Multi-Environment Markov Decision Processes: Efficient Analysis and Applications

K. Chatterjee\(^1\), M. Chmelík\(^2\), D. Karkhanis\(^3\), P. Novotný\(^4\), A. Royer\(^1\)

\(^1\)IST Austria, \(^2\)Google LLC, \(^3\)IIT Bombay, \(^4\)Masaryk University

**MEMDPs**

Multi-Environment Markov Decision Processes (MEMDPS) are an ideal tool for modeling families of MDPs with a similar transition structure, as can be encountered in many real-life applications (e.g., recommender systems).

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

- \(I\) is a finite set of environments;
- \(S\) is a finite set of control states;
- \(A\) is a finite alphabet of actions;
- \(\{\bar{\delta}_i\}_{i \in I}\) is a collection of probabilistic transition functions, one for every environment \(i \in I\);
- \(\{\bar{r}_i\}_{i \in I}\) is a set of reward functions;
- \(s_0 \in S\) is the initial state; and
- \(\lambda \in \mathcal{O}(I)\) is the initial distribution over the environments.

**Optimized Solvers**

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

- **PBVI**\(^2\): Leveraging the MEMDP structure helps with selecting a restricted set of beliefs to perform value iteration on, as well as for linear time belief updates. We call this variant **Sparse PBVI** (SPBVI).
- **POMCP**\(^3\): The sparse transition of MEMDPs can be readily applied. Building on this, we propose two variants:
  - **POMCP-ex**: Instead of estimating belief updates via Monte Carlo sampling, we can perform efficient exact belief updates in linear time in a MEMDP.
  - **PAMCP**: We use caching to retain past histories in future executions of the solver. This allows us to efficiently solve a stream of input queries while benefiting from past environment information.

**Applications and Experiments**

We explore several real-life applications modeled as MEMDPs, and explore how the proposed MEMDP-specific solvers performed compared to their standard counterpart.

**Recommender systems** benefit from being tailored to different **user profiles**. These can be modeled as different environments of a MEMDP, sharing the same transition structure (i.e., same products to recommend). Both retrieval accuracy and speed matter in these scenarios.

### Solving MEMDPs efficiently

A MEMDP can be converted to a Partially-Observable MDP (POMDP) by considering the cross-product \(I \times S\) as the new set of states. Hence standard POMDP solvers apply to this framework, however these are often suboptimal. Instead, we show that modeling and incorporating the specific nature of MEMDPS into these solvers allow for **great computational gains**, which can be summarized as the following three improvements:

1. **Sparse transitions**: The partially-observable (PO) feature (the environment \(i\)) is sampled only once, at initialization, and then kept constant. The memory required to store the transition function is thus only \(O(|I|^2 \cdot |S| \cdot |A|)\).

2. **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.

3. **Monotonic average belief entropy**: We show that, in a MEMDP, the expected change of entropy of the updated belief is always bounded. This can be used in POMDPs solver that rely on this signal as a guiding heuristic\(^\text{[1]}\).

---

**Table:**

<table>
<thead>
<tr>
<th>Environment</th>
<th>MDP</th>
<th>SPBVI</th>
<th>POMCP</th>
<th>POMCP-ex</th>
<th>PAMCP</th>
<th>PAMCP-ex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>0.61 ± 0.14</td>
<td>0.62 ± 0.14</td>
<td>0.62 ± 0.14</td>
<td>0.62 ± 0.14</td>
<td>0.77 ± 0.07</td>
<td>0.77 ± 0.07</td>
</tr>
<tr>
<td>Precision</td>
<td>0.78 ± 0.08</td>
<td>0.78 ± 0.08</td>
<td>0.78 ± 0.08</td>
<td>0.78 ± 0.08</td>
<td>0.78 ± 0.08</td>
<td>0.78 ± 0.08</td>
</tr>
<tr>
<td>Runtime</td>
<td>36s</td>
<td>36s</td>
<td>36s</td>
<td>36s</td>
<td>36s</td>
<td>36s</td>
</tr>
</tbody>
</table>

The **parametric Hallway maze problem**\(^\text{[4]}\) consists in solving a maze where the agent has a certain (unknown) probability of “skidding”, which we capture as different environments of a MEMDP.

---

\(^1\) Exact and approximate algorithms for partially-observable Markov decision processes, Cassandra, 1998

\(^2\) Point-based value iteration: An anytime algorithm for POMDPs, Pineau et al, IJCAI 2003

\(^3\) Monte-Carlo Planning in Large POMDPs, Silver and Veness, NeurIPS 2010

\(^4\) Parameter-independent strategies for POMDPs via POMDPs, Arming et al, QEST 2018

---

Example: User-aware recommender modeled as a MEMDP, and its POMDP equivalent (2 products, 2 users)