\}, \{a \mid (s, m, a) \in W\})$, and we sometimes do not distinguish between W and its associated end component.

An MDP is *strongly connected* if all its states form a single (maximal) end component. A strongly connected MDP is a *unichain* if for all end components (T, B) we have $T = S$.

Throughout this paper we will use the following standard result about MECs.

Lemma 1 ([9, Proposition 3.1])**.** *Almost all runs eventually end in a MEC, i.e.* $\mathbb{P}_s^\sigma\left[\bigcup_{C \in Mec(G)} R_C\right] = 1$ *for all* σ *and* s.

Global, local, and hybrid variance. Let $G = (S, A, Act, \delta)$ be an MDP, and $r : A \to \mathbb{Q}$ a *reward function*. We define the *mean-payoff* of a run $\omega \in Runs_G$ by

$$mp(\omega) = \limsup_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n-1} r(A_i(\omega)) \,.$$

The expected value and variance of mp in G_s^σ are denoted by $\mathbb{E}_s^\sigma[mp]$ and $\mathbb{V}_s^\sigma[mp]$, respectively (recall that $\mathbb{V}_s^\sigma[mp] = \mathbb{E}_s^\sigma\left[(mp - \mathbb{E}_s^\sigma[mp])^2\right] = \mathbb{E}_s^\sigma\left[mp^2\right] - (\mathbb{E}_s^\sigma[mp])^2$). Intuitively, $\mathbb{E}_s^\sigma[mp]$ corresponds to the "overall performance" of G_s^σ, and $\mathbb{V}_s^\sigma[mp]$ is a measure of "global stability" of G_s^σ indicating how much the mean payoffs of runs in G_s^σ tend to deviate from $\mathbb{E}_s^\sigma[mp]$ (see Section I). In the rest of this paper, we refer to $\mathbb{V}_s^\sigma[mp]$ as *global variance*.

The stability of a given run $\omega \in Runs_G$ (see Section I) is measured by its *local variance* defined as follows:

$$lv(\omega) = \limsup_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n-1} \left(r(A_i(\omega)) - mp(\omega)\right)^2$$

Note that $lv(\omega)$ is not really a "variance" in the usual sense of probability theory[1]. We call the function $lv(\omega)$ "local variance" because we find this name suggestive; $lv(\omega)$ is the long-run average square of the distance from $mp(\omega)$. The expected value of lv in G_s^σ is denoted by $\mathbb{E}_s^\sigma[lv]$.

Finally, given a run ω in G_s^σ, we define the *hybrid variance* of ω in G_s^σ as follows:

$$hv(\omega) = \limsup_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n-1} \left(r(A_i(\omega)) - \mathbb{E}_s^\sigma[mp]\right)^2$$

The definition of $hv(\omega)$ depends on the expected mean payoff, and hence it makes sense only after fixing a strategy σ and a state s. Sometimes we also write $hv^{\sigma,s}(\omega)$ instead of $hv(\omega)$ to prevent confusion about the underlying σ and s. The expected value of hv in G_s^σ is denoted by $\mathbb{E}_s^\sigma[hv]$. Intuitively, $\mathbb{E}_s^\sigma[hv]$ measures the "combined" stability of G_s^σ (see Section I).

From now on, we restrict ourselves to reward functions which only assign non-negative rewards. This is w.l.o.g., since for a reward function r we can define r' given by $r'(a) = r(a) + \min_{a' \in A} r(a')$ for all a, and under any strategy σ the expected mean payoff w.r.t. r' is by $\min_{a' \in A} r(a')$ greater than the one w.r.t. r, and the variances remain unchanged.

Pareto optimality. We say that a strategy σ is *Pareto optimal* in s wrt. global variance if for every strategy ζ we have that $(\mathbb{E}_s^\sigma[mp], \mathbb{V}_s^\sigma[mp]) \geq (\mathbb{E}_s^\zeta[mp], \mathbb{V}_s^\zeta[mp])$ implies $(\mathbb{E}_s^\sigma[mp], \mathbb{V}_s^\sigma[mp]) = (\mathbb{E}_s^\zeta[mp], \mathbb{V}_s^\zeta[mp])$, where $\geq$ is the standard component-wise ordering. Similarly, we define Pareto optimality of σ wrt. local and hybrid variance by replacing $\mathbb{V}_s^\alpha[mp]$ with $\mathbb{E}_s^\alpha[lv]$ and $\mathbb{E}_s^\alpha[hv]$, respectively. We choose the order $\geq$ for technical convenience, if one wishes to maximize the expected value while minimizing the variance, it suffices to multiply all rewards by -1. The *Pareto curve* for s wrt. global, local, and hybrid variance consists of all points of the form $(\mathbb{E}_s^\sigma[mp], \mathbb{V}_s^\sigma[mp])$, $(\mathbb{E}_s^\sigma[mp], \mathbb{E}_s^\sigma[lv])$, and $(\mathbb{E}_s^\sigma[mp], \mathbb{E}_s^\sigma[hv])$, where σ is a Pareto optimal strategy wrt. global, local, and hybrid variance, respectively.

Frequency functions. Let C be a MEC. We say that $f : C \cap A \to [0, 1]$ is a *frequency function on* C if

- $\sum_{a \in C \cap A} f(a) = 1$
- $\sum_{a \in C \cap A} f(a) \cdot \delta(a)(s) = \sum_{a \in Act(s)} f(a)$ for every $s \in C \cap S$

Define $mp[f] := \sum_{a \in C} f(a) \cdot r(a)$ and $lv[f] := \sum_{a \in C} f(a) \cdot (r(a) - mp[f])^2$.

[1] By investing some effort, one could perhaps find a random variable X such that $lv(\omega)$ is the variance of X, but this question is not really relevant—we only use lv as a *random variable which measures the level of local stability of runs*. One could perhaps study the variance of lv, but this is beyond the scope of this paper. The same applies to the function hv.

The studied problems. In this paper, we study the following basic problems connected to the three stability measures introduced above (below V_s^σ is either $\mathbb{V}_s^\sigma[mp]$, $\mathbb{E}_s^\sigma[lv]$, or $\mathbb{E}_s^\sigma[hv]$):

- *Pareto optimal strategies and their memory*. Do Pareto optimal strategies exist for all points on the Pareto curve? Do Pareto optimal strategies require memory and randomization in general? Do strategies achieving non-Pareto points require memory and randomization in general?
- *Deciding strategy existence*. For a given MDP G, an initial state s, a rational reward function r, and a point $(u, v) \in \mathbb{Q}^2$, we ask whether there exists a strategy σ such that $(\mathbb{E}_s^\sigma[mp], V_s^\sigma) \leq (u, v)$.
- *Approximation of strategy existence*. For a given MDP G, an initial state s, a rational reward function r, a number ε and a point $(u, v) \in \mathbb{Q}^2$, we want to get an algorithm which (a) outputs "yes" if there is a strategy σ such that $(\mathbb{E}_s^\sigma[mp], V_s^\sigma) \leq (u - \varepsilon, v - \varepsilon)$; (b) outputs "no" if there is no strategy such that $(\mathbb{E}_s^\sigma[mp], V_s^\sigma) \leq (u, v)$.
- *Strategy synthesis.* If there exists a strategy σ such that $(\mathbb{E}_s^\sigma[mp], V_s^\sigma) \leq (u, v)$, we wish to *compute* such strategy. Note that it is not *a priori* clear that σ is finitely representable, and hence we also need to answer the question what *type* of strategies is needed to achieve Pareto optimal points.
- *Optimal performance with zero-variance.* Here we are interested in deciding if there exists a Pareto point of the form $(u, 0)$ and computing the value of u, i.e., the optimal expected mean payoff achievable with "absolute stability" (note that the variance is always non-negative and its value 0 corresponds to stable behaviours).

Remark 1. *If the approximation of strategy existence problem is decidable, we design the following algorithm to approximate the Pareto curve up to an arbitrarily small given $\varepsilon > 0$. We compute a finite set of points $P \subseteq \mathbb{Q}^2$ such that (1) for every Pareto point (u, v) there is $(u', v') \in P$ with $(|u - u'|, |v - v'|) \leq (\varepsilon, \varepsilon)$, and (2) for every $(u', v') \in P$ there is a Pareto point (u, v) such that $(|u - u'|, |v - v'|) \leq (\varepsilon, \varepsilon)$. Let $R = \max_{a \in A} |r(a)|$. Note that $|\mathbb{E}_s^\sigma[mp]| \leq R$ and $V_s^\sigma \leq R^2$ for an arbitrary strategy σ. Hence, the set P is computable by a naive algorithm which decides the approximation of strategy existence for $O(|R|^3/\varepsilon^2)$ points in the corresponding ε-grid and puts $O(|R|^2/\varepsilon)$ points into P. The question whether the three Pareto curves can be approximated more efficiently by sophisticated methods based on deeper analysis of their properties is left for future work.*

III. Global variance

In the rest of this paper, unless specified otherwise, we suppose we work with a fixed MDP $G = (S, A, Act, \delta)$ and a reward function $r : A \to \mathbb{Q}$. We start by proving that both memory and randomization is needed even for achieving non-Pareto points; this implies that memory and randomization is needed even to approximate the value of Pareto points. Then we show that 2-memory stochastic update strategies are sufficient, which gives a tight bound.

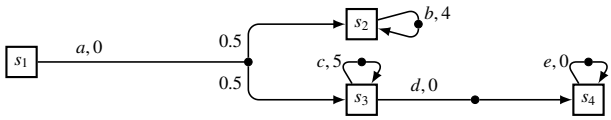

Fig. 1. An MDP witnessing the need for memory and randomization in Pareto optimal strategies for global variance.

Example 1. *Consider the MDP of Fig. 1. Observe that the point $(4, 2)$ is achievable by a strategy σ which selects c with probability $\frac{4}{5}$ and d with probability $\frac{1}{5}$ upon the* first *visit to s_3; in every other visit to s_3, the strategy σ selects c with probability 1. Hence, σ is a 2-memory randomized strategy which stays in MEC $C = (\{s_3\}, \{c\})$ with probability $\frac{1}{2} \cdot \frac{4}{5} = \frac{2}{5}$. Clearly, $\mathbb{E}_{s_1}^\sigma[mp] = \frac{1}{2} \cdot 4 + \frac{1}{2} \cdot \frac{4}{5} \cdot 5 + \frac{1}{2} \cdot \frac{1}{5} \cdot 0 = 4$ and $\mathbb{V}_{s_1}^\sigma[mp] = \frac{1}{2} \cdot 4^2 + \frac{1}{2} \cdot \frac{4}{5} \cdot 5^2 + \frac{1}{2} \cdot \frac{1}{5} \cdot 0^2 - 4^2 = 2$. Further, note that every strategy $\bar{\sigma}$ which stays in C with probability x satisfies $\mathbb{E}_{s_1}^{\bar{\sigma}}[mp] = \frac{1}{2} \cdot 4 + x \cdot 5$ and $\mathbb{V}_{s_1}^{\bar{\sigma}}[mp] = \frac{1}{2} \cdot 4^2 + x \cdot 5^2 - (2 + x \cdot 5)^2$. For $x > \frac{2}{5}$ we get $\mathbb{E}_{s_1}^{\bar{\sigma}}[mp] > 4$, and for $x < \frac{2}{5}$ we get $\mathbb{V}_{s_1}^{\bar{\sigma}}[mp] > 2$, so $(4, 2)$ is indeed a Pareto point. Every deterministic (resp. memoryless) strategy can stay in C with probability either $\frac{1}{2}$ or 0, giving $\mathbb{E}_{s_1}^{\bar{\sigma}}[mp] = \frac{9}{2}$ or $\mathbb{V}_{s_1}^{\bar{\sigma}}[mp] = 4$. So, both memory and randomization are needed to achieve the Pareto point $(4, 2)$ or a non-Pareto point $(4.1, 2.1)$.*

Interestingly, if the MDP is strongly connected, memoryless deterministic strategies always suffice, because in this case a memoryless strategy that minimizes the expected mean payoff immediately gets zero variance. This is in contrast with local and hybrid variance, where we will show that memory and randomization is required in general already for unichain MDPs. For the general case of global variance, the sufficiency of 2-memory strategies is captured by the following theorem.

Theorem 1. *If there is a strategy ζ satisfying $(\mathbb{E}_s^\zeta[mp], \mathbb{V}_s^\zeta[mp]) \leq (u, v)$, then there is a 2-memory strategy with the same properties. Moreover, Pareto optimal strategies always exist, the problem whether there is a strategy achieving a point (u, v) is in PSPACE, and approximation of the answer can be done in pseudo-polynomial time.*

Note that every $C \in MEC(G)$ can be seen as a strongly connected MDP. By using standard linear programming methods (see, e.g., [15]), for every $C \in MEC(G)$ we can compute the *minimal* and the *maximal* expected mean payoff achievable in C, denoted by α_C and β_C, in polynomial time (since C is strongly connected, the choice of initial state is irrelevant). Thus, we can also compute the system L of Fig. 2 in polynomial time. We show the following:

Proposition 1. *Let $s \in S$ and $u, v \in \mathbb{R}$.*

1) *If there is a strategy ζ with $(\mathbb{E}_s^\zeta[mp], \mathbb{V}_s^\zeta[mp]) \leq (u, v)$ then the system L of Fig. 2 has a non-negative solution.*
2) *If the system L of Fig. 2 has a non-negative solution, then there is a 2-memory stochastic-update strategy σ and $z \in \mathbb{R}$ with $(\mathbb{E}_s^\sigma[mp], \mathbb{V}_s^\sigma[mp]) \leq (u, v)$ and for all $C \in MEC(G)$ we have the following: If $\alpha_C > z$, then $x_C = \alpha_C$; if $\beta_C < z$, then $x_C = \beta_C$; otherwise $x_C = z$.*

$$\mathbf{1}_s(t) + \sum_{a \in A} y_a \cdot \delta(a)(t) = \sum_{a \in Act(t)} y_a + y_t \quad \text{for all } t \in S \tag{1}$$

$$\sum_{\substack{C \in MEC(G) \\ t \in S \cap C}} y_t = 1 \tag{2}$$

$$y_\kappa \geq 0 \qquad \text{for all } \kappa \in S \cup A \tag{3}$$

$$\alpha_C \leq x_C \qquad \text{for all } C \in MEC(G) \tag{4}$$

$$x_C \leq \beta_C \qquad \text{for all } C \in MEC(G) \tag{5}$$

$$u \geq \sum_{C \in MEC(G)} x_C \cdot \sum_{t \in S \cap C} y_t \tag{6}$$

$$v \geq \Big(\sum_{C \in MEC(G)} x_C^2 \cdot \sum_{t \in S \cap C} y_t \Big) - \Big(\sum_{C \in MEC(G)} x_C \cdot \sum_{t \in S \cap C} y_t \Big)^2 \tag{7}$$

Fig. 2. The system L. (Here $\mathbf{1}_{s_0}(s) = 1$ if $s = s_0$, and $\mathbf{1}_{s_0}(s) = 0$ otherwise.)

Observe that the existence of Pareto optimal strategies follows from the above proposition, since we define points (u, v) that some strategy can achieve by a continous function from values x_C and $\sum_{t \in S \cap C} y_t$ for $C \in MEC(G)$ to $\mathbb{R}^2$. Because the domain is bounded (all x_C and $\sum_{t \in S \cap C} y_t$ have minimal and maximal values they can achieve) and closed (the points of the domain are expressible as a projection of feasible solutions of a linear program), it is also compact, and a continuous map of a compact set is compact [16], and hence closed.

Let us briefly sketch the proof of Proposition 1, which combines new techniques with results of [2], [10]. We start with Item 1. Let ζ be a strategy satisfying $(\mathbb{E}_s^\zeta[mp], \mathbb{V}_s^\zeta[mp]) \leq (u, v)$. First, note that almost every run of G_s^ζ eventually stays in some MEC of G by Lemma 1. The way how ζ determines the values of all y_κ, where $\kappa \in S \cup A$, is exactly the same as in [2] and it is based on the ideas of [10]. The details are given in [3]. The important property preserved is that for every $C \in MEC(G)$ and every state $t \in S \cap C$, the value of y_t corresponds to the probability that a run stays in C and enters C via the state t. Hence, $\sum_{t \in S \cap C} y_t$ is the probability that a run of G_s^ζ eventually stays in C. The way how ζ determines the value of y_a, where $a \in A$, is explained in [3]. The value of x_C is the conditional expected mean payoff under the condition that a run stays in C, i.e., $x_C = \mathbb{E}_s^\zeta[mp \mid R_C]$. Hence, $\alpha_C \leq x_C \leq \beta_C$, which means that (4) and (5) are satisfied. Further, $\mathbb{E}_s^\zeta[mp] = \sum_{C \in MEC(G)} x_C \cdot \sum_{t \in S \cap C} y_t$, and hence (6) holds. Note that $\mathbb{V}_s^\zeta[mp]$ is *not* necessarily equal to the right-hand side of (7), and hence it is not immediately clear why (7) should hold. Here we need the following lemma (a proof is given in [3]):

Lemma 2. *Let $C \in MEC(G)$, and let $z_C \in [\alpha_C, \beta_C]$. Then there exists a memoryless randomized strategy σ_{z_C} such that for every state $t \in C \cap S$ we have that $\mathbb{P}_t^{\sigma_{z_C}}[mp{=}z_C] = 1$.*

Using Lemma 2, we can define another strategy ζ' from ζ such that for every $C \in MEC(G)$ we have the following: (1) the probability of R_C in G_s^ζ and in $G_s^{\zeta'}$ is the same; (2) almost all runs $\omega \in R_C$ satisfy $mp(\omega) = x_C$. This means that $\mathbb{E}_s^\zeta[mp] = \mathbb{E}_s^{\zeta'}[mp]$, and we show that $\mathbb{V}_s^\zeta[mp] \geq \mathbb{V}_s^{\zeta'}[mp]$ (see [3]). Hence, $(\mathbb{E}_s^{\zeta'}[mp], \mathbb{V}_s^{\zeta'}[mp]) \leq (u, v)$, and therefore (1)–(6) also hold if we use ζ' instead of ζ to determine the values of all variables. Further, the right-hand side of (7) is equal to $\mathbb{V}_s^{\zeta'}[mp]$, and hence (7) holds. This completes the proof of Item 1.

Item 2 is proved as follows. Let y_κ, where $\kappa \in S \cup A$, and x_C, where $C \in MEC(G)$, be a non-negative solution of L. For every $C \in MEC(G)$, we put $y_C = \sum_{t \in S \cap C} y_t$. By using the results of Sections 3 and 5 of [10] and the modifications presented in [2], we first construct a finite-memory stochastic update strategy ϱ such that the probability of R_C in G_s^ϱ is equal to y_C. Then, we construct a strategy $\hat{\sigma}$ which plays according to ϱ until a bottom strongly connected component B of G_s^ϱ is reached. Observe that the set of all states and actions which appear in B is a subset of some $C \in MEC(G)$. From that point on, the strategy $\hat{\sigma}$ "switches" to the memoryless randomized strategy σ_{x_C} of Lemma 2. Hence, $\mathbb{E}_s^\varrho[mp]$ and $\mathbb{V}_s^\varrho[mp]$ are equal to the right-hand sides of (6) and (7), respectively, and thus we get $(\mathbb{E}_s^\varrho[mp], \mathbb{V}_s^\varrho[mp]) \leq (u, v)$. Note that $\hat{\sigma}$ may use more than 2-memory elements. A 2-memory strategy is obtained by modifying the initial part of $\hat{\sigma}$ (i.e., the part before the switch) into a memoryless strategy in the same way as in [2]. Then, $\hat{\sigma}$ only needs to remember whether a switch has already been performed or not, and hence 2 memory elements are sufficient. Finally, we transform $\hat{\sigma}$ into another 2-memory stochastic update strategy σ which satisfies the extra conditions of Item 2 for a suitable z. This is achieved by modifying the behaviour of $\hat{\sigma}$ in some MECs so that the probability of staying in every MEC is preserved, the expected mean payoff is also preserved, and the global variance can only decrease. This part is somewhat tricky and the details are given in [3].

We can solve the strategy existence problem by encoding the existence of a solution to L as a closed formula Φ of the existential fragment of $(\mathbb{R}, +, *, \leq)$. Since Φ is computable in polynomial time and the existential fragment of $(\mathbb{R}, +, *, \leq)$ is decidable in polynomial space [4], we obtain Theorem 1.

The pseudo-polynomial-time approximation algorithm is obtained as follows. First note that if we had the number z above, we could simplify the system L of Fig. 2 by substituting all x_C variables with constants. Then, (4) and (5) can be eliminated, (6) becomes a linear constraint, and (7) the only quadratic constraint. Thus, the system L can be transformed into a quadratic program L_z in which the quadratic constraint is negative semi-definite with rank 1 (see [3]), and hence approximated in polynomial time [20]. Since we do not know the precise number z we try different candidates $\bar{z}$, namely we approximate the value (to the precision $\frac{\varepsilon}{2}$) of $L_{\bar{z}}$ for all numbers $\bar{z}$ between $\min_{a \in A} r(a)$ and $\max_{a \in A} r(a)$ that are a multiple of $\tau = \frac{\varepsilon}{8 \max\{N,1\}}$ where N is the maximal absolute value of an assigned reward. If any $L_{\bar{z}}$ has a solution lower than $u - \frac{\varepsilon}{2}$, we output "yes", otherwise we output "no". The correctness of the algorithm is proved in [3].

Note that if we *knew* the constant z we would even get that the approximation problem can be solved in polynomial time (assuming that the number of digits in z is polynomial

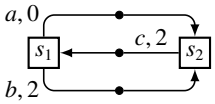

Fig. 3. An MDP showing that Pareto optimal strategies need randomization/memory for local and hybrid variance.

in the size of the problem instance). Unfortunately, our proof of Item 2 does not give a procedure for computing z, and we cannot even conclude that z is rational. We conjecture that the constant z *can* actually be chosen as a rational number with small number of digits (which would immediately lower the complexity of strategy existence to **NP** using the results of [19] for solving negative semi-definite quadratic programs). Also note that Remark 1 and Theorem 1 immediately yield the following result.

Corollary 1. *The approximate Pareto curve for global variance can be computed in pseudo-polynomial time.*

IV. Local variance

In this section we analyse the problem for local variance. As before, we start by showing the lower bounds for memory needed by strategies, and then provide an upper bound together with an algorithm computing a Pareto optimal strategy. As in the case of global variance, Pareto optimal strategies require both randomization and memory, however, in contrast to global variance where for unichain MDPs deterministic memoryless strategies are sufficient we show (in the following example) that for local variance both memory and randomization is required even for unichain MDPs.

Example 2. *Consider the MDP from Figure 3 and consider a strategy σ that in the first step in s_1 makes a random choice uniformly between a and b, and then, whenever the state s_1 is revisited, it chooses the action that was chosen in the first step. The expected mean-payoff under such strategy is* $0.5{\cdot}2{+}0.5{\cdot}1 = 1.5$ *and the variance is* $\big(0.5{\cdot}(0.5{\cdot}(0{-}1)^2{+}0.5{\cdot}(2{-}1)^2)\big){+}\big(0.5{\cdot}(2{-}2)^2\big) = 0.5$. *We show that the point* $(1.5, 0.5)$ *cannot be achieved by any memoryless randomized strategy σ'. Given $x \in \{a, b, c\}$, denote by $f(x)$ the frequency of the action x under σ'. Clearly, $f(c) = 0.5$ and $f(b) = 0.5 - f(a)$. If $f(a) < 0.2$, then the mean-payoff $\mathbb{E}^{\sigma'}_{s_1}[mp] = 2 \cdot (f(c) + f(b)) = 2 - 2f(a)$ is greater than* 1.6. *Assume that $0.2 \le f(a) \le 0.5$. Then $\mathbb{E}^{\sigma'}_{s_1}[mp] \le 1.6$ but the variance is at least* 0.64 *(see [3] for computation). Insufficiency of deterministic history-dependent strategies is proved using the same equations and the fact that there is only one run under such a strategy.*

Thus have shown that memory and randomization is needed to achieve a non-Pareto point $(1.55, 0.6)$. *The need of memory and randomization to achieve Pareto points will follow later from the fact that there always exist Pareto optimal strategies.*

In the remainder of this section we prove the following.

Theorem 2. *If there is a strategy ζ satisfying $(\mathbb{E}^{\zeta}_{s_0}[mp], \mathbb{E}^{\zeta}_{s_0}[lv]) \le (u, v)$ then there is a 3-memory strategy with the same properties. The problem whether such a strategy exists belongs to* **NP**. *Moreover, Pareto optimal strategies always exist.*

We start by proving that 3-memory stochastic update strategies achieve all achievable points wrt. local variance.

Proposition 2. *For every strategy ζ there is a 3-memory stochastic-update strategy σ satisfying*

$$(\mathbb{E}^{\sigma}_{s_0}[mp], \mathbb{E}^{\sigma}_{s_0}[lv]) \quad \le \quad (\mathbb{E}^{\zeta}_{s_0}[mp], \mathbb{E}^{\zeta}_{s_0}[lv])$$

Moreover, the three memory elements of σ, say m_1, m_2, m_2', satisfy the following:

- *The memory element m_1 is initial, σ may randomize in m_1 and may stochastically update its memory either to m_2, or to m_2'.*
- *In m_2 and m_2' the strategy ζ behaves* deterministically *and never changes its memory.*

Proof. By Lemma 1 $\sum_{C \in MEC(G)} \mathbb{P}(R_C) = 1$, and

$$\begin{aligned}&(\mathbb{E}^{\zeta}_{s_0}[mp], \mathbb{E}^{\zeta}_{s_0}[lv])\\ &= \Big(\sum_{C \in MEC(G)} \mathbb{P}(R_C){\cdot}\mathbb{E}^{\zeta}_{s_0}[mp \mid R_C], \sum_{C \in MEC(G)} \mathbb{P}(R_C){\cdot}\mathbb{E}^{\zeta}_{s_0}[lv \mid R_C]\Big).\end{aligned}$$

In what follows we sometimes treat each MEC C as a standalone MDP obtained by restricting G to C. Then, for example, C^{κ} denotes the Markov chain obtained by applying the strategy κ to the component C.

The next proposition formalizes the main idea of our proof:

Proposition 3. *Let C be a MEC. There are two frequency functions $f_C : C \to \mathbb{R}$ and $f'_C : C \to \mathbb{R}$ on C, and a number $p_C \in [0, 1]$ such that the following holds*

$$\begin{aligned}p_C \cdot (mp[f_C], lv[f_C]) + (1 - p_C) \cdot (mp[f'_C], lv[f'_C])\\ \le (\mathbb{E}^{\zeta}_{s_0}[mp|R_C], \mathbb{E}^{\zeta}_{s_0}[lv|R_C])\,.\end{aligned}$$

The proposition is proved in [3], where we first show that it follows from a relaxed version of the proposition which gives us, for any $\varepsilon > 0$, frequency functions f_ε and f'_ε and number p_ε such that

$$\begin{aligned}p_\varepsilon \cdot (mp[f_\varepsilon], lv[f_\varepsilon]) + (1 - p_\varepsilon) \cdot (mp[f'_\varepsilon], lv[f'_\varepsilon])\\ \le (\mathbb{E}^{\zeta}_{s_0}[mp|R_C], \mathbb{E}^{\zeta}_{s_0}[lv|R_C]) + (\varepsilon, \varepsilon)\,.\end{aligned}$$

Then we show that the weaker version holds by showing that there are runs ω from which we can extract the frequency functions f_ε and f'_ε. The selection of runs is rather involved, since it is not clear a priori which runs to pick or even how to extract the frequencies from them (note that the naive approach of considering the average ratio of taking a given action a does not work, since the averages might not be defined).

Proposition 3 implies that any expected mean payoff and local variance achievable on a MEC C can be achieved by a composition of two memoryless randomized strategies giving precisely the frequencies of actions specified by f_C and f'_C (note that $lv[f_C]$ and $lv[f'_C]$ may not be equal to the expected local variance of such strategies, but we show

that the "real" expected local variance cannot be larger). By further selecting BSCCs of these strategies and using some de-randomization tricks we obtain, for every MEC C, two memoryless deterministic strategies π_C and π'_C and a constant h_C such that for every $s \in C \cap S$ the value of $h_C(\mathbb{E}_s^{\pi_C}[mp], \mathbb{E}_s^{\pi_C}[lv]) + (1 - h_C)(\mathbb{E}_s^{\pi'_C}[mp], \mathbb{E}_s^{\pi'_C}[lv])$ is equal to a fixed (u', v') (since both C^{π_C} and $C^{\pi'_C}$ have only one BSCC) satisfying $(u', v') \leq (\mathbb{E}_{s_0}^{\zeta}[mp|R_C], \mathbb{E}_{s_0}^{\zeta}[lv|R_C])$. We define two memoryless deterministic strategies π and π' that in every C behave as π_C and π'_C, respectively. Details of the steps above are in [3].

Using similar arguments as in [2] (that in turn depend on results of [10]) one may show that there is a 2-memory stochastic update strategy σ', with two memory locations m_1, m_2, satisfying the following properties: In m_1, the strategy σ' may randomize and may stochastically update its memory to m_2. In m_2, the strategy σ' never changes its memory. Most importantly, the probability that σ' updates its memory from m_1 to m_2 in a given MEC C is equal to $\mathbb{P}_{s_0}^{\zeta}[R_C]$.

We modify the strategy σ' to the desired 3-memory σ by splitting the memory element m_2 into two elements m_2, m'_2. Whenever σ' updates to m_2, the strategy σ further chooses randomly whether to update either to m_2 (with prob. h_C), or to m'_2 (with prob. $1 - h_C$). Once in m_2 or m'_2, the strategy σ never changes its memory and plays according to π or π', respectively. For every MEC C we have $\mathbb{P}_{s_0}^{\sigma}(\text{update to } m_2 \text{ in } C) = \mathbb{P}(R_C) \cdot h_C$ and $\mathbb{P}_{s_0}^{\sigma}(\text{update to } m'_2 \text{ in } C) = \mathbb{P}(R_C) \cdot (1 - h_C)$. Thus we get

$$(\mathbb{E}_{s_0}^{\zeta}[mp], \mathbb{E}_{s_0}^{\zeta}[lv]) = (\mathbb{E}_{s_0}^{\sigma}[mp], \mathbb{E}_{s_0}^{\sigma}[lv]) \tag{8}$$

as shown in [3]. □

Proposition 2 combined with results of [2] allows us to finish the proof of Theorem 2.

Proof (of Theorem 2). Intuitively, the non-deterministic polynomial time algorithm works as follows: First, guess two memoryless deterministic strategies π and π'. Verify whether there is a 3-memory stochastic update strategy σ with memory elements m_1, m_2, m'_2 which in m_2 behaves as π, and in m'_2 behaves as π' such that $(\mathbb{E}_{s_0}^{\sigma}[mp], \mathbb{E}_{s_0}^{\sigma}[lv]) \leq (u, v)$. Note that it suffices to compute the probability distributions chosen by σ in the memory element m_1 and the probabilities of updating to m_2 and m'_2. This can be done by a reduction to the controller synthesis problem for two dimensional mean-payoff objectives studied in [2].

More concretely, we construct a new MDP $G[\pi, \pi']$ with

- the set of states $S' := \{s_{in}\} \cup (S \times \{m_1, m_2, m'_2\})$ (Intuitively, the m_1, m_2, m'_2 correspond to the memory elements of σ.)
- the set of actions[2] $A \cup \{[\pi], [\pi'], \mathit{default}\}$
- the mapping Act' defined by $Act'(s_{in}) = \{[\pi], [\pi'], \mathit{default}\}$, $Act'((s, m_1)) = Act(s) \cup \{[\pi], [\pi']\}$ and $Act'((s, m_2)) = Act'((s, m'_2)) = \{\mathit{default}\}$ (Intuitively, the actions $[\pi]$ and $[\pi']$ simulate the update of the memory element m_2 and to m'_2, respectively, in σ. As σ is supposed to behave in a fixed way in m_2 and m'_2, we do not need to simulate its behavior in these states in $G[\pi, \pi']$. Hence, the $G[\pi, \pi']$ just loops under the action *default* in the states (s, m_2) and (s, m'_2). The action *default* is also used in the initial state to denote that the initial memory element is m_1.)
- the probabilistic transition function δ' defined as follows:
  - $\delta'(s_{in})(\mathit{default})((s_0, m_1)) = \delta(s_{in}, [\pi])((s_0, m_2)) = \delta(s_{in}, [\pi'])((s_0, m'_2)) = 1$ for $a \in A$ and $t \in S$
  - $\delta'((s, m_1), a)((t, m_1)) = \delta(s, a)(t)$ for $a \in A$ and $t \in S$
  - $\delta'((s, m_1), [\pi])((s, m_2)) = \delta'((s, m_1), [\pi'])((s, m'_2)) = 1$
  - $\delta'((s, m_2), \mathit{default})((s, m_2)) = \delta'((s, m'_2), \mathit{default})((s, m'_2)) = 1$

We define a vector of rewards $\vec{r} : S' \to \mathbb{R}^2$ as follows: $\vec{r}((s, m_2)) := (\mathbb{E}_s^{\pi}[mp], \mathbb{E}_s^{\pi}[lv])$ and $\vec{r}((s, m'_2)) := (\mathbb{E}_s^{\pi'}[mp], \mathbb{E}_s^{\pi'}[lv])$ and $\vec{r}(s_{in}) = \vec{r}((s, m_1)) := (\max_{a \in A} r(a) + 1, (\max_{a \in A} r(a) - \min_{a \in A} r(a))^2 + 1)$. (Here the rewards are chosen in such a way that no (Pareto) optimal scheduler can stay in the states of the form (s, m_1) with positive probability.) Note that $\vec{r}$ can be computed in polynomial time using standard algorithms for computing mean-payoff in Markov chains [14].

In [3] we show that if there is a strategy ζ for G such that $(\mathbb{E}_{s_0}^{\zeta}[mp], \mathbb{E}_{s_0}^{\zeta}[lv]) \leq (u, v)$, then there is a (memoryless randomized) strategy ρ in $G[\pi, \pi']$ such that $(\mathbb{E}_{s_{in}}^{\rho}[mp^{\vec{r}_1}], \mathbb{E}_{s_{in}}^{\rho}[mp^{\vec{r}_2}]) \leq (u, v)$. Also, we show that such ρ can be computed in polynomial time using results of [2]. Finally, it is straightforward to move the second component of the states of $G[\pi, \pi']$ to the memory of a stochastic update strategy which gives a 3-memory stochastic update strategy σ for G with the desired properties. Thus a non-deterministic polynomial time algorithm works as follows: (1) guess π, π' (2) construct $G[\pi, \pi']$ and $\vec{r}$ (3) compute ρ (if it exists). As noted above, ρ can be transformed to the 3-memory stochastic update strategy σ in polynomial time.

Finally, we can show that Pareto optimal strategies exist by a reasoning similar to the one used in global variance. □

Theorem 2 and Remark 1 give the following corollary.

Corollary 2. *The approximate Pareto curve for local variance can be computed in exponential time.*

V. Hybrid variance

We start by showing that memory or randomization is needed for Pareto optimal strategies in unichain MDPs for hybrid variance; and then show that both memory and randomization is required for hybrid variance for general MDPs.

Example 3. *Consider again the MDP from Fig. 3, and any memoryless deterministic strategy. There are in fact two of these. One, which choses a in s_1, yields the variance* 1*, and the other, which chooses b in s_1, yields the expectation* 2*.*

However, a memoryless randomized strategy σ which randomizes uniformly between a and b yields the expectation 1.5

[2]To keep the presentation simple, here we do not require that every action is enabled in at most one step.

and variance 0.75 *which makes it incomparable to either of the memoryless deterministic strategies. Similarly, the deterministic strategy which alternates between a and b on subsequent visits of* s_1 *yields the same values as the* σ *above. This gives us that memory or randomization is needed even to achieve a non-Pareto point* $(1.6, 0.8)$.

Before proceeding with general MDPs, we give the following proposition, which states an interesting and important relation between the three notions of variance[3]. The proposition is proved in [3].

Proposition 4. *Suppose* σ *is a strategy under which for almost all* ω *the limits exists for* $hv(\omega)$, $mp(\omega)$, *and* $lv(\omega)$ *(i.e. the* $\limsup$ *in their definitions can be swapped for* $\lim$*). Then*

$$\mathbb{E}^{\sigma}_{s}[hv] = \mathbb{V}^{\sigma}_{s}[mp] + \mathbb{E}^{\sigma}_{s}[lv] \ .$$

Now we can show that both memory and randomization is needed, by extending Example 1.

Example 4. *Consider again the MDP from Fig. 1. Under every strategy, every run* ω *satisfies* $lv(\omega) = 0$*, and the limits for* $mp(\omega)$, $lv(\omega)$ *and* $hv(\omega)$ *exist. Thus* $\mathbb{E}^{\zeta}_{s}[lv] = 0$ *for all* ζ *and by Proposition 4 we get* $\mathbb{E}^{\zeta}_{s}[hv] = \mathbb{V}^{\zeta}_{s}[mp]$*. Hence we can use Example 1 to reason that both memory and randomization is needed to achieve the Pareto point* $(4, 2)$ *in Fig. 1.*

Now we prove the main theorem of this section.

Theorem 3. *If there is a strategy* ζ *satisfying* $(\mathbb{E}^{\zeta}_{s}[mp], \mathbb{E}^{\zeta}_{s_0}[hv]) \leq (u, v)$*, then there is a 2-memory strategy with the same properties. The problem whether such a strategy exists belongs to* **NP***, and approximation of the answer can be done in polynomial time. Moreover, Pareto optimal strategies always exist.*

We start by proving that 2-memory stochastic update strategies are sufficient for Pareto optimality wrt. hybrid variance.

Proposition 5. *Let* $s_0 \in S$ *and* $u, v \in \mathbb{R}$.

1) *If there is a strategy* ζ *satisfying* $(\mathbb{E}^{\zeta}_{s_0}[mp], \mathbb{E}^{\zeta}_{s_0}[hv]) \leq (u, v)$*, then the system* L_H *(Fig. 4) has a non-negative solution.*
2) *If there is a non-negative solution for the system* L_H *(Fig. 4), then there is a 2-memory stochastic-update strategy* σ *satisfying* $(\mathbb{E}^{\sigma}_{s_0}[mp], \mathbb{E}^{\sigma}_{s_0}[hv]) \leq (u, v)$.

Notice that we get the existence of Pareto optimal strategies as a side product of the above proposition, similarly to the case of global variance.

We briefly sketch the main ingredients for the proof of Proposition 5. We first establish the sufficiency of finite-memory strategies by showing that for an arbitrary strategy ζ, there is a 3-memory stochastic update strategy σ such that $(\mathbb{E}^{\sigma}_{s_0}[mp], \mathbb{E}^{\sigma}_{s_0}[hv]) \leq (\mathbb{E}^{\zeta}_{s_0}[mp], \mathbb{E}^{\zeta}_{s_0}[hv])$. The key idea of the proof of the construction of a 3-memory stochastic update strategy σ from an arbitrary strategy ζ is similar to the proof of Proposition 2. The details are in [3]. We then focus on finite-memory strategies. For a finite-memory strategy ζ, the frequencies are well-defined, and for an action $a \in A$, let $f(a) := \lim_{\ell\to\infty} \frac{1}{\ell} \sum_{t=0}^{\ell-1} \mathbb{P}^{\zeta}_{s_0}[A_t = a]$ denote the frequency of action a. We show that setting $x_a := f(a)$ for all $a \in A$ satisfies Eqns. (12), Eqns. (13) and Eqns. (14) of L_H. To obtain y_a and y_s, we define them in the same way as done in [2, Proposition 2] using the results of [10]. The details are in [3]. This completes the proof of the first item. The proof of the second item is as follows: the construction of a 2-memory stochastic update strategy σ from the constraints of the system L_H (other than constraint of Eqns 14) was presented in [2, Proposition 1]. The key argument to show that strategy σ also satisfies Eqns 14 is obtained by establishing that for the strategy σ we have: $\mathbb{E}^{\sigma}_{s}[hv] = \mathbb{E}^{\sigma}_{s}[mp_{r^2}] - \mathbb{E}^{\sigma}_{s}[mp]^2$ (here mp_{r^2} is the value of mp w.r.t. reward function defined by $r^2(a) = r(a)^2$; the equality is shown in [3]). It follows immediately that Eqns 14 is satisfied. This completes the proof of Proposition 5. Finally we show that for the quadratic program defined by the system L_H, the quadratic constraint satisfies the conditions of negative semi-definite programming with matrix of rank 1 (see [3]). Since negative semi-definite programs can be decided in NP [19] and with the additional restriction of rank 1 can be approximated in polynomial time [20], we get the complexity bounds of Theorem 3. Finally, Theorem 3 and Remark 1 give the following result.

$$\mathbf{1}_{s_0}(s) + \sum_{a\in A} y_a \cdot \delta(a)(s) = \sum_{a\in Act(s)} y_a + y_s \quad \text{for all } s \in S \tag{9}$$

$$\sum_{C\in MEC(G)} \sum_{s\in S\cap C} y_s = 1 \tag{10}$$

$$\sum_{s\in C} y_s = \sum_{a\in A\cap C} x_a \quad \text{for all } C \in MEC(G) \tag{11}$$

$$\sum_{a\in A} x_a \cdot \delta(a)(s) = \sum_{a\in Act(s)} x_a \quad \text{for all } s \in S \tag{12}$$

$$u \geq \sum_{a\in A} x_a \cdot r(a) \tag{13}$$

$$v \geq \sum_{a\in A} x_a \cdot r^2(a) - \Big(\sum_{a\in A} x_a \cdot r(a)\Big)^2 \tag{14}$$

Fig. 4. The system L_H. (Here $\mathbf{1}_{s_0}(s) = 1$ if $s = s_0$, and $\mathbf{1}_{s_0}(s) = 0$ otherwise.)

Corollary 3. *The approximate Pareto curve for hybrid variance can be computed in pseudo-polynomial time.*

VI. Zero variance with optimal performance

Now we present polynomial-time algorithms to compute the optimal expectation that can be ensured along with zero variance. The results are captured in the following theorem.

Theorem 4. *The minimal expectation that can be ensured*

1) *with zero hybrid variance can be computed in* $O((|S| \cdot |A|)^2)$ *time using discrete graph theoretic algorithms;*

[3]Note that Proposition 4 does *not* simplify the decision problem for hybrid variance, since it does not imply that the algorithms for global and local variance could be combined.

2) *with zero local variance can be computed in PTIME;*
3) *with zero global variance can be computed in PTIME.*

Hybrid variance. The algorithm for zero hybrid variance is as follows: (1) Order the rewards in an increasing sequence $\beta_1 < \beta_2 < \ldots < \beta_n$; (2) find the least i such that A_i is the set of actions with reward β_i and it can be ensured with probability 1 (almost-surely) that eventually only actions in A_i are visited, and output β_i; and (3) if no such i exists output "NO" (i.e., zero hybrid variance cannot be ensured). Since almost-sure winning for MDPs with eventually always property (i.e., eventualy only actions in A_i are visited) can be decided in quadratic time with discrete graph theoretic algorithm [6], [5], we obtain the first item of Theorem 4. The correctness is proved in [3].

Local variance. For zero local variance, we make use of the previous algorithm. The intuition is that to minimize the expectation with zero local variance, a strategy σ needs to reach states s in which zero hybrid variance can be ensured by strategies σ_s, and then mimic them. Moreover, σ minimizes the expected value of mp among all possible behaviours satisfying the above. The algorithm is as follows: (1) Use the algorithm for zero hybrid variance to compute a function β that assigns to every state s the minimal expectation value $\beta(s)$ that can be ensured along with zero hybrid variance when starting in s, and if zero hybrid variance cannot be ensured, then $\beta(s)$ is assigned $+\infty$. Let $M = 1 + \max_{s \in S} \beta(s)$. (2) Construct an MDP $\overline{G}$ as follows: For each state s such that $\beta(s) < \infty$ we add a state $\overline{s}$ with a self-loop on it, and we add a new action a_s that leads from s to $\overline{s}$. (3) Assign a reward $\beta(s) - M$ to a_s, and 0 to all other actions. Let $T = \{a_s \mid \beta(s) < \infty\}$ be the target set of actions. (4) Compute a strategy that minimizes the cumulative reward and ensures almost-sure (probability 1) reachability to T in $\overline{G}$. Let $\widehat{\beta}(s)$ denote the minimal expected payoff for the cumulative reward; and $\overline{\beta}(s) = \widehat{\beta}(s) + M$. In [3] we show that $\overline{\beta}(s)$ is the minimal expectation that can be ensured with zero local variance, and every step of the above computation can be achieved in polynomial time. This gives us the second item of Theorem 4.

Global variance. The basic intuition for zero global variance is that we need to find the minimal number y such that there is an almost-sure winning strategy to reach the MECs where expectation *exactly* y can be ensured with zero variance.

The algorithm works as follows: (1) Compute the MEC decomposition of the MDP and let the MECs be $C_1, C_2, \ldots, C_n$. (2) For every MEC C_i compute the minimal expectation $\alpha_{C_i} = \inf_\sigma \min_{s \in C_i} \mathbb{E}^\sigma_s[mp]$ and the maximal expectation $\beta_{C_i} = \sup_\sigma \max_{s \in C_i} \mathbb{E}^\sigma_s[mp]$ that can be ensured in the MDP induced by the MEC C_i. (3) Sort the values α_{C_i} in a non-decreasing order as $\ell_1 \leq \ell_2 \leq \ldots \leq \ell_n$. (4) Find the least i such that (a) $C_i = \{C_j \mid \alpha_{C_j} \leq \ell_i \leq \beta_{C_j}\}$ is the MEC's whose interval contains ℓ_i; (b) almost-sure (probability 1) reachability to the set $\bigcup_{C_j \in C_i} C_j$ (the union of the MECs in C_i) can be ensured; and output ℓ_i. (5) If no such i exists, then the answer to zero global variance is "NO" (i.e., zero global variance cannot be ensured). All the above steps can be computed in polynomial time. The correctness is proved in [3], and we obtain the last item of Theorem 4.

VII. Conclusion

We studied three notions of variance for MDPs with mean-payoff objectives: global (the standard one), local and hybrid variance. We established a strategy complexity (i.e., the memory and randomization required) for Pareto optimal strategies. For the zero variance problem, all the three cases are in PTIME. There are several interesting open questions. The most interesting open questions are whether the approximation problem for local variance can be solved in polynomial time, and what are the exact complexities of the strategy existence problem.

Acknowledgements. T. Brázdil is supported by the Czech Science Foundation, grant No P202/12/P612. K. Chatterjee is supported by the Austrian Science Fund (FWF) Grant No P 23499-N23; FWF NFN Grant No S11407-N23 (RiSE); ERC Start grant (279307: Graph Games); Microsoft faculty fellows award. V. Forejt is supported by a Royal Society Newton Fellowship and EPSRC project EP/J012564/1.

References

[1] E. Altman. *Constrained Markov Decision Processes (Stochastic Modeling)*. Chapman & Hall/CRC, 1999.
[2] T. Brázdil, V. Brožek, K. Chatterjee, V. Forejt, and A. Kučera. Two views on multiple mean-payoff objectives in Markov decision processes. In *Proceedings of LICS 2011*. IEEE, 2011.
[3] T. Brázdil, K. Chatterjee, V. Forejt, and A. Kučera. Trading performance for stability in Markov decision processes. Available at arXiv.org, 2013.
[4] J. Canny. Some algebraic and geometric computations in PSPACE. In *Proceedings of STOC'88*, pages 460–467. ACM Press, 1988.
[5] K. Chatterjee and M. Henzinger. Faster and dynamic algorithms for maximal end-component decomposition and related graph problems in probabilistic verification. In *SODA*, pages 1318–1336. SIAM, 2011.
[6] K. Chatterjee and M. Henzinger. An $O(n^2)$ time algorithm for alternating Büchi games. In *SODA*, pages 1386–1399. SIAM, 2012.
[7] K. Chatterjee, R. Majumdar, and T. Henzinger. Markov decision processes with multiple objectives. In *Proceedings of STACS 2006*, volume 3884 of *LNCS*, pages 325–336. Springer, 2006.
[8] K-J. Chung. Mean-variance tradeoffs in an undiscounted MDP: The unichain case. *Operations Research*, 42:184–188, 1994.
[9] C. Courcoubetis and M. Yannakakis. Markov decision processes and regular events. *IEEE Transactions on Automatic Control*, 43(10):1399–1418, 1998.
[10] K. Etessami, M. Kwiatkowska, M. Vardi, and M. Yannakakis. Multi-objective model checking of Markov decision processes. *Logical Methods in Computer Science*, 4(4):1–21, 2008.
[11] J. A. Filar, L.C.M. Kallenberg, and H-M. Lee. Variance-penalize Markov decision processes. *Math. of Oper. Research*, 14:147–161, 1989.
[12] V. Forejt, M. Kwiatkowska, and D. Parker. Pareto curves for probabilistic model checking. In *Proc. of ATVA'12*, volume 7561 of *LNCS*, pages 317–332. Springer, 2012.
[13] S. Mannor and J. Tsitsiklis. Mean-variance optimization in Markov decision processes. In *Proceedings of ICML-11*, pages 177–184, New York, NY, USA, June 2011. ACM.
[14] J.R. Norris. *Markov Chains*. Cambridge University Press, 1998.
[15] M.L. Puterman. *Markov Decision Processes*. Wiley, 1994.
[16] H. L. Royden. *Real analysis*. Macmillan, New York, 3rd edition, 1988.
[17] M. J. Sobel. The variance of discounted MDP's. *Journal of Applied Probability*, 19:794–802, 1982.
[18] M. J. Sobel. Mean-variance tradeoffs in an undiscounted MDP. *Operations Research*, 42:175–183, 1994.
[19] S. A. Vavasis. Quadratic programming is in NP. *Information Processing Letters*, 36(2):73 – 77, 1990.
[20] S. A. Vavasis. Approximation algorithms for indefinite quadratic programming. *Math. Program.*, 57(2):279–311, November 1992.