Long-run Average Reward for Markov Decision Processes

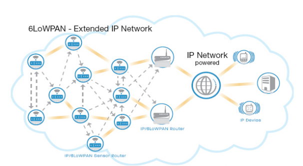
Based on a paper at CAV 2017

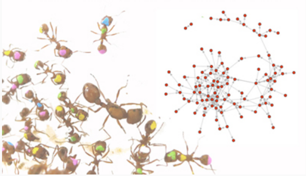
Pranav Ashok[1], Krishnendu Chatterjee[2], Przemysław Daca[2],
Jan Křetínský[1] and Tobias Meggendorfer[1]

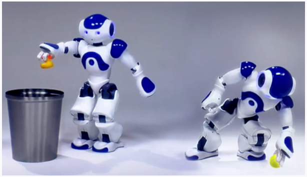
August 9, 2017

[1]Technical University of Munich, Germany
[2]IST Austria

Motivation







Markov Decision Processes (MDPs) are a standard model for describing systems which display probabilistic as well as non-deterministic behaviour.

Motivation

Open Problem

Long-run Average Reward or **Mean-payoff** using Value Iteration (VI)

- VI approaches exist for subclasses, but not for general MDPs
- Other existing approaches: Linear Programming (LP) and Strategy Iteration (SI)[1]

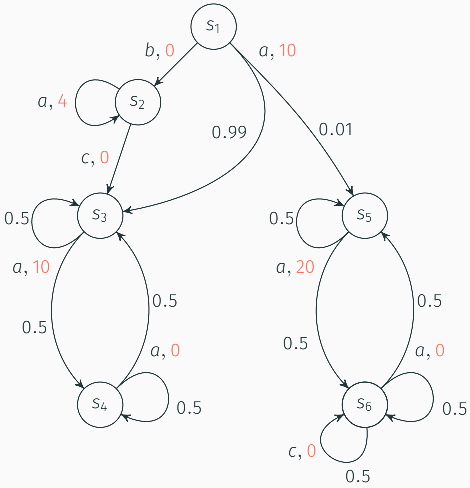
[1]Martin L. Puterman, Markov Decision Processes, 1994.

Contributions

- Disprove conjectured stopping criterion for VI[2]
- General solution using VI
- Improve performance using ideas from Machine Learning

[2] Not covered in this talk

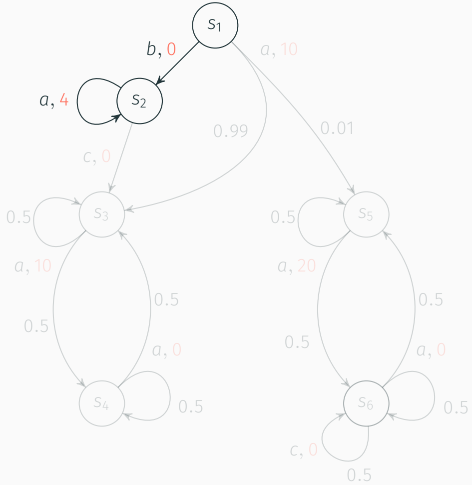
Markov Decision Process (MDPs)



Strategy

A strategy (or policy) gives the action to be taken at every state.

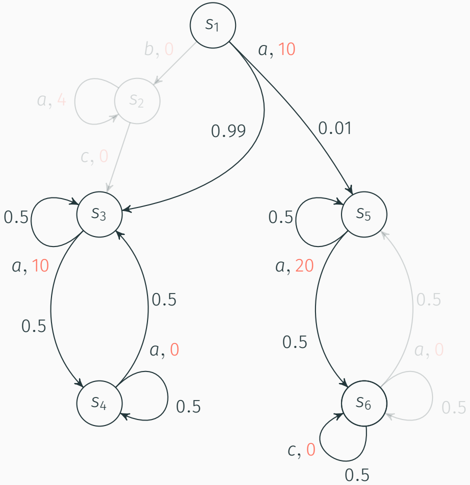
$$\pi := \{s_1 \mapsto b, s_2 \mapsto a\}$$



Strategy

A strategy (or policy) gives the action to be taken at every state.

$$\pi := \{s_1 \mapsto a, \ldots, s_6 \mapsto a\}$$



Mean-payoff or Long-run Average Reward

$$\rho = 15 \quad 35 \quad 20 \quad 50 \quad 0 \quad 10 \quad 0 \quad 10 \quad 0 \quad 10 \quad 0 \ldots$$

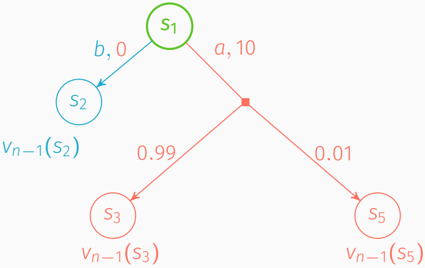
Then, n-step average reward is given by

$$\boxed{MP_n(\rho)} \;:=\; \frac{1}{n} \cdot \sum_{i=1}^{n} \rho_i$$

$$MP \;:=\; \sup_{\pi} \liminf_{n \to \infty} \mathbb{E}^{\pi}[\,\boxed{MP_n(\rho)}\,]$$

VI for Total Rewards (Bellman Equation)

$$v_n(s_1) = max\{\ 0 + v_{n-1}(s_2)\ ,\ 10 + (\ 0.99 \cdot v_{n-1}(s_3) + 0.01 \cdot v_{n-1}(s_5)\ )\ \}$$



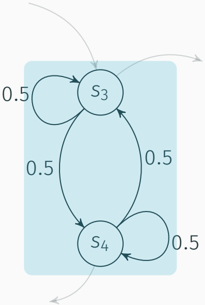
Value Iteration

Average reward[3]

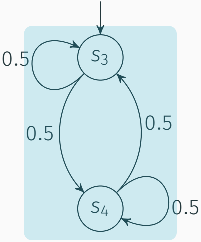
$$\lim_{n\to\infty} \frac{v_n(s)}{n} \approx v_n(s) - v_{n-1}(s)$$

[3] After making the MDP aperiodic

Towards a General VI: Maximal End Components (MECs)



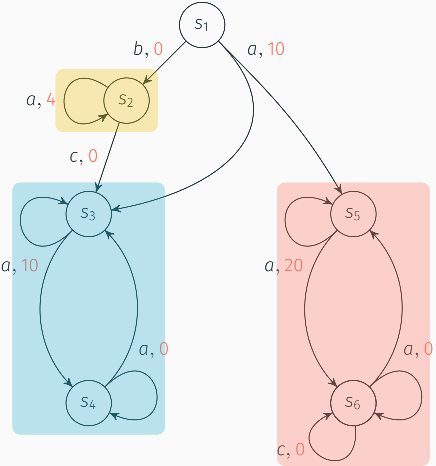
An example MEC



Communicating MDP

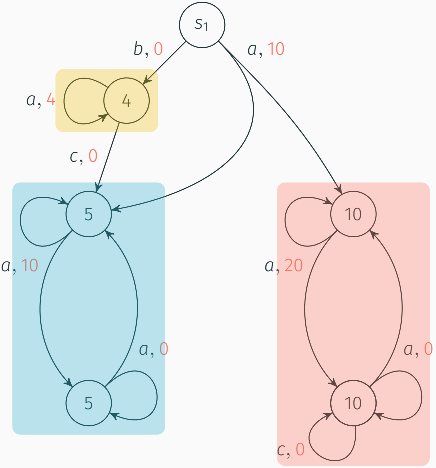
VI ✓

Towards a General VI: Step 1 – MEC-Decomposition



Find all MECs

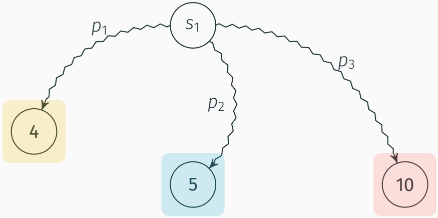
Towards a General VI: Step 1 – MEC-Decomposition



Find all MECs and run VI on them until ε-convergence

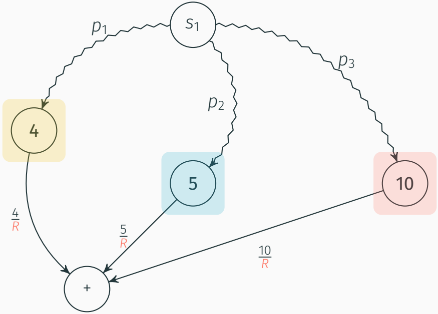
Towards a General VI: Step 2 – Weighted Reachability

$$\text{Max. mean-payoff} = \sup_{\pi} p_1 \cdot 4 + p_2 \cdot 5 + p_3 \cdot 10$$



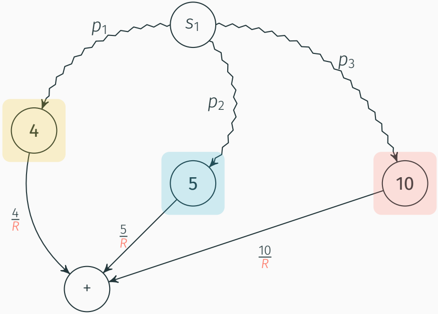
Towards a General VI: Step 2 – Weighted Reachability

$$\frac{\text{Max. mean-payoff}}{R} = \sup_{\pi} p_1 \frac{4}{R} + p_2 \frac{5}{R} + p_3 \frac{10}{R}$$



Towards a General VI: Step 2 – Weighted Reachability

$$\frac{\text{Max. mean-payoff}}{R} = \sup_{\pi} p_1 \frac{4}{R} + p_2 \frac{5}{R} + p_3 \frac{10}{R}$$



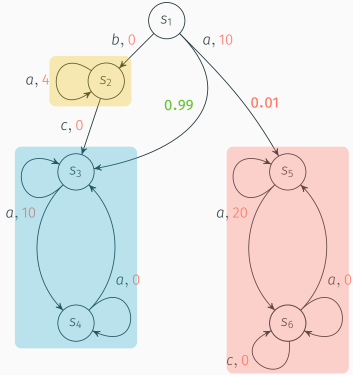
Mean-payoff reduced to reachability: $P_{max}(\Diamond +)$

Limitations of this method

- Whole state space is being explored
- Finding MECs on large state spaces is computationally expensive

Improvement: avoid full state-space exploration

Idea: let sampling guide us to the "important" regions



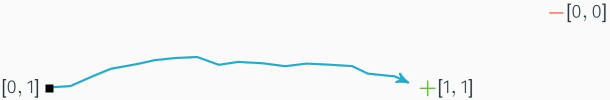
Contribution of red region to MP is potentially low

Improvement: guarantees through sampling

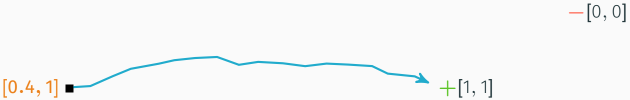
- Existing: BRTDP approach for reachability[4,5]
- Repeatedly samples paths from initial state
- Back-propagates values along the path using VI operator

[4] Bounded Real-Time Dynamic Programming, McMahan et. al., ICML ’05
[5] Verification of Markov Decision Processes using Learning Algorithms, Brazdil et. al., ATVA ’14

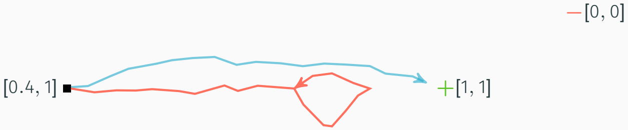
Improvement: BRTDP in action



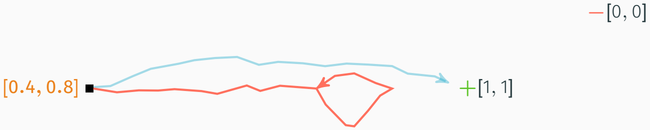
Improvement: BRTDP in action



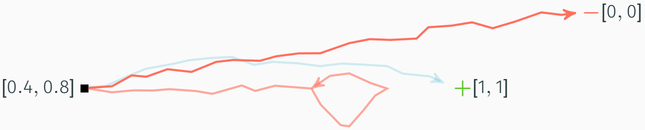
Improvement: BRTDP in action



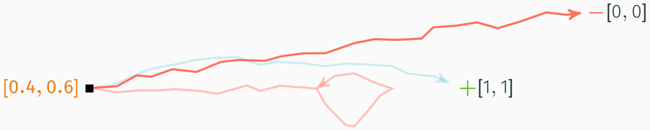
Improvement: BRTDP in action



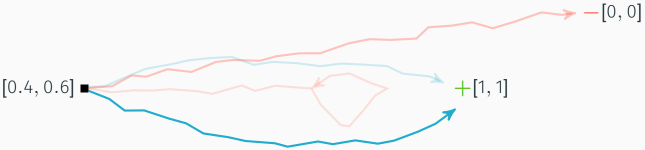
Improvement: BRTDP in action



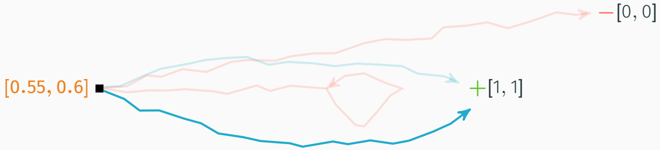
Improvement: BRTDP in action



Improvement: BRTDP in action



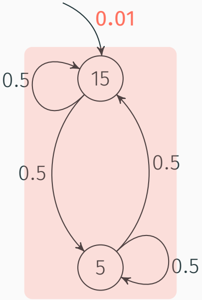
Improvement: BRTDP in action



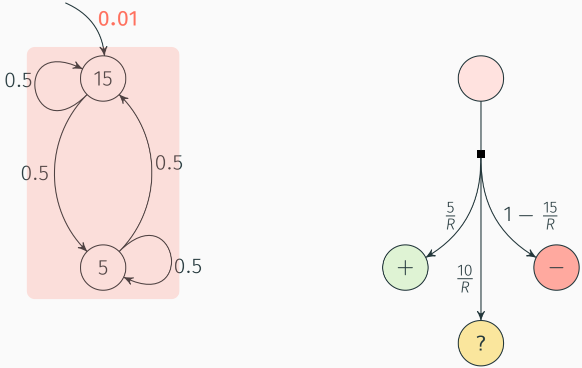
Improvement: Algorithm

1. Run BRTDP in search of the $+$ state
2. Repeatedly collapse MECs amongst the states explored so far
3. While collapsing, compute their values and add special transition to **+** and **–** states

Improvement: can do even more...



Improvement: can do even more...



Collapse MEC into a single state and add a special action

Final Algorithm: On-demand Value Iteration (ODV)

1. Sample paths like in BRTDP
2. When MECs are detected, collapse them but...
3. Don't compute MEC value until ε-convergence
4. Add transition with probability $\propto$ (U-L) to **?** state
5. Refine value of MEC only when **?** encountered

Summary

We saw two methods which can be used to obtain mean-payoff with guarantees

1. Collapse MECs, add transitions to +/− states, run reachability
2. ODV: Run sampling, collapse on-the-go, refine MEC values on-demand

Benchmarks

Model	States	MECs	LP[1]	MEC-VI[2]
virus	809	1	0.19	**0.05**
cs_nfail4	960	176	0.7	**0.18**
investor	6 688	837	2.8	**0.51**
phil-nofair5	93 068	1	TO	**6.67**
rabin4	668 836	1	TO	**112.38**

1. MultiGain, Brazdil et. al. 2015.
2. MEC-VI: Straightforward conversion to reachability, then VI

Benchmarks

On-demand VI better by orders of magnitude depending on topology

`Model`	States	`MEC-VI`	`ODV`	`ODV` States	`ODV` MECs
zeroconf(40,10)	3 001 911	MO	5.05	481	3
avoid				582	3
zeroconf(300,15)	4 730 203	MO	16.6	873	3
avoid				5 434	3
sensors(2)	7 860	18.9	20.1	3 281	917
sensors(3)	77 766	2293.0	37.0	10 941	2 301