MultiGain: A controller synthesis tool for MDPs with multiple mean-payoff objectives

Tomáš Brázdil[1], Krishnendu Chatterjee[2], Vojtěch Forejt[3], and Antonín Kučera[1]

[1] Faculty of Informatics, Masaryk University, Brno, Czech Republic
[2] IST Austria
[3] Department of Computer Science, University of Oxford, UK

Abstract. We present MultiGain, a tool to synthesize strategies for Markov decision processes (MDPs) with multiple mean-payoff objectives. Our models are described in PRISM, and our tool uses the existing interface and simulator of PRISM. Our tool extends PRISM by adding novel algorithms for multiple mean-payoff objectives, and also provides features such as (i) generating strategies and exploring them for simulation, and checking them with respect to other properties; and (ii) generating an approximate Pareto curve for two mean-payoff objectives. In addition, we present a new practical algorithm for the analysis of MDPs with multiple mean-payoff objectives under memoryless strategies.

1 Introduction

Markov decision processes (MDPs) are a standard model for analysis of probabilistic systems with non-determinism [12], with a wide range of applications [5]. In each state of an MDP, a controller chooses one of several actions (the nondeterministic choices), and the current state and action gives a probability distribution over the successor states. One classical objective used to study quantitative properties of systems is the *limit-average (or mean-payoff)* objective, where a reward (or cost) is associated with each transition and the objective assigns to every run the average of the rewards over the run. MDPs with single mean-payoff objectives have been well studied in the literature (see, e.g., [14]). However, in many modeling domains, there is not a single goal to be optimized, but multiple, potentially interdependent and conflicting goals. For example, in designing a computer system, the goal is to maximize average performance while minimizing average power consumption. Similarly, in an inventory management system, the goal is to optimize several dependent costs for maintaining each kind of product. The complexity of MDPs with multiple mean-payoff objectives was studied in [6].

In this paper we present MultiGain, which is, to the best of our knowledge, the first tool for synthesis of controller strategies in MDPs with multiple mean-payoff objectives. The MDPs and the mean-payoff objectives are specified in the well-known PRISM modelling language. Our contributions are as follows: (1) we extend PRISM with novel algorithms for multiple mean-payoff objectives from [6]; (2) develop on the results of [6] to synthesize strategies, and explore them for simulation, and check them with respect to other properties (as done in PRISM-games [9]); and (3) for the important special case of two mean-payoff objectives we provide the feature to visualize the approximate Pareto curve (where the Pareto curve represents the "trade-off" curve and consists of solutions that are not strictly dominated

by any other solution). Finally, we present a new practical approach for analysis of MDPs with multiple mean-payoff objectives under memoryless strategies: previously an NP bound was shown in [8] by guessing all bottom strongly connected components (BSCCs) of the MDP graph for a memoryless strategy and this gave an exponential enumerative algorithm; in contrast, we present a linear reduction to solving a boolean combination of linear constraints (which is a special class of mixed integer linear programming where the integer variables are binary).

2 Definitions

MDPs and strategies. An MDP $G = (S, A, Act, \delta)$ consists of (i) a *finite* set S of states; (ii) a *finite* set A of actions, (iii) an action enabledness function $Act : S \to 2^A \setminus \{\emptyset\}$ that assigns to each state s the set $Act(s)$ of actions enabled at s, and (iv) a transition function $\delta : S \times A \to dist(S)$ that given a state s and an action $a \in Act(s)$ gives a probability distribution over the successor states ($dist(S)$ denotes all probability distributions over S). W.l.o.g. we assume that every action is enabled in exactly one state, and we denote this state $Src(a)$. Thus, we will assume that $\delta : A \to dist(S)$. *Strategies* describe how to choose the next action given a finite path (of state and action pairs) in the MDP. A strategy consists of a set of memory elements to remember the history of the paths. The memory elements are updated stochastically in each transition, and the next action is chosen probabilistically (among enabled actions) based on the current state and current memory [6]. A strategy is *memoryless* if it depends only on the current state.

Multiple mean-payoff objectives. A single mean-payoff objective consists of a reward function r that assigns a real-valued reward $r(s, a)$ to every state s and action a enabled in s, and the mean-payoff objective $\mathsf{mp}(r)$ assigns to every infinite path (or run) the long-run average of the rewards of the path, i.e., for $\pi = (s_0 a_0 s_1 a_1 \ldots)$ we have $\mathsf{mp}(r)(\pi) = \liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} r(s_i, a_i)$. In multiple mean-payoff objectives, there are k reward functions $r_1, r_2, \ldots, r_k$, and each reward function r_i defines the respective mean-payoff objective $\mathsf{mp}(r_i)$. Given a strategy σ and a random variable X, we denote by $\mathbb{E}_s^\sigma[X]$ the expectation of the σ w.r.t. X, given a starting state s. Thus for a mean-payoff objective $\mathsf{mp}(r)$, the expected mean-payoff is $\mathbb{E}_s^\sigma[\mathsf{mp}(r)]$.

Synthesis questions. The relevant questions in analysis of MDPs with multiple objectives are as follows: (1) *(Existence).* Given an MDP with k reward functions, starting state s_0, and a vector $\boldsymbol{v} = (v_1, v_2, \ldots, v_k)$ of k real-values, the existence question asks whether there exists a strategy σ such that for all $1 \le i \le k$ we have $\mathbb{E}_{s_0}^\sigma[\mathsf{mp}(r_i)] \ge v_i$. (2) *(Synthesis).* If the answer to the existence question is yes, the synthesis question asks for a witness strategy to satisfy the existence question. An optimization question related to multiple objectives is the computation of the Pareto-curve (or the trade-off curve), where the Pareto curve consists of vectors $\boldsymbol{v}$ such that the answer to the existence question is yes, and for all vectors $\boldsymbol{v}'$ that strictly dominate $\boldsymbol{v}$ (i.e., $\boldsymbol{v}'$ is at least $\boldsymbol{v}$ in all dimensions and strictly greater in at least one dimension) the answer to the existence question is no.

3 Algorithms and Implementation

We first recall the existing results for MDPs with multiple mean-payoff objectives [6], and then describe our implementation and extensions. Before presenting the existing results, we first recall the notion of maximal end-components in MDPs.

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

$$\sum_{s \in S_{MEC}} y_s = 1 \qquad (2)$$

$$\sum_{s \in C} y_s = \sum_{a \in A \cap C} x_a \qquad \text{for all MECs } C \text{ of } G \qquad (3)$$

$$\sum_{a \in A} x_a \cdot \delta(a)(s) = \sum_{a \in Act(s)} x_a \qquad \text{for all } s \in S \qquad (4)$$

$$\sum_{a \in A} x_a \cdot \boldsymbol{r}_i(a) \geq \boldsymbol{v}_i \qquad \text{for all } 1 \leq i \leq k \qquad (5)$$

Fig. 1: System L of linear inequalities (here $\mathbf{1}_{s_0}(s)$ is 1 if $s{=}s_0$, and 0 otherwise).

Maximal end-components. 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 $a \in B$, whenever $\delta(a)(s') > 0$ then $s' \in T$; and (2) for all $s, t \in T$ there is a finite path from s to t such that all states and actions that appear in the path 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. An MDP is *unichain* if for all $B \subseteq A$ satisfying $B \cap Act(s) \neq \emptyset$ for any $s \in S$ we have that (S, B) is a MEC. Given an MDP, we denote S_{MEC} the set of states s that are contained within a MEC.

Result from [6]. The results of [6] showed that (i) the existence question can be answered in polynomial time, by reduction to linear programming; (ii) if there exists a strategy for the existence problem, then there exists a witness strategy with only two-memory states. It also established that if the MDP is unichain, then memoryless strategies are sufficient. The polynomial-time algorithm is as follows: it was shown in [6] that the answer to the existence problem is yes iff there exists a non-negative solution to the system of linear inequalities given in Fig. 1.

Syntax and semantics. Our tool accepts PRISM MDP models as input, see [1] for details. The multi-objective properties are expressed as `multi(`*list*`)` or `mlessmulti(`*list*`)` where *list* is a comma separated list of mean-payoff reward properties, which can be *boolean*, e.g. `R{'r1'}>=0.5 [S]`, and in the case of `multi` also *numerical*, e.g. `R{'r2'}min=? [S]`. In the reward properties, `S` stands for steady-state, following PRISM's terminology.

If all properties in the list are boolean, the multi-objective property `multi(`*list*`)` is also boolean and is true iff there is a strategy under which all given reward properties in the list are simultaneously satisfied. If there is a single numerical query, the multi-objective query intuitively asks for the maximal achievable reward of the numerical reward query, subject to the restriction given by the boolean queries. We also allow two numerical queries; in such case MultiGain generates a Pareto curve. The semantics of `mlessmulti` follows the same pattern, the only difference being that only memoryless (randomised) strategies are being considered. The reason we don't allow numerical reward properties in `mlessmulti` is that the supremum among all memoryless strategies might not be realised.

Implementation of existence question. We have implemented the algorithm of [6]. Our implementation takes as input an MDP with multiple mean-payoff objectives and a value vector $\boldsymbol{v}$, and computes the linear inequalities of Fig. 1 or a *mixed integer linear programming* (MILP) extension in case of memoryless strategies. The system of linear inequalities is solved with LPsolve [2] or Gurobi [3].

Implementation of the synthesis question. We now describe how to obtain witness strategies. Assume that the linear program from Fig. 1 has a solution, where a solution to a variable z is denoted by $\overline{z}$. We construct a new linear program, comprising Eq. 1 together with the equations $y_s = \sum_{a \in Act(s)} \overline{x}_a$ for all $s \in S_{MEC}$.

Let $\widehat{z}$ denote a solution to variables z in this linear program. The stochastic-update strategy is defined to have 2 memory states ("transient" and "recurrent"), with the transition function defined to be $\sigma_t(s)(a) = \widehat{y}_a / \sum_{b \in Act(s)} \widehat{y}_b$ and $\sigma_r(s)(a) = \overline{x}_a / \sum_{b \in Act(s)} \overline{x}_b$, and the probability of switching from "transient" to "recurrent" state upon entering s being $\widehat{y}_s / (\sum_{a \in Act(s)} \widehat{y}_a + \widehat{y}_s)$. The correctness of the witness construction follows from [6].

MILP for memoryless strategies. For memoryless strategies, the current upper bound is NP [8] and the previous algorithm enumerates all possible BSCCs under a memoryless strategy. We present a polynomial-time reduction to solving a boolean combination of linear constraints, that can be easily encoded using MILP with binary variables [16]. The key requirement for memoryless strategies is that a state can either be recurrent or transient. For the existence question restricted to memoryless strategies we modify the linear constraints from Fig. 1 as follows: (i) we add constraints; for all states s and actions $b \in Act(s)$: $y_b > 0 \implies (x_b > 0 \vee \sum_{a \in Act(Src(b))} x_a = 0)$; (ii) we replace constraint (3) from Fig. 1 by constraints that for all states s: $y_s = \sum_{a \in Act(s)} x_a$. The constraint (ii) is a strengthening of constraint (3), as the above constraint implies constraint (3). Further details are in [7].

Approximate Pareto curve for two objectives. To generate a Pareto curve, we successively compute solutions to several linear programs for a single mean-payoff objective, where every time the objective is obtained as a weighted sum of the objectives for which the Pareto curve is generated. The weights are selected in a way similar to [11], allowing us to obtain the approximation of the curve.

Unlike the PRISM implementation for multi-objective cumulative rewards, our tool is able to generate the Pareto curve for objectives of the form `multi(R{'r1'}max=?[S], R{'r2'}max=? [S], R{'r3'}>=0.5 [S])` where the objectives to be optimised are subject to restrictions given by other rewards.

Features of our tool. In summary, our tool extends PRISM by developing algorithms to solve MDPs with multiple mean-payoff objectives. Along with the algorithm from [6] we have also implemented a visual representation of the Pareto curve for two-dimensional objectives. The implementation utilises a multi-objective visualisation available in PRISM for cumulative reward and LTL objectives.

In addition, we adapted a feature from PRISM-games [9] which allows the user to generate strategies, so that they can be explored and investigated by simulation. A product (Markov chain) of an MDP and a strategy can be constructed, allowing the user to employ it for verification of other properties.

The tool is available at http://qav.cs.ox.ac.uk/multigain/, and the source code is provided under GPL. For licencing reasons, Gurobi is not included with the download, but it can be added manually by following provided steps.

4 Experimental Results: Case Studies

We have evaluated our tool on two standard case studies, adapted from [1], and also mention other applications where our tool could be used.

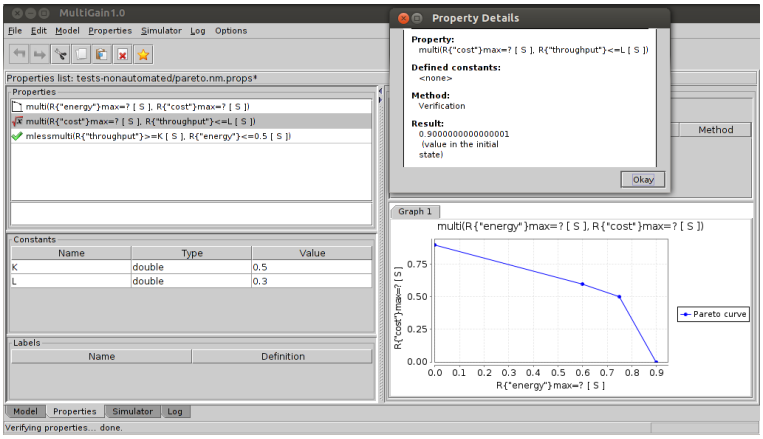

Fig. 2: Screenshot of MultiGain (largely inheriting from the PRISM GUI).

Dining philosophers is a case study based on the algorithm of [10], which extends Lehmann and Rabin's randomised solution [13] to the dining philosophers problem so that there is no requirement for fairness assumptions. The constant N gives the number of philosophers. We use two reward structures, `think` and `eat` for the number of philosophers currently thinking and eating, respectively.

Randomised Mutual Exclusion models a solution to the mutual exclusion problem by [15]. The parameter N gives the number of processes competing for the access to the critical section. Here we defined reward structures `try` and `crit` for the number of processes that are currently trying to access the critical section, and those which are in it, currently (the latter number obviously never being more than 1).

Evaluation The statistics for some of our experiments are given in Table 1 (the complete results are available from the tool's website). The experiments were run on a 2.66GHz PC with 4GB RAM, the LP solver used was Gurobi and the timeout ("t/o") was set to 2 hours. We observed that our approach scales to mid-size models, the main limitation being the LP solver.

model	para.	property (A: `multi(...)`, B: `mlessmulti(...)`)	MDP states	LP vars (binary)	LP rows	total time (s)	solving time (s)	value
phil	3	A: `R{"think"}max=?,R{"eat"}>=0.3`	956	6344	1915	0.23	0.08	2.119
	3	B: `R{"think"}>=2.11,R{"eat"}>=0.3`	956	12553 (6344)	11773	209.9	209.7	true
	3	B: `R{"think"}>=2.12,R{"eat"}>=0.3`	956	12553 (6344)	11773	20.9	20.7	false
	4	A: `R{"think"}max=?,R{"eat"}>=1`	9440	80368	18883	4.4	3.8	2.429
	5	A: `R{"think"}max=?,R{"eat"}>=1`	93068	967168	186139	616.0	606.4	3.429
mutex	3	A: `R{"try"}max=? [S],R{"crit"}>=0.2`	27766	119038	55535	214.9	212.7	2.679
	4	A: `R{"try"}max=? [S],R{"crit"}>=0.3`	668836	3010308	1337675	*t/o*	*t/o*	*t/o*
	4	A: `R{"try"}>=3.5 [S],R{"crit"}>=0.3`	668836	3010308	1337676	4126	4073	true

Table 1: Experimental results. For space reasons, the `[S]` argument to `R` is omitted.

Other applications. We mention two applications which are solved using MDPs with multiple mean-payoff objectives. (A) The problem of synthesis from incompatible specifications was considered in [17]. Given a set of specifications $\varphi_1, \varphi_2, \ldots, \varphi_k$ that cannot be all satisfied together, the goal is to synthesize a system such that

for all $1 \leq i \leq k$ the distance to specification φ_i is at most v_i. In adversarial environments the problem reduces to games and for probabilistic environments to MDPs, with multiple mean-payoff objectives [17]. (B) The problem of synthesis of steady state distributions for ergodic MDPs was considered in [4]. The problem can be modeled with multiple mean-payoff objectives by considering indicator reward functions r_s, for each state s, that assign reward 1 to every action enabled in s and 0 to all other actions. The steady state distribution synthesis question of [4] then reduces to the existence question for multiple mean-payoff MDPs.

Concluding remarks. We presented the first tool for analysis of MDPs with multiple mean-payoff objectives. The limiting factor is the LP solver, and so an interesting direction would be to extend the results of [18] to multiple objectives.

Acknowledgements The authors were in part supported by Austrian Science Fund (FWF) Grant No P23499- N23, FWF NFN Grant No S11407-N23 (RiSE), ERC Start grant (279307: Graph Games) and the research centre Institute for Theoretical Computer Science (ITI), grant No. P202/12/G061.

References

1. http://www.prismmodelchecker.org/.
2. http://sourceforge.net/projects/lpsolve/.
3. http://www.gurobi.com.
4. S. Akshay, N. Bertrand, S. Haddad, and L. Hélouët. The steady-state control problem for Markov decision processes. In *QEST*, pages 290–304, 2013.
5. C. Baier and J.-P. Katoen. *Principles of model checking*. MIT Press, 2008.
6. T. Brázdil, V. Brožek, K. Chatterjee, V. Forejt, and A. Kučera. Two views on multiple mean-payoff objectives in Markov decision processes. In *LICS 2011*, pages 33–42. IEEE Computer Society, 2011.
7. T. Brázdil, K. Chatterjee, V. Forejt, and A. Kučera. MultiGain: A controller synthesis tool for mdps with multiple mean-payoff objectives. *CoRR*, abs/1501.03093, 2015.
8. K. Chatterjee. Markov decision processes with multiple long-run average objectives. In *FSTTCS*, pages 473–484, 2007.
9. T. Chen, V. Forejt, M. Kwiatkowska, D. Parker, and A. Simaitis. PRISM-games: A model checker for stochastic multi-player games. In *TACAS'13*, volume 7795 of *LNCS*. Springer, 2013.
10. M. Duflot, L. Fribourg, and C. Picaronny. Randomized dining philosophers without fairness assumption. *Distributed Computing*, 17(1):65–76, 2004.
11. V. Forejt, M. Kwiatkowska, and D. Parker. Pareto curves for probabilistic model checking. In *ATVA'12*, volume 7561 of *LNCS*, pages 317–332. Springer, 2012.
12. R. A. Howard. *Dynamic Programming and Markov Processes*. MIT Press, 1960.
13. D. Lehmann and M. Rabin. On the advantage of free choice: A symmetric and fully distributed solution to the dining philosophers problem. In *POPL'81*, 1981.
14. M. L. Puterman. *Markov Decision Processes*. J. Wiley and Sons, 1994.
15. M. Rabin. N-process mutual exclusion with bounded waiting by $4 \log_2 N$-valued shared variable. *Journal of Computer and System Sciences*, 25(1):66–75, 1982.
16. A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, 1998.
17. P. Černý, S. Gopi, T. A. Henzinger, A. Radhakrishna, and N. Totla. Synthesis from incompatible specifications. In *EMSOFT*, pages 53–62, 2012.
18. R. Wimmer, B. Braitling, B. Becker, E. M. Hahn, P. Crouzen, H. Hermanns, A. Dhama, and O. E. Theel. Symblicit calculation of long-run averages for concurrent probabilistic systems. In *QEST*, pages 27–36. IEEE Computer Society, 2010.