Solving Multi-Model MDPs by Coordinate Ascent and Dynamic Programming (Supplementary Material)

Xihong Su[1] **Marek Petrik**[1]

[1]Department of Computer Science , University of New Hampshire , Durham, NH, USA

A PROOF OF THEOREM 4.1

Proof of Theorem 4.1. For any time step $\hat{t} \in \mathcal{T}$, we can express the return as

$$\begin{aligned}
\rho(\pi) &= \mathbb{E}^{\lambda,\pi,p^{\tilde{m}},\mu}\left[\sum_{t=1}^{T} r_t^{\tilde{m}}(\tilde{s}_t, \tilde{a}_t)\right] \\
&= \mathbb{E}^{\lambda,\pi,p^{\tilde{m}},\mu}\left[\sum_{t=1}^{\hat{t}-1} r_t^{\tilde{m}}(\tilde{s}_t, \tilde{a}_t)\right] + \mathbb{E}^{\lambda,\pi,p^{\tilde{m}},\mu}\left[\sum_{t=\hat{t}}^{T} r_t^{\tilde{m}}(\tilde{s}_t, \tilde{a}_t)\right] \\
&\overset{\text{(a)}}{=} C + \mathbb{E}^{\lambda,\pi,p^{\tilde{m}},\mu}\left[\mathbb{E}\left[\sum_{t=\hat{t}}^{T} r_t^{\tilde{m}}(\tilde{s}_t, \tilde{a}_t) \mid \tilde{s}_{\hat{t}}, \tilde{m}\right]\right] \\
&\overset{\text{(b)}}{=} C + \sum_{m\in\mathcal{M}, s_{\hat{t}}\in\mathcal{S}, a_{\hat{t}}\in\mathcal{A}} \mathbb{P}\left[\tilde{m} = m, \tilde{s}_{\hat{t}} = s_{\hat{t}}\right] \pi_{\hat{t}}(s_{\hat{t}}, a_{\hat{t}}) \cdot \mathbb{E}^{\lambda,\pi,p^{\tilde{m}},\mu}\left[\sum_{t=\hat{t}}^{T} r_t^{\tilde{m}}(\tilde{s}_t, \tilde{a}_t) \mid \tilde{s}_{\hat{t}} = s_{\hat{t}}, \tilde{a}_{\hat{t}} = a_{\hat{t}}, \tilde{m} = m\right] \\
&\overset{\text{(c)}}{=} C + \sum_{m\in\mathcal{M}, s_{\hat{t}}\in\mathcal{S}, a_{\hat{t}}\in\mathcal{A}} b^{\pi}_{\hat{t},m}(s_{\hat{t}}) \cdot \pi_{\hat{t}}(s_{\hat{t}}, a_{\hat{t}}) \cdot q^{\pi}_{\hat{t},m}(s_{\hat{t}}, a_{\hat{t}})\,.
\end{aligned}$$

Here, we use $C = \mathbb{E}^{\lambda,\pi,p^{\tilde{m}},\mu}\left[\sum_{t=1}^{\hat{t}-1} r_t^{\tilde{m}}(\tilde{s}_t, \tilde{a}_t)\right]$ for brevity. The step (a) follows from the law of total expectation, the step (b) follows from the definition of conditional expectation, and the step (c) holds from the definitions of b and q in (3), (4), and (8).

Using the expression above, we can differentiate the return for each $s \in \mathcal{S}$ and $a \in \mathcal{A}$ as

$$\frac{\partial \rho(\pi)}{\partial \pi_{\hat{t}}(s, a)} \quad = \quad b^{\pi}_{\hat{t},m}(s) \cdot q^{\pi}_{\hat{t},m}(s, a)\,,$$

which uses the fact that C, $b^{\pi}_{\hat{t},m}$, and $q^{\pi}_{\hat{t},m}$ are constant with respect to $\pi_{\hat{t}}$. The desired result then holds by substituting t for $\hat{t}$, $\hat{s}$ for s, and $\hat{a}$ for a.

□

B PROOF OF THEOREM 5.1

Proof of Theorem 5.1. Assume some iteration n. The proof then follows directly from the contruction of the policy π^n from π^{n-1}. By the construction in Eq. (13), we have that:

$$\rho(\pi_1^{n-1}, \ldots \pi_{t-1}^{n-1}, \pi_t^n, \pi_{t+1}^n \ldots, \pi_T^n) \;\geq\; \rho(\pi_1^{n-1}, \ldots \pi_{t-1}^{n-1}, \pi_t^{n-1}, \pi_{t+1}^n \ldots, \pi_T^n)\,.$$

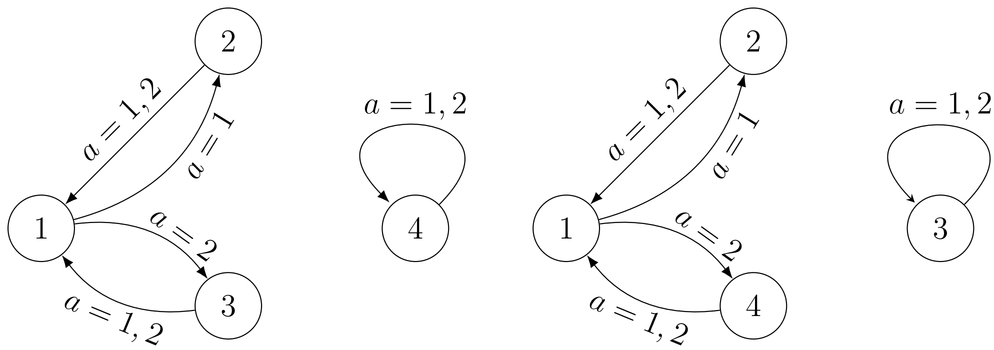

Figure 1: Left: model m_1, right: model m_2

Note that the optimal form of the policy in Eq. (13) follows immediately from the standard first-order optimality criteria over a simplex (e.g, Ex. 3.1.2 in [Bertsekas, 2016]) and the fact that the function optimized in Eq. (13) is linear (Corollary 4.2). In particular, we have that

$$\pi_t^n \in \operatorname*{argmax}_{\hat{\pi}_t \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}} \rho(\pi_1^{n-1}, \ldots, \hat{\pi}_t, \ldots, \pi_T^n)$$

if and only if for each $s \in \mathcal{S}$ and $a \in \mathcal{A}$

$$\frac{\partial \rho(\pi_1^{n-1}, \ldots, \pi_t^n, \ldots, \pi_T^n)}{\partial \pi_t(s, \pi_t^n(s))} \geq \frac{\partial \rho(\pi_1^{n-1}, \ldots, \pi_t^n, \ldots, \pi_T^n)}{\partial \pi_t(s, a)}.$$

Intuitively, this means that the optimal policy π_t^n must choose actions that have the *maximum* gradient for each state. The optimization in Eq. (13) then follows by algebraic manipulation from Theorem 4.1.

□

C PROOF OF THEOREM 5.3

Proof of Theorem 5.3. Consider the MMDP illustrated in Figure 1.

First, we describe the time steps, states, rewards, and actions for this MMDP. This MMDP has three time steps, four states $\mathcal{S} = \{1, 2, 3, 4\}$, two actions $\mathcal{A} = \{1, 2\}$, and two models $\mathcal{M} = \{1, 2\}$. The model weight for m_1 is λ, then the model weight for m_2 is $1 - \lambda$. State 1 is the only initial state. In model m_1, the only non-zero reward 2 is received upon reaching state 2. The agent takes action 1, which leads to a transition to state 2 with a probability of 1. The agent takes action 2, which leads to a transition to state 3 with probability 1. The agent takes action 1 or 2 in state 2, which leads to a transition to state 1 with probability 1. The agent takes action 1 or 2 in state 3, which leads to a transition to state 1 with probability 1. The agent takes action 1 or 2 in state 4, which leads to a transition to state 4 with probability 1.

In model m_2, the agent receives rewards 3 upon reaching state 4 and receives rewards 2 upon reaching state 2. The agent takes action 1, which leads to a transition to state 2 with probability 1. The agent takes action 2, which leads to a transition to state 4 with probability 1. The agent takes action 1 or 2 in state 2, which leads to a transition to state 1 with probability 1. The agent takes action 1 or 2 in state 4, which leads to a transition to state 1 with probability 1. The agent takes action 1 or 2 in state 3, which leads to a transition to state 3 with probability 1.

Now, let us analyze the regret of this MMDP. The optimal policy of the above example is a history-dependent policy. That is, to take action 2 at time step 1. At time step 2, the agent takes action 1 or 2, which leads to a transaction back to state 1. From time step 3, if the agent is in model m_1, then take action 1; if the agent is in model m_2, then take action 2.

Next, let us analyze the regret of a Markov policy for the MMDP. S_t represents a state at time step t. State 1 has two options: select action 1 or select action 2. If action 1 is selected, this will give a regret value of 0 in model 1 and a regret value of 1 in model 2. If action 2 is selected, this will give a regret value of 2 in model 1 and a regret value of 0 in model 2. Therefore, at time step 1, the total regret is 2λ or $1(1-\lambda)$. At time step 2, the agent takes action 1 or 2 in state S_2(1,3 or 4), which leads to a transition back to state 1, and gets zero rewards and zero regrets. Then repeat the procedure. At time step 3, the agent can take action 1 or action 2 in state 1 again. For $T = 3$, the trajectory of a Markov policy can be $(S_1 = 1, A_1 = 1, S_2, A_2, S_3 = 1, A_3 = 1)$, $(S_1 = 1, A_1 = 1, S_2, A_2, S_3 = 1, A_3 = 0)$, $(S_1 = 1, A_1 = 0, S_2, A_2, S_3 = 1, A_3 = 1)$, or $(S_1 = 1, A_1 = 0, S_2, A_2, S_3 = 1, A_3 = 0)$. The accumulated regret can be 2λ +$1(1-\lambda)$, 2λ + 2λ, $1(1-\lambda)$ + $1(1-\lambda)$. That is, the regret is increased by $1(1-\lambda)$ or 2λ for every two time steps.

$$R_t(\pi) \geq \frac{\min\{2\lambda, 1-\lambda\}}{2} \cdot t$$

Let $c = \frac{\min\{2\lambda, 1-\lambda\}}{2}$, $t' \geq 2$, then we always have

$$R_t(\pi) \geq c \cdot t \quad \text{for all} \quad t \geq t'$$

No matter which Markovian policy the agent follows, the accumulated regret will be linear with respect to t. Therefore, for this MMDP, there exists no Markovian policy that achieves sub-linear regret. $\square$

D ADAPTED MIXTS ALGORITHM

The adapted MixTS algorithm is formalized in Algorithm 1. P_0 is the prior of MDPs and follows the uniform distribution. At the beginning of episode t, sample a MDP M_t from the posterior P_t and compute a policy π_t that maximizes the value of M_t. Then at each time step h, take the action A_h based on the policy π_t and obtain reward Y_h. For each MDP $m \in \mathcal{M}$, update its posterior based on the received rewards.

Algorithm 1 Adapted MixTS

Input: The prior of MDPs P_0

Initialize $P_1 \longleftarrow P_0$
for episodes t =1, $\cdots$,$\mathcal{N}$ **do**
  Sample $M_t \sim P_t$
  Compute π_t = π^{M_t}
  for timesteps h = 1, $\cdots$, H **do**
    Select $A_h \leftarrow \pi_t(S_h)$
    Observe reward Y_h
    Update $P_{t+1}(m) \propto P_t(m)P(Y_h \mid A_h; m), \forall m \in \mathcal{M}$
  end for
end for

E NUMERICAL RESULTS: DETAILS

E.1 DOMAIN DETAILS

The CSV files of all domains are available at https://github.com/suxh2019/CADP. “initial.csv” specifies the initial distribution over states. “parameters.csv” contains the discount factor. “training.csv”and “test.csv”have the following columns: “idstatefrom”, “idaction”, “idstateto”, “idoutcome”, “probability”, and “reward”. Each row entry specifies a transition from “idstatefrom” after taking an action “idaction" to state “idstateto” with the associated “probability” and “reward” in model “idoutcome”. A policy is computed from the “training.csv”, and the policy is evaluated on the “test.csv”. The models are identified with integer values $0, \cdots, M-1$, and each model is defined on the same state space and the action space. The states are identified with integer values $0, \cdots, S-1$, and the actions are identified with integer values $0, \cdots, A-1$. Note that the number of actions taken in each state s is less or equal to A. Each MDP model has its unique reward functions and transition probability functions.

E.2 ADDITIONAL SIMULATION RESULTS

Table 1 shows mean returns of algorithms on five domains at different time steps. Table 2 shows the standard deviations of returns of algorithms on five domains at different time steps. The algorithm “Oracle” knows the true model and its standard deviation summarizes the variability of MDP models in an MMDP. The standard deviations of other algorithms include both the variability of MDP models and the variability of a policy in a MDP model. Table 3 shows runtimes of algorithms on five domains at different time steps. $CADP$ performs best with some runtime penalty.

Table 1: Mean Returns $\rho(\pi)$ on the Test Set of Policies π Computed by Each Algorithm.

Algorithm	**RS**		**POP**		**POPS**		**INV**		**HIV**	
	T = 100	T =150	T = 100	T =150	T = 100	T =150	T = 100	T =150	T = 5	T =20
CADP	**207**	**207**	**-368**	**-368**	**-1082**	**-1082**	348	**350**	**33348**	**42566**
WSU	206	206	-551	-551	-1934	-1932	347	349	**33348**	42564
MVP	204	204	-717	-717	-2178	-2179	348	**350**	**33348**	42564
Mirror	183	183	-1601	-1600	-3810	-3800	343	345	**33348**	**42566**
Gradient	206	206	-551	-551	-1934	-1932	347	349	**33348**	42564
MixTS	172	176	-1961	-1711	-3042	-3016	**350**	**350**	293	-1026
QMDP	201	183	-	-	-	-	-	-	30705	39626
POMCP	54	64	-	-	-	-	-	-	25794	30910
Oracle	213	213	-172	-172	-894	-894	358	360	40159	53856

Table 2: Standard Deviation of Returns of Algorithms on Five Domains.

Algorithm	**RS**		**POP**		**POPS**		**INV**		**HIV**	
	T = 100	T =150	T = 100	T =150	T = 100	T =150	T = 100	T =150	T = 5	T =20
CADP	98	98	**1095**	**1095**	**2007**	**2007**	**51**	**51**	9342	**11309**
MVP	90	90	2046	2046	3619	3620	52	52	**7729**	12234
WSU	100	100	1364	1364	3147	3146	53	53	**7729**	12234
Mirror	**70**	**70**	2081	2081	4534	4530	57	58	**7729**	12237
Gradient	100	100	1364	1364	3147	3146	53	53	**7729**	12234
MixTS	226	231	4436	4187	5507	5542	58	58	23689	27792
QMDP	193	204	-	-	-	-	-	-	42987	61596
POMCP	66	118	-	-	-	-	-	-	42208	57772
Oracle	95	95	1045	1045	1889	1889	51	51	9029	14796

Table 3: Run-times of Algorithms on Five Domains in Minutes.

Algorithm	**RS**		**POP**		**POPS**		**INV**		**HIV**	
	T = 100	T =150	T = 100	T =150	T = 100	T =150	T = 100	T =150	T = 50	T =100
MVP	**0.05**	**0.05**	**27.68**	**27.51**	**0.36**	**0.36**	**0.22**	**0.22**	**0.0003**	**0.0003**
WSU	0.12	0.14	40.02	45.39	1.53	2.37	0.67	0.89	0.0033	0.0048
CADP	0.52	1.13	124.39	173.04	12.12	16.21	1.53	2.22	0.0109	0.0164
Mirror	1.86	3.11	113.08	158.06	8.08	11.90	35.90	53.6	0.0221	0.0330
Gradient	0.51	0.74	56.82	69.32	2.97	4.31	1.12	1.44	0.0083	0.0123
MixTS	0.09	0.12	32.08	35.36	0.80	1.03	0.47	0.59	0.0033	0.0047
QMDP	712	712	-	-	-	-	-	-	0.7071	0.7071
POMCP	68	68	-	-	-	-	-	-	0.2066	0.2066