IST AUSTRIA
Institute of Science and Technology

Multiple-Environment Markov Decision Processes: Efficient Analysis and Applications

ICAPS 2020

K. Chatterjee, M. Chmelík, D. Karkhanis, P. Novotný, A. Royer

October 27-30th, 2020

Introducing MEMDPS

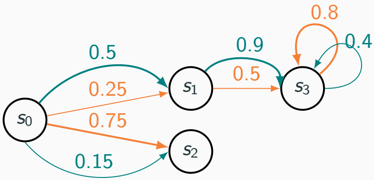


Figure: A MEMDP augments the standard MDP framework with the notion of *environments* or *contexts*

[1]Multiple-Environment Markov Decision Processes, J.F. Raskin and O. Sancur, 2014

Introducing MEMDPS

Definition[1]

Formally, a MEMDP is a tuple $(\mathcal{I}, \mathcal{S}, \mathcal{A}, \delta, r, s_0, \lambda)$, where:

- $\mathcal{S}$, is a finite set of control states;
- $\mathcal{A}$, is a finite alphabet of actions;
- $\mathcal{I}$, is a finite set of environments;
- $\{\delta_i\}_{i \in \mathcal{I}}$, is a collection of probabilistic transition functions, one for every environment
- $\{r_i\}_{i \in \mathcal{I}}$, is a set of reward functions
- $s_0 \in \mathcal{S}$, is the initial state;
- $\lambda \in \mathcal{D}(\mathcal{I})$, is the initial distribution over the environments

[1]Multiple-Environment Markov Decision Processes, J.F. Raskin and O. Sancur, 2014

Introducing MEMDPS

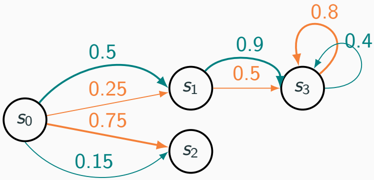
Definition[1]

Formally, a MEMDP is a tuple $(\mathcal{I}, \mathcal{S}, \mathcal{A}, \delta, r, s_0, \lambda)$, where:

- $\mathcal{S}$, is a finite set of control states;
- $\mathcal{A}$, is a finite alphabet of actions;
- $\mathcal{I}$, is a finite set of environments;
- $\{\delta_i\}_{i \in \mathcal{I}}$, is a collection of probabilistic transition functions, one for every environment
- $\{r_i\}_{i \in \mathcal{I}}$, is a set of reward functions
- $s_0 \in \mathcal{S}$, is the initial state;
- $\lambda \in \mathcal{D}(\mathcal{I})$, is the initial distribution over the environments

[1]Multiple-Environment Markov Decision Processes, J.F. Raskin and O. Sancur, 2014

Introducing MEMDPS



In summary, MEMDPs augment MDPs with *multiple environment hypotheses*, aiming to design a controller that perform well for all. Previous work[1] study the existence of winning and almost winning strategies in MEMDPs.

[1]Multiple-Environment Markov Decision Processes, J.F. Raskin and O. Sancur, 2014

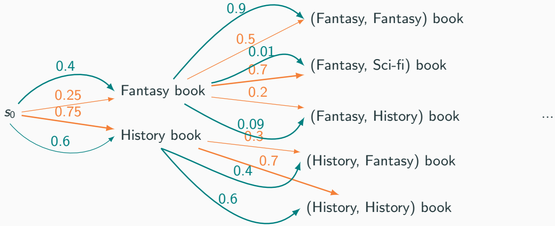
Applications

Orthogonal to this, in this work, we explore the *practicality* of MEMDPS across different settings and applications.

Orthogonal to this, in this work, we explore the *practicality* of MEMDPS across different settings and applications.

Example: Recommendation systems as MEMDPs

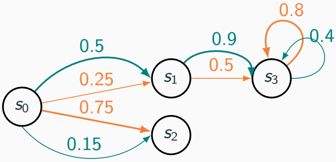
A MEMDP can be used to build a MDP-based recommender which is tailored to different user profiles (environments), with potentially different transition functions.



A subcase of POMDPS

MEMDPs are POMDPs

Every MEMDP can be formulated as a partially-observable MDP, by considering the cross-product of states and environments



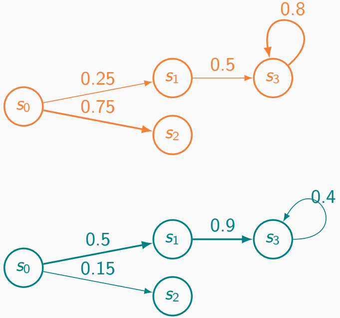


Figure: Converting a MEMDP (left) to a POMDP (right)

A subcase of POMDPS

MEMDPs are POMDPs

Every MEMDP can be formulated as a partially-observable MDP, by considering the cross-product of states and environments

Consequently, POMDP solvers can be readily applied to the MEMDP framework. However, we show that developping MEMDP-specific solvers can significantly improve performance.

Solving MEMDPS: A summary

Sparse transition function

The partially-observable (PO) feature (the environment $\mathcal{I}$) is sampled only once, at initialization, and then kept constant. Thus there is no transitions across environments, and we can store the transition function more efficiently.

Solving MEMDPS: A summary

Sparse transition function

$\implies$ Memory usage: $O(|\mathcal{S}|^2|\mathcal{I}||\mathcal{A}|)$ (instead of $O(|\mathcal{S}|^2|\mathcal{I}|^2|\mathcal{A}|)$)

Solving MEMDPS: A summary

Sparse transition function

$\implies$ Memory usage: $O(|\mathcal{S}|^2|\mathcal{I}||\mathcal{A}|)$ (instead of $O(|\mathcal{S}|^2|\mathcal{I}|^2|\mathcal{A}|)$)

Faster belief updates

In a MEMDP, the uncertainty lies on the environment, rather than on states. Furthermore, as noted before, the PO features are static, once sampled.

Solving MEMDPS: A summary

Sparse transition function

$\implies$ Memory usage: $O(|\mathcal{S}|^2|\mathcal{I}||\mathcal{A}|)$ (instead of $O(|\mathcal{S}|^2|\mathcal{I}|^2|\mathcal{A}|)$))

Faster belief updates

$\implies$ Belief update can be done linearly in $O(|\mathcal{I}|)$ (rather than quadratic in terms of states $O(|\mathcal{S}|^2|\mathcal{I}|^2)$)

Solving MEMDPS: A summary

Sparse transition function

$\implies$ Memory usage: $O(|\mathcal{S}|^2|\mathcal{I}||\mathcal{A}|)$ (instead of $O(|\mathcal{S}|^2|\mathcal{I}|^2|\mathcal{A}|))$)

Faster belief updates

$\implies$ Belief update can be done linearly in $O(|\mathcal{I}|)$ (rather than quadratic in terms of states $O(|\mathcal{S}|^2|\mathcal{I}|^2)$)

Monotonic expected belief entropy

In a MEMDP, the entropy of the current belief captures uncertainty on the environments, and is a (non-strictly) decreasing function in expectation.

Solving MEMDPS: A summary

Sparse transition function

$\Longrightarrow$ Memory usage: $O(|\mathcal{S}|^2|\mathcal{I}||\mathcal{A}|)$ (instead of $O(|\mathcal{S}|^2|\mathcal{I}|^2|\mathcal{A}|)$))

Faster belief updates

$\Longrightarrow$ Belief update can be done linearly in $O(|\mathcal{I}|)$ (rather than quadratic in terms of states $O(|\mathcal{S}|^2|\mathcal{I}|^2)$)

Monotonic expected belief entropy

$\Longrightarrow$ Monotonocity guarantee when using this quantity as a heuristics[7]

[7] Exact and approximate algorithms for partially observable Markov decision processes, Cassandra, 1998

Optimized Solvers

We use these properties to optimize two classic POMDP solvers for MEMDPs applications:

- SPBVI: Based on PBVI[3], with faster and memory-efficient belief expansion sets.

[3]Point-based value iteration: An anytime algorithm for POMDPs, Pineau et al, IJCAI 2003
[4]Monte-Carlo Planning in Large POMDPs, Silver and Veness, NeurIPS 2010

Optimized Solvers

We use these properties to optimize two classic POMDP solvers for MEMDPs applications:

- SPBVI: Based on PBVI[3], with faster and memory-efficient belief expansion sets.
- POMCP[4]: On top of faster belief update, we propose two further variants:
  - POMCP-ex: Exact belief update (rather than approximation) can be performed efficiently in MEMDPS
  - PAMCP: Caching mechanism to retain past histories in future executions, to better handle a stream of input queries

[3] Point-based value iteration: An anytime algorithm for POMDPs, Pineau et al, IJCAI 2003

[4] Monte-Carlo Planning in Large POMDPs, Silver and Veness, NeurIPS 2010

Experiment: Recommender systems

In prior work, MDPs have been used for capturing long-term interactions in recommender systems[5], assuming *a fixed environment* for each user. We instead propose to learn a controller that handles different user profiles by modeling this task using a MEMDP.

[5] An MDP-based Recommender System, Shani et al, JMLR 2005

Experiment: Recommender systems

In prior work, MDPs have been used for capturing long-term interactions in recommender systems[5], assuming *a fixed environment* for each user. We instead propose to learn a controller that handles different user profiles by modeling this task using a MEMDP.

(synthetic)	MDP	SPBVI	POMCP	POMCP-ex	PAMCP	PAMCP-ex
Accuracy	0.12 ± 0.03	-	0.64 ± 0.27	$\mathbf{0.77 \pm 0.07}$	0.68 ± 0.24	0.75 ± 0.08
Env. prediction	-	-	0.79 ± 0.33	$\mathbf{0.96 \pm 0.04}$	0.85 ± 0.30	0.94 ± 0.06
Runtime	5h30mn	OOM	9mn36s	**14s**	**14s**	36s

Table 1: Synthetic dataset experiments (using 8 environments, 8 products, sequence of length 5)

[5]An MDP-based Recommender System, Shani et al, JMLR 2005

Experiment: Recommender systems

In prior work, MDPs have been used for capturing long-term interactions in recommender systems[5], assuming *a fixed environment* for each user. We instead propose to learn a controller that handles different user profiles by modeling this task using a MEMDP.

(Foodmart)	MDP	SPBVI	POMCP	POMCP-ex
Accuracy	0.61 ± 0.14	0.62 ± 0.14	0.62 ± 0.14	0.62 ± 0.14
Precision	0.74 ± 0.09	-	$\mathbf{0.78 \pm 0.07}$	$\mathbf{0.78 \pm 0.08}$
Env. prediction	-	$\mathbf{0.60 \pm 0.31}$	0.54 ± 0.35	0.53 ± 0.36
Runtime	11mn57s	12mn 38s	46s	23s

Table 2: Foodmart dataset experiments (using 8 environments*, 3 products, sequence of length 8)

*: Environments are generated in a greedy manner, using perplexity as a metric

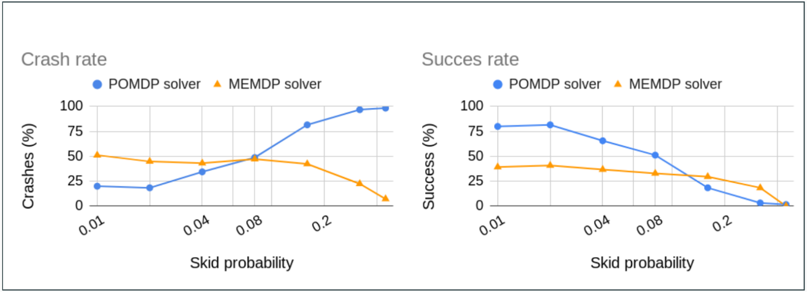
[5]An MDP-based Recommender System, Shani et al, JMLR 2005

Experiment: Maze solving with failure rate

The parametric Hallway maze problem consists in solving a maze where the agent has a certain (unknown) probability of “skidding”, i.e., failure, which we capture as different environments in a MEMDP.

Experiment: Maze solving with failure rate

The parametric Hallway maze problem consists in solving a maze where the agent has a certain (unknown) probability of "skidding", i.e., failure, which we capture as different environments in a MEMDP.



Conclusions

- MEMDPs are a straightforward tool for introducing context in MDPs
- Standard POMDPs solvers can be significantly optimized by considering specificities of MEMDPs
  - Sparse transition function
  - Faster belief update
  - Monotonicity of the average belief entropy
- We additionally verify the practicality of MEMDP-specific solvers through several experiments on recommender systems and a parametric version of the standard maze solving problem