Don't Be Pessimistic Too Early: Look K Steps Ahead!

Chaoqi Wang
Unversity of Chicago

Ziyu Ye
Unversity of Chicago

Kevin Murphy
Google DeepMind

Yuxin Chen
Unversity of Chicago

Abstract

Offline reinforcement learning (RL) considers to train highly rewarding policies exclusively from existing data, showing great real-world impacts. Pessimism, *i.e.*, avoiding uncertain states or actions during decision making, has long been the main theme for offline RL. However, existing works often lead to overly conservative policies with rather sub-optimal performance. To tackle this challenge, we introduce the notion of *lookahead pessimism* within the model-based offline RL paradigm. Intuitively, while the classical pessimism principle asks to terminate whenever the RL agent reaches an uncertain region, our method allows the agent to use a lookahead set carefully crafted from the learned model, and to make a move by properties of the lookahead set. Remarkably, we show that this enables learning a *less conservative* policy with a *better performance* guarantee. We refer to our method as Lookahead Pessimistic MDP (LP-MDP). Theoretically, we provide a rigorous analysis on the performance lower bound, which monotonically improves with the lookahead steps. Empirically, with the easy-to-implement design of LP-MDP, we demonstrate a solid performance improvement over baseline methods on widely used offline RL benchmarks.

1 Introduction

Offline reinforcement learning presents a fertile ground for research, especially in its applicability to scenarios requiring RL policies to be learnt strictly from existing datasets, negating the need for online exploration [Levine et al., 2020]. This approach is crucial in real-world applications such as autonomous driving [Fang et al., 2022] and healthcare [Wang et al., 2018], where direct interactions can be expensive or dangerous. However, a unique challenge presents itself in the offline RL paradigm: the *distribution shift* issue, primarily due to the often inadequate representation of online scenarios by pre-existing datasets. To address this, an essential principle for offline RL is to incorporate algorithmic regularization to prevent erroneous extrapolations beyond the available data, a concept commonly referred to as *pessimism* [Jin et al., 2021].

Research in this domain is largely bifurcated into model-free and model-based approaches. While model-free methods utilize policy regularization [Fujimoto et al., 2019, Wang et al., 2020] or the learning of a conservative value function [Kumar et al., 2020, Kostrikov et al., 2022], they are inherently limited by the constraints of offline data, leading to potential poor generalization [Yu et al., 2020]. Model-based methods, in contrast, typically manifests the principle of pessimism by truncating uncertain transitions [Kidambi et al., 2020], or by imposing penalties on rewards in uncertain states [Yu et al., 2020, 2021]. In addition, they also afford the ability to interpolate and extrapolate from offline data through learning dynamics models, promising superior generalization and sample efficiency [Kidambi et al., 2020, Yu et al., 2020, 2021].

Building on this, we focus on the model-based offline RL paradigm, and more specifically, on extending the work of Kidambi et al. [2020], known as P-MDP. In comparison, methods like MOPO [Yu et al., 2020] and COMBO [Yu et al., 2021] rollout from the offline data using the fitted dynamics model for only a few steps (e.g., 1 to 5 steps), functioning as a form of data augmentation, akin to MBPO [Janner et al., 2019]. Hence, they could also be considered a hybrid of model-free and model-based methods, potentially still constrained by the static offline data due to the restricted rollout steps. Thus, our goal is to augment the performance of algorithms in the style of P-MDP, which exclusively perform rollouts in the learned model for a potentially extensive number of steps.

The main issue with P-MDP [Kidambi et al., 2020] is

Proceedings of the 27$^{\text{th}}$ International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238. Correspondence to chaoqi@uchicago.edu.

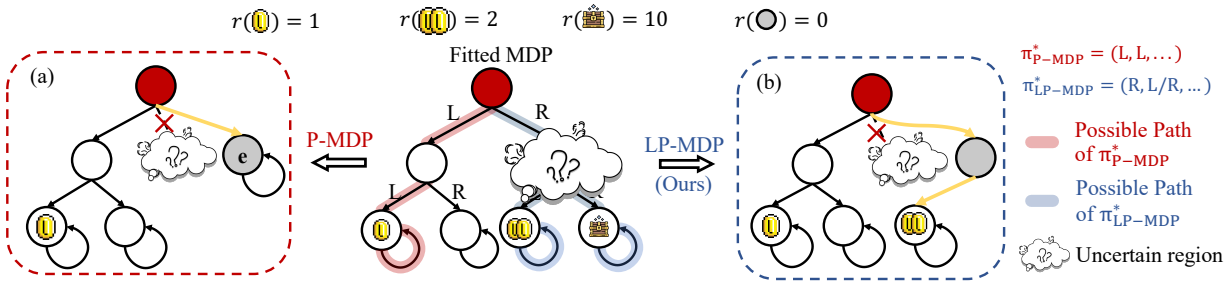


Figure 1: In this tree-structured MDP (middle figure), the fitted dynamics model isn't sure about the right action (R) at the red node. P-MDP (or MOReL) sets the next state as the absorbing state (see (a) on the left). In contrast, LP-MDP looks one step ahead in the fitted model dynamics to create a less pessimistic path around the uncertain region (see (b) on the right). As a result, the policies $\pi^\star_{\text{P-MDP}}$ (edges highlighted with ▬) and $\pi^\star_{\text{LP-MDP}}$ (edges highlighted with ▬) have different trajectories in the true MDP.

its tendency to be overly pessimistic too soon when it comes across an uncertain state. In this study, we put forward a refined version of the P-MDP approach by introducing the idea of *lookahead pessimism*, which we call LP-MDP. Our approach lessens the conservatism and provides a tighter lower bound for the actual performance of the policy learned, compared to P-MDP. We achieve this by creating a lookahead set that starts from the uncertain states, as opposed to the immediate termination when encountering them, as in P-MDP.

As an elucidatory example, we provide a graphical illustration in Figure 1, demonstrating how P-MDP can lead to a highly conservative policy with subpar performance. In the tree-structured MDP depicted in the center, the cloudy region symbolizes the area where the learned model is uncertain about the true transition distribution. In such a scenario, P-MDP truncates the transition and compels the subsequent state to be an absorbing state **e** with the minimal reward (as shown in Figure 1(a)). Conversely, our LP-MDP method looks ahead one step in the search for a less conservative MDP, as depicted in Figure1 (b). Consequently, the policies learned under P-MDP ($\pi^\star_{\text{P-MDP}}$) and LP-MDP ($\pi^\star_{\text{LP-MDP}}$) will exhibit different behaviors when deployed in the true MDP. In particular, $\pi^\star_{\text{P-MDP}}$ will avoid the right branch as it will yield the lowest return in the P-MDP, whereas $\pi^\star_{\text{LP-MDP}}$ will opt for the right branch, given that it provides a superior payoff in the LP-MDP, which is indeed better.

Main Contributions:

- We present the Lookahead Pessimistic MDP (LP-MDP), a novel model-based method for offline RL. Notably, we bring forth the concept of *lookahead pessimism*, which lessens the conservatism by incorporating future transitions from the learned model.
- Theoretically, we prove a performance lower bound for the policy learnt under LP-MDP. We also illustrate that this lower bound improves with the lookahead steps K, thus offering a tighter lower bound compared to previous works [Kidambi et al., 2020].
- Empirically, we showcase a simple-to-implement modular design of LP-MDP, and demonstrate a solid improvement over previous methods on both gridworld tasks and D4RL benchmarks.

2 Backgrounds and Preliminaries

Notations. We focus on the fully-observed Markov decision processes (MDPs), which can be represented by $\mathcal{M} = (\mathcal{S}, \mathcal{A}, p, r, \rho_0, \gamma)$. Here, $\mathcal{S}$ and $\mathcal{A}$ are the state space and action space, $p(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a})$ is the transition dynamics, $r(\boldsymbol{s}, \boldsymbol{a}) : \mathcal{S} \times \mathcal{A} \to [-R, R]$ is the reward function, $\rho_0(\cdot)$ denotes the initial state distribution, and $\gamma \in (0, 1)$ is the discount factor. The goal of reinforcement learning is to find a policy that maximizes the expected discounted return,

$$\eta_{\mathcal{M}}(\pi) \coloneqq \mathbb{E}_{\boldsymbol{s}\sim\rho_0(\cdot)}\left[V^\pi_{\mathcal{M}}(\boldsymbol{s})\right], \tag{1}$$

where $V^\pi_{\mathcal{M}}(\boldsymbol{s})$ is the value function defined as $V^\pi_{\mathcal{M}}(\boldsymbol{s}) \coloneqq \mathbb{E}_{\pi,p,\rho_0}\left[\sum_{t=0}^{\infty}\gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t)|\boldsymbol{s}_0 = \boldsymbol{s}\right]$. Lastly, we say two players $(x^\star, y^\star)$ with utility functions f_x and f_y are at an equilibrium if no player can deviate from its current strategy to achieve a better utility, *i.e.*, $\forall x, y$

$$f_x(x^\star, y^\star) \geq f_x(x, y^\star),\ f_y(x^\star, y^\star) \geq f_y(x^\star, y). \tag{2}$$

Offline RL as the maximin optimization. In offline RL, the goal is to learn a high-performing policy from the logged data. The key challenge is to handle the distribution shift in online deployment. Essentially, this problem can be formulated as the following maximin optimization problem,

$$\max_{\pi\in\Pi}\min_{\mathcal{M}\in\mathbb{M}} \eta_{\mathcal{M}}(\pi), \tag{3}$$

which aims to maximize the worst case performance. Consequently, the choice of the class of MDPs (i.e., $\mathbb{M}$) is crucial to the performance of the resulting policy. As long as $\mathbb{M}$ contains the true MDP, we will have that the true performance of the resulting policy is lower bounded by Equation 3.

In Uehara and Sun [2022], the authors choose $\mathbb{M}$ to be the set of MDPs that are not too far away from the MDP obtained by maximuim likelihood estimator, *i.e.*, $\mathcal{M}_{\widehat{p}}$, and the distance is measured by the total variation (TV) distance $\mathbf{d}_{\mathrm{TV}}(\cdot|\cdot)$, *i.e.*, $\mathbb{M} = \{\mathcal{M}|\mathbf{d}_{\mathrm{TV}}(\mathcal{M}_{\widehat{p}}|\mathcal{M}) \leq \xi\}$. However, their formulation is impractical due to the poor computation feasibility and efficiency of the set $\mathbb{M}$.

In contrast, Kidambi et al. [2020] propose a family of pessimistic MDP (P-MDP) that truncates transitions of $\mathcal{M}_{\widehat{p}}$ based on such TV distances, which is more tractable to handle. Specifically, for the state-action pair $(\boldsymbol{s}, \boldsymbol{a})$ who has a large TV distance between the its estimated transition distribution and the true transition distribution, their next state will be replaced by the absorbing state $\mathbf{e}$, that is:

$$\forall\, \boldsymbol{a} \in \mathcal{A}:\ p(\mathbf{e}|\boldsymbol{s}, \boldsymbol{a}) = 1, \quad \text{and} \quad r(\mathbf{e}, \boldsymbol{a}) = -R. \tag{4}$$

To be noted, the class of P-MDPs may not contain the true MDP. However, for any policy π, its performance under the P-MDP is (approximately) a lower bound of that in the true MDP. Therefore, maximizing the performance under the P-MDP is equivalent to (approximately) maximizing the performance lower bound in the real environment.

Revisiting P-MDPs. Our example in Figure 1 demonstrates that P-MDPs may be too conservative, leading to inferior performance. In this tree-structured MDP, the agent can decide to either go left (`L`) or right (`R`) at each node. We assume that the fitted dynamics model $\mathcal{M}_{\widehat{p}}$ is uncertain about the transitions of taking the right action (`R`) on the red node. For the remaining nodes, $\mathcal{M}_{\widehat{p}}$ is aligned with the true distribution, i.e., the predicted transition distribution has a small TV distance with the true transition distribution.

Consequently, P-MDP will force the next state of taking action `R` on red node to be the absorbing state $\mathbf{e}$ (see Figure 1 (a)). However, although we are unsure about the true transitions, what we are *certain* is that it will either transit to the node with treasure or the one with two coins in the end. Therefore, we can construct a worst-case path to circumvent the uncertain region as demonstrated in Figure 1 (b). In contrast, the policy obtained under (a) will be {`L`, `L`, ...}. However, for (b), the resulting policy is {`R`, `L` or `R`, ...}, which will perform better in the true environment.

3 The Lookahead Pessimistic MDP

In this section, we formalize the intuition conveyed from Figure 1. We use $p(\cdot|\cdot,\cdot)$ to denote the true transition distribution, $\widehat{p}(\cdot|\cdot,\cdot)$ is the transition distribution estimated from the offline data, and $\widetilde{p}(\cdot|\cdot,\cdot)$ is the pessimistic transition distribution, which will be instantiated later. To distinguish different MDPs, we denote the true MDP as $\mathcal{M}_p$, and the one obtained by maximizing likelihood as $\mathcal{M}_{\widehat{p}}$.

As demonstrated by Figure 1, it may be overly conservative if we truncate *all* the transitions on those state action pairs that have a large TV distance with the true distribution. The motivating example implies, that instead of replacing the next state to be an absorbing state, we can lookahead several steps to construct a less conservative transition distribution.

We denote the *estimation error* on any pair $(\boldsymbol{s}, \boldsymbol{a})$ as

$$d(\boldsymbol{s}, \boldsymbol{a}) \coloneqq \mathbf{d}_{\mathrm{TV}}\left(\widehat{p}(\cdot|\boldsymbol{s}, \boldsymbol{a})|\, p(\cdot|\boldsymbol{s}, \boldsymbol{a})\right), \tag{5}$$

which quantifies the total variation distance from the estimated distribution $\widehat{p}$ to the true distribution p.[1] This leads to the definition of the following set containing all state-action pairs with estimation errors larger than some threshold ξ:

Definition 1 (Uncertain State-Action Set) *$\forall\, \xi \geq 0$, the set of uncertain state-action pairs is*

$$\mathcal{U}(\xi) \coloneqq \{(\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{S} \times \mathcal{A} : d(\boldsymbol{s}, \boldsymbol{a}) \geq \xi\}. \tag{6}$$

For simplicity, we drop the dependency on ξ in the notations. Intuitively, $\mathcal{U}$ refers to the regions that the learned model has less confidence in, which may lead to more inaccurate transition estimation and end up with biased exploitation. To avoid such issues, the *pessimism* principle is implemented to tweak the transitions for all the state action pairs in $\mathcal{U}$. Specifically in MOReL, the next state is forced to be the absorbing state $\mathbf{e}$ if the current state action pair is in $\mathcal{U}$.

Next, we consider to assign *less conservative* transition distributions for some state-action pairs in $\mathcal{U}$ given some properties of their *lookahead sets* (which we later show in Section 3.1 that guarantees a better lower bound). We hereby formally introduce the k_{th}-step lookahead set for a state-action pair, which contains all the states that are *reachable at exactly k_{th} step away* from this state-action pair, by following the policy π under the designated MDP.

Definition 2 (Lookahead Set) *For any $(\boldsymbol{s}_t, \boldsymbol{a}_t) \in \mathcal{S} \times \mathcal{A}$ under MDP $\mathcal{M}_{p'}$ with $p'(\cdot|\cdot,\cdot,\pi)$ denoting the transition distribution relying on π, the k_{th}-step lookahead set is*

$$\mathcal{L}^{\pi,k}_{\mathcal{M}_{p'}}(\boldsymbol{s}_t, \boldsymbol{a}_t) \coloneqq \{\boldsymbol{s} \in \mathcal{S}|\ p'(\boldsymbol{s}_{t+k} = \boldsymbol{s}|\boldsymbol{s}_t, \boldsymbol{a}_t, \pi) > 0\}.$$

[1] Be aware that $d(\boldsymbol{s}, \boldsymbol{a})$ cannot be computed in practice yet is widely adopted in many prior works [Jin et al., 2020, Kidambi et al., 2020, Uehara and Sun, 2022] to make the theoretical analysis feasible. We discuss a practical surrogate for it in Section 3.2.

The intuition is, that considering a far-sighted agent, though the current transition is uncertain, if it is sure that reachable future transitions give better payoffs, the agent will go for it. Our decision under current uncertainty thus depends on such an outlook – technically this asks to divide $\mathcal{U}$ by some certainty criteria of lookahead sets, as elaborated below.

We say a state-action pair $(\boldsymbol{s}, \boldsymbol{a})$ is $\boldsymbol{k_{\text{th}}}$**-certain** if for all the states in its k_{th}-step lookahead sets, the fitted dynamics $\mathcal{M}_{\widehat{p}}$ induces only a small estimation error, that is:

$$\forall \boldsymbol{s}' \in \mathcal{L}^{\pi,k}_{\mathcal{M}_{\widehat{p}}}(\boldsymbol{s}, \boldsymbol{a}),\ d(\boldsymbol{s}', \pi(\boldsymbol{s}')) \leq \xi, \tag{7}$$

where $d(\boldsymbol{s}, \pi(\boldsymbol{s})) \coloneqq \max_{\boldsymbol{a}:\pi(\boldsymbol{a}|\boldsymbol{s})>0} d(\boldsymbol{s}, \boldsymbol{a})$. The set $\mathcal{U}$ thus can be partitioned into disjoint subsets by the above lookahead certainty criteria. For all $k \in [1, K]$, we define the subset $\mathcal{U}^{\pi}_{k}$ as

$$\mathcal{U}^{\pi}_{k} \coloneqq \left\{(\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{U} \setminus \cup_{i=1}^{k-1}\mathcal{U}^{\pi}_{i} : (\boldsymbol{s}, \boldsymbol{a}) \text{ is } \boldsymbol{k_{\text{th}}}\textbf{-certain}\right\}.$$

Intuitively, $\mathcal{U}^{\pi}_{k}$ contains all $\boldsymbol{k_{\text{th}}}$**-certain** state action pairs in $\mathcal{U}$ which are uncertain for intermediary steps before k. [2] We further use $\mathcal{U}^{\pi}_{-(1:K)} \coloneqq \mathcal{U} \setminus \cup_{i=1}^{K}\mathcal{U}^{\pi}_{i}$ to denote all the remaining state-action pairs in $\mathcal{U}$. We are now ready to define LP(ξ, π, K)-MDP, or LP-MDP for short.

Definition: Pessimistic MDP with K-step Lookahead (LP-MDP)

Definition 3 (LP(ξ, π, K)-MDP) *For any $(\boldsymbol{s}_t, \boldsymbol{a}_t)$, and $\mathcal{M}_{\widehat{p}}$ and $\mathcal{M}_p$, the LP(ξ, π, K)-MDP of $\mathcal{M}_p$, (i.e., $\mathcal{M}^{\pi}_{\widetilde{p}}$) is constructed by modifying the transition $\widehat{p}$ to be $\widetilde{p}$ as:*[a]

Case 1 *(current transition is* certain*) If $(\boldsymbol{s}_t, \boldsymbol{a}_t) \notin \mathcal{U}$:*

$$\widetilde{p}(\cdot|\boldsymbol{s}_t, \boldsymbol{a}_t) = \widehat{p}(\cdot|\boldsymbol{s}_t, \boldsymbol{a}_t), \tag{8}$$

Case 2 *(current transition is* uncertain*) If $(\boldsymbol{s}_t, \boldsymbol{a}_t) \in \mathcal{U}$:*

Case 2.1 *(all K-step look ahead is uncertain) If $(\boldsymbol{s}_t, \boldsymbol{a}_t) \in \mathcal{U}^{\pi}_{-(1:K)}$:*

$$\widetilde{p}(\boldsymbol{s}_{t+1} = \mathbf{e}|\boldsymbol{s}_t, \boldsymbol{a}_t) = 1, \tag{9}$$

Case 2.2 *(some k_{th}-step look ahead is certain) If $(\boldsymbol{s}_t, \boldsymbol{a}_t) \in \mathcal{U}^{\pi}_{k}$ for some $k \in [K]$, then we construct a deterministic path such that $\forall\, i \in \{1, ..., k-1\}$,*

$$\widetilde{p}(\boldsymbol{s}_{t+k} = \boldsymbol{s}^{\star}|\boldsymbol{s}_t, \boldsymbol{a}_t, \pi) = 1,\ \widetilde{r}(\boldsymbol{s}_{t+i}, \cdot) = -R. \tag{10}$$

where $\boldsymbol{s}^{\star}$ is defined as: $\boldsymbol{s}^{\star} = \arg\min_{\boldsymbol{s}' \in \mathcal{L}^{\pi,k}_{\mathcal{M}_{\widehat{p}}}(\boldsymbol{s}_t, \boldsymbol{a}_t)} V^{\pi}_{\mathcal{M}^{\pi}_{\widetilde{p}}}(\boldsymbol{s}')$,

[a] The associated reward $\tilde{r}(\cdot) \coloneqq r(\cdot)$ unless otherwise stated.

[2] Notice that $\mathcal{U}^{\pi}_{1} \coloneqq \{(\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{U} : (\boldsymbol{s}, \boldsymbol{a}) \text{ is } 1_{\text{th}}\text{-certain}\}$.

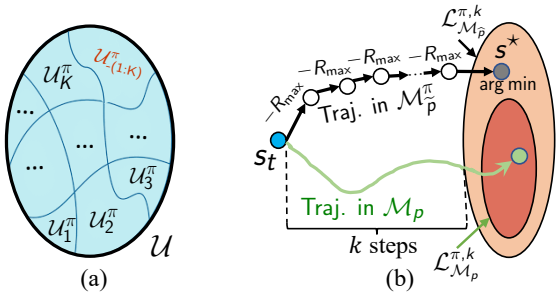


Figure 2: Illustration of (a) partitioning $\mathcal{U}$ into $\{\mathcal{U}^{\pi}_{k}\}_{k=1}^{K}$ and $\mathcal{U}^{\pi}_{-(1:K)}$; and (b) Case 2.2 of Definition 3 under K-step lookahead, where the gray dot represents $\boldsymbol{s}^{\star}$ and the blue dot denotes a current state $\boldsymbol{s}_t$ whose transitions are unknown; as a side note, while P-MDP halts after $\boldsymbol{s}_t$, LP-MDP constructs a worse-case (locally, only for k steps) but less conservative (globally) path, enabling the agent to transit from $\boldsymbol{s}_t$ to $\boldsymbol{s}^{\star}$ after exactly k steps.

For LP(ξ, π, K)-MDP $\mathcal{M}^{\pi}_{\widetilde{p}}$, the modification to $\widehat{p}$ is done in three parts.[3] For the state action pairs that fall in Case 1, we keep $\widetilde{p}$ the same as the original transition distribution $\widehat{p}$. In Case 2.1 when the current state action pair is in $\mathcal{U}^{\pi}_{-(1:K)}$, it will deterministically transit to the terminal state with the lowest reward $-R$. To be noted, if we ignore Case 2.2, the P-MDP formulation is recovered.

Case 2.2 plays a central role in LP(ξ, π, K)-MDP. For any state action pairs that fall in this case, we can construct a less pessimistic path for them instead of always forcing their next state to be the absorbing state $\mathbf{e}$. Specifically, we will set the new transition probability $\widetilde{p}$ such that it will deterministically transit to the state in $\mathcal{L}^{\pi,k}_{\mathcal{M}_{\widehat{p}}}(\boldsymbol{s}_t, \boldsymbol{a}_t)$ with minimal values after *exactly* k steps.

To further demonstrate this *lookahead pessimism* concept, we provide an illustration for Case 2.2 in Figure 2 (b), where the blue dot represents the state action pair $(\boldsymbol{s}_t, \boldsymbol{a}_t) \in \mathcal{U}^{\pi}_{k}$. Here, we force the transition of $\widetilde{p}$ to follow the *black* deterministic path for k steps, and the k_{th} state $\boldsymbol{s}_{t+k}$ is set to be the one in $\mathcal{L}^{\pi,k}_{\mathcal{M}_{\widehat{p}}}(\boldsymbol{s}_t, \boldsymbol{a}_t)$ that has the minimal reward-to-go. Notably, the RL agent will deterministically navigate through $\boldsymbol{s}_t$ to $\boldsymbol{s}_{t+k}$ by transitioning to an *external* state with the minimal reward $-R$ at each step. Notice that such path construction is *fundamentally different from model rollouts* [Yu et al., 2021, Sikchi et al., 2022]. Intuitively, our agent will *circumvent* the uncertain regions in $\widehat{p}$, by constructing a worst-case (locally, only for k steps) but less conservative (globally) path in $\widetilde{p}$.

Once we construct the LP(ξ, π, K)-MDP $\mathcal{M}^{\pi}_{\widetilde{p}}$, we can optimize the policy under it with any off-the-shelf RL

[3] We add a superscript π to indicate that the transition distribution depends on the running policy π.

algorithms. Next, we provide a theoretical analysis on the properties of $\mathcal{M}_{\widetilde{p}}^{\pi}$.

3.1 Theoretical Analysis

In this section, we present the theoretical guarantee of the policy learned under LP-MDP. We also illustrate the effect of K, *i.e.*, the number of lookahead steps, on the policy performance.

We start by defining several notations to use throughout the analysis. We first define a intermediary distribution $\widetilde{q}$, which satisfies

$$(\boldsymbol{s},\boldsymbol{a}) \in \mathcal{U} : \widetilde{q}(\cdot|\boldsymbol{s},\boldsymbol{a}) = \widetilde{p}(\cdot|\boldsymbol{s},\boldsymbol{a}), \tag{11}$$

$$(\boldsymbol{s},\boldsymbol{a}) \notin \mathcal{U} : \widetilde{q}(\cdot|\boldsymbol{s},\boldsymbol{a}) = p(\cdot|\boldsymbol{s},\boldsymbol{a}). \tag{12}$$

The distribution $\widetilde{q}$ can be regarded as a distribution which shifts between $\widetilde{p}$ or p, depending on whether the next transition for the current state action pair is uncertain or not. Next, we introduce the hitting time and the occupancy measure.

Definition 4 (Hitting Time & Occupancy Measure) *Given an MDP $\mathcal{M}_p$, a policy π, a state action pair $(\boldsymbol{s},\boldsymbol{a})$ and an initial state $\boldsymbol{s}_0$, we define*

- **Hitting time:** $\mathcal{T}^{\pi}_{\boldsymbol{s}_0\to(\boldsymbol{s},\boldsymbol{a})}$ is a random variable denoting the first time step that the action $\boldsymbol{a}$ is taken at $\boldsymbol{s}$ by following the policy π starting from $\boldsymbol{s}_0$ in $\mathcal{M}_p$. For any set of state action pairs $\mathcal{U}$, we define $\mathcal{T}^{\pi}_{\boldsymbol{s}_0\to\mathcal{U}} \coloneqq \min_{(\boldsymbol{s}',\boldsymbol{a}')\in\mathcal{U}} \mathcal{T}^{\pi}_{\boldsymbol{s}_0\to(\boldsymbol{s}',\boldsymbol{a}')}$.
- **Occupancy measure:** $\rho^{\pi}_{\boldsymbol{s}_0\to(\boldsymbol{s},\boldsymbol{a})}$ denotes the probability of π encounter $(\boldsymbol{s},\boldsymbol{a})$ when navigating in $\mathcal{M}_p$ by starting from $\boldsymbol{s}_0$. For a set of state action pairs $\mathcal{U}$, we define $\rho^{\pi}_{\boldsymbol{s}_0\to\mathcal{U}} \coloneqq \sum_{(\boldsymbol{s}',\boldsymbol{a}')\in\mathcal{U}} \rho^{\pi}_{\boldsymbol{s}_0\to(\boldsymbol{s}',\boldsymbol{a}')}$.

We hereby present the theorem quantifying the performance of the policy learned under LP-MDP.

Theorem 1 (Performance Guarantee) *Let $\widetilde{\pi}^\star$ denote the equilibrium policy learned under the LP$(\xi,\widetilde{\pi}^\star,K)$-MDP, and let $\pi^\star$ be the optimal policy under the true MDP $\mathcal{M}_p$. Suppose that for any $(\boldsymbol{s}_t,\boldsymbol{a}_t) \in \mathcal{U}_k^{\widetilde{\pi}^\star}$ with $k\in[1,K]$, $\mathcal{L}^{\widetilde{\pi}^\star,k}_{\mathcal{M}_p}(\boldsymbol{s}_t,\boldsymbol{a}_t) \subseteq \mathcal{L}^{\widetilde{\pi}^\star,k}_{\mathcal{M}_{\widetilde{p}}}(\boldsymbol{s}_t,\boldsymbol{a}_t)$ holds, then for any state $\boldsymbol{s}$*

$$\begin{aligned} &V^{\pi^\star}_{\mathcal{M}_p}(\boldsymbol{s}) - V^{\widetilde{\pi}^\star}_{\mathcal{M}_p}(\boldsymbol{s}) \\ &\le \underbrace{\frac{4\gamma\xi R_{\max}}{(1-\gamma)^2}}_{\text{(a)}} + \underbrace{\frac{2\rho^{\pi^\star}_{\boldsymbol{s}\to\mathcal{U}^{\widetilde{\pi}^\star}_{-(1:K)}} \mathbb{E}\left[\gamma^{\mathcal{T}^{\pi^\star}_{\boldsymbol{s}\to\mathcal{U}^{\widetilde{\pi}^\star}_{-(1:K)}}}\right] R_{\max}}{1-\gamma}}_{\text{(b) Incurred by hitting } \mathcal{U}^{\widetilde{\pi}^\star}_{-(1:K)}} \\ &+ \underbrace{\sum_{k=1}^{K} \rho^{\pi^\star}_{\boldsymbol{s}\to\mathcal{U}^{\widetilde{\pi}^\star}_k} \mathbb{E}\left[\gamma^{\mathcal{T}^{\pi^\star}_{\boldsymbol{s}\to\mathcal{U}^{\widetilde{\pi}^\star}_k}}\right] \left(\textstyle\sum_{i=1}^{k-1} 2\gamma^i R_{\max} + \gamma^k \Delta^{\pi^\star,k}_{p,\widetilde{q}}(\mathcal{U}^{\widetilde{\pi}^\star}_k)\right)}_{\text{(c) Incurred by hitting } \mathcal{U}^{\widetilde{\pi}^\star}_k \text{ for } k\in[K]}. \end{aligned} \tag{13}$$

The notation $\Delta^{\pi,k}_{p,q}(\mathcal{U})$ above is defined as (we omitted the dependency on $\widetilde{\pi}^\star$ in notation for clarity.)

$$\Delta^{\pi,k}_{p,q}(\mathcal{U}) \coloneqq \max_{(\boldsymbol{s},\boldsymbol{a})\in\mathcal{U}} \Delta^{\pi,k}_{p,q}(\boldsymbol{s},\boldsymbol{a}), \tag{14}$$

where $\Delta^{\pi,k}_{p,q}(\boldsymbol{s},\boldsymbol{a}) \coloneqq \max_{\boldsymbol{s}_1,\boldsymbol{s}_2\in\mathcal{L}^{\widetilde{\pi}^\star,k}_{\mathcal{M}_p}(\boldsymbol{s},\boldsymbol{a})} V^{\pi}_{\mathcal{M}_p}(\boldsymbol{s}_1) - V^{\pi}_{\mathcal{M}_q}(\boldsymbol{s}_2)$.

Theorem 1 bounds the difference between the performance of the policy learnt under LP(ξ,π,K)-MDP and that of the optimal policy in the true environment $\mathcal{M}_p$. The bound can be decomposed into three parts: (a) comes from those certain state action pairs $(\boldsymbol{s},\boldsymbol{a})$ with $d(\boldsymbol{s},\boldsymbol{a}) \le \xi$; (b) is incurred by hitting the state action pair in $\mathcal{U}^{\widetilde{\pi}^\star}_{-(1:K)}$, whose next state is the absorbing state $\mathbf{e}$; Lastly, (c) is caused by hitting the state action pair in $\mathcal{U}^{\widetilde{\pi}^\star}_k$ with $k\in[K]$, of which the k_{th} next state is the one that minimizes the reward to go. To be noted, our proof requires a detailed analysis on the property of the lookahead set and cannot be directly adapted from the proof of Kidambi et al. [2020].

Next, we show that we can achieve a tighter lower bound by increasing the value of K, such that LP-MDP is guaranteed to be comparable or better than P-MDP, the classical model-based offline RL baseline [Kidambi et al., 2020]. Denoting Equation 13 in Theorem 1 as $\texttt{LB}^K$, we present the corollary below, which holds due to the fact that the blue term is smaller than $2R/(1-\gamma)$.

Corollary 1 *Under the same conditions as in Theorem 1, $\texttt{LB}^K \le \texttt{LB}^{K+1}$ holds for any $K\in\mathbb{Z}_{\ge 0}$.*

In particular, when $K=0$, the term (c) in Equation 13 will be 0, which recovers the lower bound of P-MDP. The above corollary shows that a simple way to improve the lower bound is using a larger value of K to look ahead. [4] In addition to the above results, we also prove that the policy improvement lemma holds for LP-MDP under some additional assumptions in Appendix A.5.

3.2 Practical Implementation

We introduce the practical implementation of LP-MDP in this section. Our algorithm can be decomposed into three parts: 1) the model learning; 2) the LP-MDP construction; and 3) the policy learning. We elaborate as follows.

Dynamics model learning. We use the deep ensemble [Lakshminarayanan et al., 2017] for dynamics model learning, which has been commonly adopted in

[4] We have provided empirical verification on the effect of K. Additional computational cost may arise though – in practice, we should properly choose the value of K to trade-off the performance and computation.

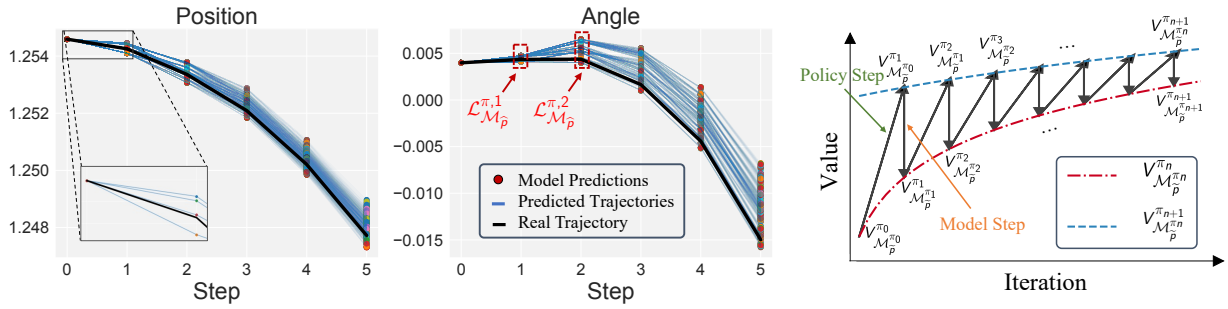


Figure 3: (1) The left two figures show the trajectory rollouts under a ensemble of 4 networks trained on the `Hopper-medium-replay` dataset with a fixed policy π. The black curve corresponds to the trajectory of running the policy π in real environment. (2) The rightmost figure presents an illustration of policy learning under LP-MDP, where the policy step and model step correspond to updating policy and constructing LP-MDP for the policy (Equation 19).

the RL literature [Janner et al., 2019, Chua et al., 2018, Kidambi et al., 2020]. For each network in the ensemble $f_{\boldsymbol{\theta}}$, it takes a normalized state action pair $(\boldsymbol{s}_t, \boldsymbol{a}_t)$ as input, and predicts the residual of the next state $\Delta \boldsymbol{s}_t$,

$$f_{\boldsymbol{\theta}}\left((\boldsymbol{s}_t - \boldsymbol{\mu}_s)/\boldsymbol{\sigma}_s, (\boldsymbol{a}_t - \boldsymbol{\mu}_a)/\boldsymbol{\sigma}_a\right) = \Delta \boldsymbol{s}_t, \tag{15}$$

where $\Delta \boldsymbol{s}_t \coloneqq (\boldsymbol{s}_{t+1} - \boldsymbol{s}_t)/\boldsymbol{\sigma}_s$, and $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are computed using the offline dataset. All the networks are trained on the offline dataset independently using the mean-squared loss with the Adam optimizer [Kingma and Ba, 2015].

LP-MDP construction. Given the learned dynamics models, we construct the LP-MDP based on Definition 3. As a practical solution, we consider the discrepancy-based uncertainty quantification as a proxy for $d(\boldsymbol{s}, \boldsymbol{a})$, which has been successfully applied in prior works [Kidambi et al., 2020] and achieved impressive performance. Specifically, given an ensemble of dynamics models $\{f_{\boldsymbol{\theta}_i}\}_{i=1}^N$, the discrepancy for a state action pair $(\boldsymbol{s}_t, \boldsymbol{a}_t)$ is computed by,

$$\mathrm{disc}(\boldsymbol{s}_t, \boldsymbol{a}_t) = \max_{i,j} \left\| f_{\boldsymbol{\theta}_i}(\boldsymbol{s}_t, \boldsymbol{a}_t) - f_{\boldsymbol{\theta}_j}(\boldsymbol{s}_t, \boldsymbol{a}_t) \right\|_2. \tag{16}$$

In addition, for constructing the lookahead set defined in Definition 2 (see the middle plot in Figure 3), we discretize the search space using with a finite number of particles, where each particle corresponds to the prediction of a single neural network in the ensemble for continuous tasks.

Policy learning under LP-MDP. Learning policy under LP-MDP can be viewed as a game between the policy and LP-MDP. In contrast to typical RL settings, there is no standard workhorse for solving continuous games. Following prior works in online model-based RL [Rajeswaran et al., 2020], we formulate our problem as a Stackelberg game [Von Stackelberg, 2010], where the two players are the policy π and its corresponding LP(ξ, π, K)-MDP. Since the best response of LP-MDP can be computed analytically, we choose the policy to be the leader and LP-MDP to be the follower, which results in the optimization problem below,

$$\max_{\pi} \left\{ \eta_{\mathcal{M}}(\pi) \quad s.t. \quad \mathcal{M} = \mathcal{M}_{\widetilde{p}}^{\pi} \right\}. \tag{17}$$

For policy learning, we adopt the soft actor-critic algorithm [Haarnoja et al., 2018]. Then, the above nested optimization can be solved by repeating the following updates (see Figure 3 for illustration),

$$\text{Model Step}: \quad \mathcal{M}_i \leftarrow \mathcal{M}_{\widetilde{p}}^{\pi_i}, \tag{18}$$

$$\text{Policy Step}: \quad \pi_{i+1} \leftarrow \texttt{SAC\_Step}(\pi_i, \mathcal{M}_i). \tag{19}$$

4 Experiments

In this section, we present two sets of tests to check how well our proposed algorithm works. The first set of experiments is conducted on a grid-world task, where the dynamics of the environment are known. This enables us to calculate the estimation error of the model and gain insights into how our method enhances performance. In the second set of experiments, we investigate our method's performance on a standard benchmark, D4RL [Fu et al., 2020], compared to baseline methods. Further details about the experimental setups can be found in the appendix.

4.1 Grid Environment

Our gridworld environment is adapted from the one used in Eysenbach et al. [2022] with increased difficulty. The environment is a 10×10 gridworld with stochastic dynamics (see Figure 9 (a)). At each grid, the agent can take four possible actions, i.e., $\{\leftarrow, \uparrow, \rightarrow, \downarrow\}$. Once an action is taken, there is 20% of the chance it gets flipped, due to the stochastic dynamics. Each episode lasts for 200 steps.

At the beginning, the agent starts from the top left corner, and the treasure is located at the top right

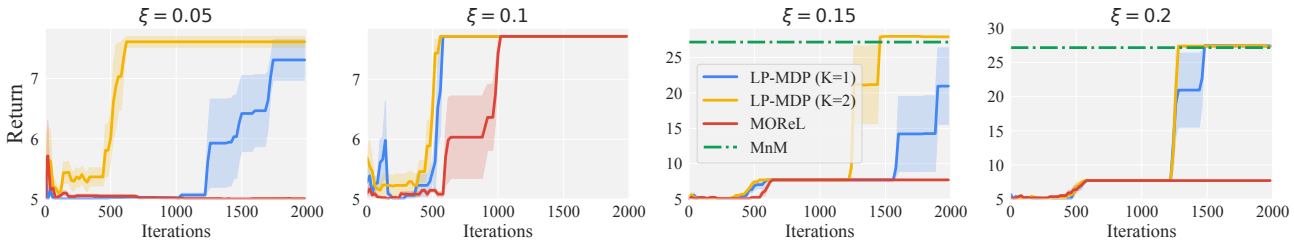


Figure 4: Results on the gridworld task. The dashed green curve is the performance of the expert policy obtained with MnM [Eysenbach et al., 2022]. The shaded region is the one standard error of three trials with different random seeds.

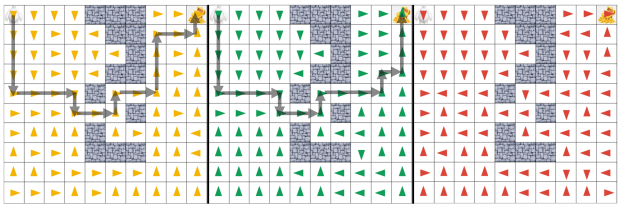


Figure 5: A visualization of the policies. The yellow policy (left) is the one learnt under LP-MDP with $k = 2$ and $\xi = 0.15$. In the middle, the green policy is the expert policy learnt using MnM [Eysenbach et al., 2022]. Lastly, the red policy corresponds to the one learnt using MOReL. The grey trajectories are possible paths from the starting grid to the treasure by following the learned policies.

corner. If the agent hits the grid with the treasure, it receives a reward of +50. For those grids with yellow shades, the reward is +1, and all the remaining grids has a reward of +0.5.

To generate the offline data, we first train a policy using MnM [Eysenbach et al., 2022] until convergence. We then randomly sample 20% of the data from its replay buffer as the data for offline model learning. The results are presented in Figure 4 under different model estimation error constraints $\xi \in \{0.05, 0.1, 0.15, 0.2\}$ and lookahead steps $K \in \{1, 2\}$. Remarkably, we can observe that the larger the value of K, the better the performance of the learned policy as well as a tighter lower bound on the values (see Figure 5), which is consistent to our Theorem 1 and Corollary 1. Our method also converges either faster than MOReL or achieves better returns in the true environment.

We further visualize the learned policies of MnM, MOReL and our method in Figure 5, where we observe that our method learns a policy that is similar to that of MnM. In contrast, the policy of MOReL will get stuck at the lower left region, where each grid has a slightly higher reward. To investigate why MOReL fails to find a path that leads to the treasure, we further plot the uncertain states in Figure 6 right, *i.e.*, $\mathcal{U}$. For MOReL (middle figure), the grid with treasure is mostly surrounded by those uncertain states, whereas our method (right figure) has far fewer uncertain states.

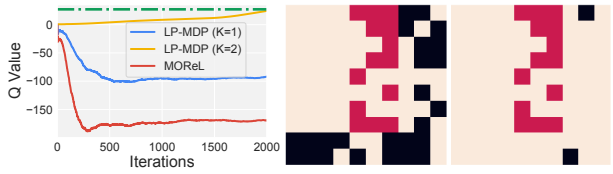


Figure 6: (1) The left figure shows the estimated Q value of the starting grid (up left corner). (2) The right two figures (the middle figure for P-MDP, and the right figure for LP-MDP) visualize the uncertain states (colored in black) whose next states contain the absorbing state. The red grids correspond to walls in the map.

Intuitively, the lookahead mechanism serves as a way to *increase the data coverage*, which is crucial to the performance of offline RL algorithms [Uehara and Sun, 2022, Kumar et al., 2022, Lu et al., 2022].

4.2 D4RL

Our second set of experiments is conducted on the D4RL benchmark [Fu et al., 2020], covering three environments: `HalfCheetah`, `Hopper`, and `Walker2d`. We benchmark our method against several model-free algorithms, such as SAC [Haarnoja et al., 2018], BEAR [Kumar et al., 2019], BRAC [Wu et al., 2019], and CQL [Kumar et al., 2020]; along with three model-based algorithms, MOPO [Yu et al., 2020] and MOReL [Kidambi et al., 2020]. For MOReL, given that its original implementation utilized the natural policy gradient (NPG) for policy optimization, we adapted MOReL to also use SAC to ensure a fair comparison. The configurations (e.g., hyperparameters) for our method are the same as those used for MOReL-SAC, which were determined by fine-tuning MOReL-SAC for reasonable performance. For our approach, we set the lookahead step K to be 1.

We report the results in Table 1. Firstly, with lookahead, our method achieves a solid improvement over our major baseline MOReL-SAC across 5 out 12 tasks and a significant improvement in the overall scores. Our method is also the best performing one in terms of the overall score. We further plot the distribution of trajectory length and the average trajectory length

Table 1: We present the highest average normalized scores obtained from D4RL throughout the training process, calculated across three distinct random seeds, and include the standard error for each score. Furthermore, we highlight the best-averaged scores using a blue box. Also, we use * to indicate the tasks where our method has a solid improvement over MOReL-SAC.

Dataset	Environment	LP-MDP (Ours)	MOReL (SAC)	MOReL (NPG)	MOPO	CQL	SAC-off	BEAR	BRAC-p	BRAC-v
random	halfcheetah	47.4 ±2.2	47.6	25.6	34.4	35.4	30.5	25.1	24.1	21.2
random	hopper	33.3 ±0.5	33.0	53.6	11.7	10.8	11.3	11.4	11	12.2
random	walker2d	22.2 ±0.2	22.2	37.3	13.6	7.0	4.1	7.3	-0.2	1.9
medium	halfcheetah	47.8 ±3.4	47.6	42.1	42.1	44.4	-4.3	41.7	43.8	46.3
medium	hopper	101.9 ±1.1*	75.8	95.4	28.0	86.6	0.8	52.1	32.7	31.1
medium	walker2d	64.5±5.1	76.8	77.8	17.8	74.5	0.9	59.1	77.5	81.1
medium-replay	halfcheetah	50.2±2.3	56.4	40.2	53.1	46.2	-2.4	38.6	45.4	47.7
medium-replay	hopper	101.2 ±0.8	101.1	93.6	67.5	48.6	3.5	33.7	0.6	0.6
medium-replay	walker2d	82.7 ±5.9*	46.5	49.8	39.0	32.6	1.9	19.2	-0.3	0.9
medium-expert	halfcheetah	68.8 ±8.9*	67.6	53.3	63.3	62.4	1.8	53.4	44.2	41.9
medium-expert	hopper	103.7±1.2*	78.5	108.7	23.7	111.0	1.6	96.3	1.9	0.8
medium-expert	walker2d	81.5±8.5*	68.0	95.6	44.6	98.7	-0.1	40.1	76.9	81.6
Overall Scores		805.2 *	721.1	773.0	438.7	658.2	49.6	478.0	363.0	378.3

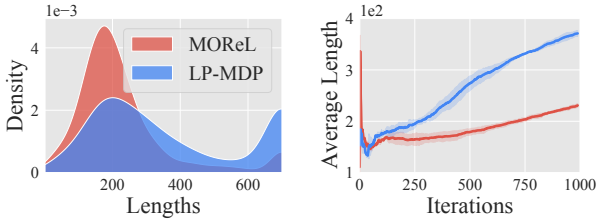


Figure 7: The distribution of trajectory lengths (left) and the averaged trajectory length (right) during the training of MOReL and LP-MDP on `Walker2d-medium-replay`.

during training in Figure 7. In contrast to MOReL-SAC, our method can explore longer trajectories more frequently during training. This is achieved by "*stitching*" together trajectories with lookahead when encountering uncertain states in LP-MDP, whereas P-MDP will directly truncate the trajectory. Furthermore, the (L)P-MDP algorithms significantly outperform MOPO on random splits. This could be attributed to their confinement on low-performing data, a result of the limited rollout steps.

5 Related Works

Model-Based Offline RL. This line of work usually focuses on learning *robust* transition dynamics by the pessimism principle which penalizes unknown state visitation [Kidambi et al., 2020, Yu et al., 2020, 2021, Argenson and Dulac-Arnold, 2021, Uehara and Sun, 2022, Rigter et al., 2022], or training a more *adaptable* policy based on a diverse model dynamics set [Chen et al., 2021]. Specifically, Kidambi et al. [2020] (MOReL) propose to truncate state transitions for those uncertain state-action pairs. Nevertheless, naive pessimism-based methods may behave too conservatively, leading to inferior performance. In contrast, our LP-MDP construction allows the offline RL agent to embark on a less conservative path, while guaranteeing a even better performance lower bound.

Lookahead in RL. The concept of K-step lookahead has been considered in general RL algorithms [Efroni et al., 2020, El Shar and Jiang, 2020, Sikchi et al., 2022, Rosenberg et al., 2023], yet the goal is mainly to aid model-free planning. Specifically, Sikchi et al. [2022] propose to learn a model for looking ahead into the future (*i.e.*, imaginary model rollouts) to choose the best action sequence, compensating for the inaccuracy of the value function for typical model-free agents. To our knowledge, our work is the first to employ the concept of lookahead in model-based offline RL. Importantly, compared to prior methods relying on model rollouts, our lookahead set construction leads to an *extrinsic* pessimistic path (as illustrated in Figure 2) in the offline scenario.

6 Conclusion

We have presented a novel class of pessimistic MDPs with lookahead (LP-MDPs) for model-based offline RL. LP-MDPs learn a less conservative policy yet with remarkable performance guarantees in the real environment. Particularly, we prove that the guarantee can be improved by simply increasing the lookahead horizon, enabling a better guarantee than the baseline's [Kidambi et al., 2020]. LP-MDPs are straightforward to implement and have demonstrated a consistent performance improvement for both (discrete) gridworld tasks and (continuous) locomotion control tasks. One promising future direction is offline-to-online RL [Nair et al., 2020, Schrittwieser et al., 2021], where our lookahead concept naturally kicks in as a bridge.

References

Arthur Argenson and Gabriel Dulac-Arnold. Model-based offline planning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=OMNB1G5xzd4.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Xiong-Hui Chen, Yang Yu, Qingyang Li, Fan-Ming Luo, Zhiwei Qin, Wenjie Shang, and Jieping Ye. Offline model-based adaptable policy learning. *Advances in Neural Information Processing Systems*, 34:8432–8443, 2021.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

Yonathan Efroni, Mohammad Ghavamzadeh, and Shie Mannor. Online planning with lookahead policies. *Advances in Neural Information Processing Systems*, 33:14024–14033, 2020.

Ibrahim El Shar and Daniel Jiang. Lookahead-bounded q-learning. In *International Conference on Machine Learning*, pages 8665–8675. PMLR, 2020.

Benjamin Eysenbach, Alexander Khazatsky, Sergey Levine, and Ruslan Salakhutdinov. Joint model-policy optimization of a lower bound for model-based RL. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=LYfFj-Vk6lt.

Xing Fang, Qichao Zhang, Yinfeng Gao, and Dongbin Zhao. Offline reinforcement learning for autonomous driving with real world driving data. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 3417–3422. IEEE, 2022.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pages 2052–2062. PMLR, 2019.

Dibya Ghosh, Anurag Ajay, Pulkit Agrawal, and Sergey Levine. Offline rl policies should be trained to be adaptive. In *International Conference on Machine Learning*, pages 7513–7530. PMLR, 2022.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

Matt Hoffman, Bobak Shahriari, John Aslanides, Gabriel Barth-Maron, Feryal Behbahani, Tamara Norman, Abbas Abdolmaleki, Albin Cassirer, Fan Yang, Kate Baumli, et al. Acme: A research framework for distributed reinforcement learning. *arXiv preprint arXiv:2006.00979*, 2020.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, pages 2137–2143. PMLR, 2020.

Ying Jin, Zhuoran Yang, and Zhaoran Wang. Is pessimism provably efficient for offline rl? In *International Conference on Machine Learning*, pages 5084–5096. PMLR, 2021.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33:21810–21823, 2020.

Diederick P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=68n2s9ZJWF8.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 32, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

Aviral Kumar, Joey Hong, Anikait Singh, and Sergey Levine. Should i run offline reinforcement learning or behavioral cloning? In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=AP1MKT37rJ.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Cong Lu, Philip Ball, Jack Parker-Holder, Michael Osborne, and Stephen J. Roberts. Revisiting design choices in offline model based reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=zz9hXVhf40.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

Aravind Rajeswaran, Igor Mordatch, and Vikash Kumar. A game theoretic framework for model based reinforcement learning. In *International conference on machine learning*, pages 7953–7963. PMLR, 2020.

Marc Rigter, Bruno Lacerda, and Nick Hawes. RAMBO-RL: Robust adversarial model-based offline reinforcement learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=nrksGSRT7kX.

Aviv Rosenberg, Assaf Hallak, Shie Mannor, Gal Chechik, and Gal Dalal. Planning and learning with adaptive lookahead. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2023.

Julian Schrittwieser, Thomas Hubert, Amol Mandhane, Mohammadamin Barekatain, Ioannis Antonoglou, and David Silver. Online and offline reinforcement learning by planning with a learned model. *Advances in Neural Information Processing Systems*, 34:27580–27591, 2021.

Harshit Sikchi, Wenxuan Zhou, and David Held. Learning off-policy with online planning. In *Conference on Robot Learning*, pages 1622–1633. PMLR, 2022.

Masatoshi Uehara and Wen Sun. Pessimistic model-based offline reinforcement learning under partial coverage. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=tyrJsbKAe6.

Heinrich Von Stackelberg. *Market structure and equilibrium*. Springer Science & Business Media, 2010.

Lu Wang, Wei Zhang, Xiaofeng He, and Hongyuan Zha. Supervised reinforcement learning with recurrent neural network for dynamic treatment recommendation. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2447–2456, 2018.

Ziyu Wang, Alexander Novikov, Konrad Zolna, Josh S Merel, Jost Tobias Springenberg, Scott E Reed, Bobak Shahriari, Noah Siegel, Caglar Gulcehre, Nicolas Heess, et al. Critic regularized regression. *Advances in Neural Information Processing Systems*, 33:7768–7778, 2020.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967, 2021.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [No]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes, code will be released. Other details are explained in the supplemental material.]

(b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]

(c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes, we used TPUs.]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

   (a) Citations of the creator If your work uses existing assets. [Yes]

   (b) The license information of the assets, if applicable. [Not Applicable]

   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]

   (d) Information about consent from data providers/curators. [Not Applicable]

   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

   (a) The full text of instructions given to participants and screenshots. [Not Applicable]

   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]

   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Algorithm 1 Policy Learning under Pessimistic MDP with look-ahead (LP(ξ, π, K)-MDP).

Require: Offline data $\mathcal{D}$, threshold ξ, look-ahead steps K, and learning rate α.
1: Fit the dynamics model $\mathcal{M}_{\widehat{p}}$ on $\mathcal{D}$
2: Initialize the policy π_0
3: **for** $n = 0, 1, 2, \ldots$ **do**
4: Construct the LP(ξ, π, K)-MDP for π_n: $\mathcal{M}_{\widetilde{p}}^{\pi_n}$
5: Improve policy: $\pi_{n+1} \leftarrow \texttt{SAC\_Step}(\pi_n, \mathcal{M}_{\widetilde{p}}^{\pi_n})$
6: **end for**
7: **return** π_n

A Appendix

A.1 Limitations

The primary limitation in our research lies in the presumption that we can access the total variation distance between the fitted transition distribution and the true transition distribution, an assumption that may not be feasible in practical situations. Consequently, we resort to a surrogate for the total variation distance, specifically, we adopt a disagreement-based metric. This method, previously utilized in extant literature [Kidambi et al., 2020, Yu et al., 2020], has proven effective in actual implementations. Nonetheless, in contrast to methods like MOPO [Yu et al., 2020] and COMBO [Yu et al., 2021], where a substantial divergence exists between the theoretical constructs and their real-world algorithms (for instance, they only execute the model for a limited number of steps, starting from the offline data), our algorithm exhibits a tighter correlation with the theoretical version, exhibiting minimal divergences.

A.2 Extended Related Works

Model-Free Offline RL. The typical practice for such work is to regularize the learned policy or the value function [Kumar et al., 2019, Wu et al., 2019, Kumar et al., 2020, Wang et al., 2020, Jin et al., 2021, Kostrikov et al., 2022]. For example, Fujimoto et al. [2019], Wu et al. [2019], Kumar et al. [2019], Wang et al. [2020] propose to *regularize the learned policy* so that it stays close to the behavior policy; Kumar et al. [2020], Kostrikov et al. [2022] consider to *regularize the value function* by penalizing rewards for out-of-distribution state action pairs. Some recent work also considers to adaptively train policies under a Bayesian perspective [Ghosh et al., 2022]. However, model-free approaches can have high sample complexity or unstable convergence properties, or are weak at generalizing to unseen data, which leads to the rising of model-based approaches.

A.3 Definitions

We first restate the definitions from the main paper to make them more easily referenced for the readers.

Definition 1 (Uncertain State-Action Set) *$\forall\ \xi \geq 0$, the set of uncertain state-action pairs is*

$$\mathcal{U}(\xi) \coloneqq \{(\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{S} \times \mathcal{A} : d(\boldsymbol{s}, \boldsymbol{a}) \geq \xi\}\,. \tag{6}$$

Definition 2 (Lookahead Set) *For any $(\boldsymbol{s}_t, \boldsymbol{a}_t) \in \mathcal{S} \times \mathcal{A}$ under MDP $\mathcal{M}_{p'}$ with $p'(\cdot|\cdot, \cdot, \pi)$ denoting the transition distribution relying on π, the k_{th}-step lookahead set is*

$$\mathcal{L}_{\mathcal{M}_{p'}}^{\pi,k}(\boldsymbol{s}_t, \boldsymbol{a}_t) \coloneqq \{\boldsymbol{s} \in \mathcal{S} |\ p'(\boldsymbol{s}_{t+k} = \boldsymbol{s} | \boldsymbol{s}_t, \boldsymbol{a}_t, \pi) > 0\}\,.$$

Definition 3 (LP(ξ, π, K)-MDP) *For any $(\boldsymbol{s}_t, \boldsymbol{a}_t)$, and $\mathcal{M}_{\widehat{p}}$ and $\mathcal{M}_p$, the LP(ξ, π, K)-MDP of $\mathcal{M}_p$, (i.e., $\mathcal{M}_{\widetilde{p}}^{\pi}$) is constructed by modifying the transition $\widehat{p}$ to be $\widetilde{p}$ as:*[5]

<u>**Case 1**</u> *(current transition is* certain*) If $(\boldsymbol{s}_t, \boldsymbol{a}_t) \notin \mathcal{U}$:*

$$\widetilde{p}(\cdot|\boldsymbol{s}_t, \boldsymbol{a}_t) = \widehat{p}(\cdot|\boldsymbol{s}_t, \boldsymbol{a}_t), \tag{8}$$

<u>**Case 2**</u> *(current transition is* uncertain*) If $(\boldsymbol{s}_t, \boldsymbol{a}_t) \in \mathcal{U}$:*

[5]The associated reward $\tilde{r}(\cdot) := r(\cdot)$ unless otherwise stated.

Case 2.1 *(all K-step look ahead is uncertain) If $(\boldsymbol{s}_t, \boldsymbol{a}_t) \in \mathcal{U}^{\pi}_{-(1:K)}$:*

$$\widetilde{p}(\boldsymbol{s}_{t+1} = \mathbf{e} | \boldsymbol{s}_t, \boldsymbol{a}_t) = 1, \tag{9}$$

Case 2.2 *(some k_{th}-step look ahead is certain) If $(\boldsymbol{s}_t, \boldsymbol{a}_t) \in \mathcal{U}^{\pi}_{k}$ for some $k \in [K]$, then we construct a deterministic path such that $\forall\, i \in \{1, ..., k-1\}$,*

$$\widetilde{p}(\boldsymbol{s}_{t+k} = \boldsymbol{s}^\star | \boldsymbol{s}_t, \boldsymbol{a}_t, \pi) = 1,\ \widetilde{r}(\boldsymbol{s}_{t+i}, \cdot) = -R. \tag{10}$$

where $\boldsymbol{s}^\star$ is defined as: $\boldsymbol{s}^\star = \arg\min_{\boldsymbol{s}' \in \mathcal{L}^{\pi,k}_{\mathcal{M}_{\widehat{p}}}(\boldsymbol{s}_t, \boldsymbol{a}_t)} V^{\pi}_{\mathcal{M}^{\pi}_{\widetilde{p}}}(\boldsymbol{s}')$,

Definition 4 (Hitting Time & Occupancy Measure) *Given an MDP $\mathcal{M}_p$, a policy π, a state action pair $(\boldsymbol{s}, \boldsymbol{a})$ and an initial state $\boldsymbol{s}_0$, we define*

A.4 Proof of Theorems

Theorem 1 (Performance Guarantee) *Let $\widetilde{\pi}^\star$ denote the equilibrium policy learned under the LP($\xi, \widetilde{\pi}^\star, K$)-MDP, and let $\pi^\star$ be the optimal policy under the true MDP $\mathcal{M}_p$. Suppose that for any $(\boldsymbol{s}_t, \boldsymbol{a}_t) \in \mathcal{U}^{\widetilde{\pi}^\star}_{k}$ with $k \in [1, K]$, $\mathcal{L}^{\widetilde{\pi}^\star, k}_{\mathcal{M}_p}(\boldsymbol{s}_t, \boldsymbol{a}_t) \subseteq \mathcal{L}^{\widetilde{\pi}^\star, k}_{\mathcal{M}_{\widehat{p}}}(\boldsymbol{s}_t, \boldsymbol{a}_t)$ holds, then for any state $\boldsymbol{s}$*

$$\begin{aligned}
&V^{\pi^\star}_{\mathcal{M}_p}(\boldsymbol{s}) - V^{\widetilde{\pi}^\star}_{\mathcal{M}_p}(\boldsymbol{s}) \\
&\le \underbrace{\frac{4\gamma\xi R_{\max}}{(1-\gamma)^2}}_{\text{(a)}} + \underbrace{\frac{2\rho^{\pi^\star}_{\boldsymbol{s} \to \mathcal{U}^{\widetilde{\pi}^\star}_{-(1:K)}} \mathbb{E}\left[\gamma^{\mathcal{T}^{\pi^\star}_{\boldsymbol{s} \to \mathcal{U}^{\widetilde{\pi}^\star}_{-(1:K)}}}\right] R_{\max}}{1-\gamma}}_{\text{(b) Incurred by hitting } \mathcal{U}^{\widetilde{\pi}^\star}_{-(1:K)}} \\
&+ \underbrace{\sum_{k=1}^{K} \rho^{\pi^\star}_{\boldsymbol{s} \to \mathcal{U}^{\widetilde{\pi}^\star}_{k}} \mathbb{E}\left[\gamma^{\mathcal{T}^{\pi^\star}_{\boldsymbol{s} \to \mathcal{U}^{\widetilde{\pi}^\star}_{k}}}\right] \left(\textstyle\sum_{i=1}^{k-1} 2\gamma^i R_{\max} + \gamma^k \Delta^{\pi^\star, k}_{p, \widetilde{q}}(\mathcal{U}^{\widetilde{\pi}^\star}_{k})\right)}_{\text{(c) Incurred by hitting } \mathcal{U}^{\widetilde{\pi}^\star}_{k} \text{ for } k \in [K]}.
\end{aligned} \tag{13}$$

Proof: Our proof relies on bounding the difference in value function caused by hitting those state action pairs in each groups separately. Specifically, for those state actions pairs that are not in $\mathcal{U}$, the fitted transition distributions on them have a small total variation distance with the true distribution. Therefore, the simulation lemma can be applied to characterize the difference. For the state action pairs that are in $\mathcal{U}$, we will need to take the effect of pessimism into account.

Recall the definition of the following sets for a policy π,

$$k = 0: \quad \mathcal{U} \coloneqq \{(\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{S} \times \mathcal{A} : d(\boldsymbol{s}, \boldsymbol{a}) \ge \xi\}, \tag{20}$$

$$K \ge k \ge 1: \quad \mathcal{U}^{\pi}_{k} \coloneqq \left\{ (\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{U} \setminus \cup_{i=1}^{k-1} \mathcal{U}^{\pi}_{i} : \underbrace{d(\boldsymbol{s}', \pi(\boldsymbol{s}')) \le \xi \ \ \forall\, \boldsymbol{s}' \in \mathcal{L}^{\pi}_{\mathcal{M}_{\widehat{p}}, k}(\boldsymbol{s}, \boldsymbol{a})}_{(\boldsymbol{s}, \boldsymbol{a}) \text{ is } k_{th} \text{ certain.}} \right\}, \tag{21}$$

$$k = K+1: \quad \mathcal{U}^{\pi}_{K+1} \coloneqq \mathcal{U} \setminus \cup_{i=1}^{K} \mathcal{U}^{\pi}_{i}. \tag{22}$$

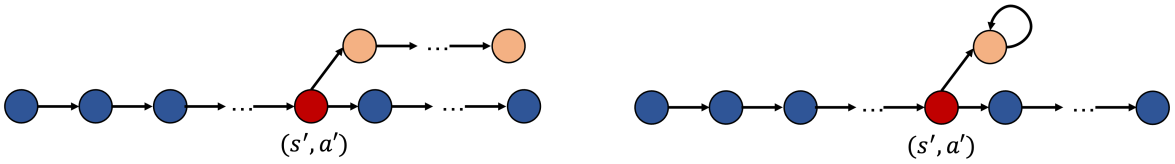
In addition, we also rely on the following intermediate transition distribution $\widetilde{q}$ for derivation.

$$\forall\, (\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{U}: \quad \widetilde{q}(\cdot | \boldsymbol{s}, \boldsymbol{a}) = \widetilde{p}(\cdot | \boldsymbol{s}, \boldsymbol{a}), \tag{23}$$

$$\text{Otherwise}: \quad \widetilde{q}(\cdot | \boldsymbol{s}, \boldsymbol{a}) = p(\cdot | \boldsymbol{s}, \boldsymbol{a}). \tag{24}$$

The MDP $\mathcal{M}^{\pi}_{\widetilde{q}}$ has the same transition distribution with the true MDP $\mathcal{M}_p$ except for those state action pairs in $\mathcal{U}$, which follows the same transition as $\widetilde{p}$. For any $(\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{S} \times \mathcal{A}$, we have $\mathbf{d}_{\text{TV}}\left(\widetilde{q}(\cdot | \boldsymbol{s}, \boldsymbol{a}) | \widetilde{p}(\cdot | \boldsymbol{s}, \boldsymbol{a})\right) \le \xi$. This gives the following bound by applying the simulation lemma,

$$\left| V^{\pi'}_{\mathcal{M}^{\pi}_{\widetilde{p}}}(\boldsymbol{s}) - V^{\pi'}_{\mathcal{M}^{\pi}_{\widetilde{q}}}(\boldsymbol{s}) \right| \le \frac{2\gamma\xi R_{\max}}{(1-\gamma)^2}. \tag{25}$$



(a) $(s', a') \in \mathcal{U}_k^\pi$ with $k \le K$ (b) $(s', a') \in \mathcal{U}_{K+1}^\pi$

Figure 8: An illustration of two cases that will result in a difference in the value functions of the true MDP and the LP-MDP. Two MDPs have the same transitions on all blue nodes, and differs on the red node, which will result in a difference in the reward to go. For (b), since $(\boldsymbol{s}', \boldsymbol{a}') \in \mathcal{U}_{K+1}^\pi$, the next state will be replaced by an absorbing state $\mathbf{e}$.

In the next, we bound the difference between $V_{\mathcal{M}_{\widetilde{q}}^\pi}^{\pi'}(\boldsymbol{s})$ and $V_{\mathcal{M}_p}^{\pi'}(\boldsymbol{s})$. To be noted, for all the state action pairs that are not in $\mathcal{U}$, we have $\widetilde{q} = p$. Therefore, we only need to deal with those state action pairs that are in $\mathcal{U}$. If the policy π hits any state action pairs $(\boldsymbol{s}', \boldsymbol{a}') \in \mathcal{U}_k^\pi$ with $1 \le k \le K$, it will incur at most the following expected difference between the total rewards due to hitting $(\boldsymbol{s}', \boldsymbol{a}')$ (see Figure 8 (a) for an illustration),

$$\mathbb{E}\left[\gamma^{\mathcal{T}_{\boldsymbol{s}\to(\boldsymbol{s}',\boldsymbol{a}')}^{\pi'}}\right] \cdot \underbrace{\left(\sum_{i=1}^{k-1} 2\gamma^i R_{\max} + \gamma^k \Delta_{p,\widetilde{q}}^{\pi',k}(\boldsymbol{s}', \boldsymbol{a}')\right)}_{\le 2R_{\max}/(1-\gamma)}. \tag{26}$$

Lastly, the difference incurred by hitting any state action pair $(\boldsymbol{s}', \boldsymbol{a}') \in \mathcal{U}_{-(1:K)}^\pi$ is trivially upper bounded by the property of the absorbing state (see Figure 8 (b)),

$$\frac{2R_{\max}\mathbb{E}\left[\gamma^{\mathcal{T}_{\boldsymbol{s}\to(\boldsymbol{s}',\boldsymbol{a}')}^{\pi'}}\right]}{1-\gamma}. \tag{27}$$

Combining the above two results, we can have the following result for $\mathcal{M}_{\widetilde{q}}$ and $\mathcal{M}_p$ for any running policy π',

$$\begin{aligned}\left|V_{\mathcal{M}_p}^{\pi'}(\boldsymbol{s}) - V_{\mathcal{M}_{\widetilde{q}}^\pi}^{\pi'}(\boldsymbol{s})\right| \le & \frac{2\rho_{\boldsymbol{s}\to\mathcal{U}_{-(1:K)}^\pi}^{\pi'}\mathbb{E}\left[\gamma^{\mathcal{T}_{\boldsymbol{s}\to\mathcal{U}_{-(1:K)}^\pi}^{\pi'}}\right] R_{\max}}{1-\gamma} \\ & + \sum_{k=1}^{K} \rho_{\boldsymbol{s}\to\mathcal{U}_k^\pi}^{\pi'}\mathbb{E}\left[\gamma^{\mathcal{T}_{\boldsymbol{s}\to\mathcal{U}_k^\pi}^{\pi'}}\right] \cdot \left(\sum_{i=1}^{k-1} 2\gamma^i R_{\max} + \gamma^k \Delta_{p,\widetilde{q}}^{\pi',k}(\mathcal{U}_k^\pi)\right).\end{aligned} \tag{28}$$

In the next, we give a performance bound for the equilibrium or optimal policy $\widetilde{\pi}^\star$ of the LP$(\xi, \widetilde{\pi}^\star, K)$-MDP under the true MDP $\mathcal{M}_p$. We further denote $\pi^\star$ as the optimal policy under $\mathcal{M}_p$.

$$V_{\mathcal{M}_p}^{\pi^\star}(\boldsymbol{s}) - V_{\mathcal{M}_p}^{\widetilde{\pi}^\star}(\boldsymbol{s}) \tag{29}$$

$$= V_{\mathcal{M}_p}^{\pi^\star}(\boldsymbol{s}) - V_{\mathcal{M}_{\widetilde{q}}^{\widetilde{\pi}^\star}}^{\pi^\star}(\boldsymbol{s}) + V_{\mathcal{M}_{\widetilde{q}}^{\widetilde{\pi}^\star}}^{\pi^\star}(\boldsymbol{s}) - V_{\mathcal{M}_p}^{\widetilde{\pi}^\star}(\boldsymbol{s}) \tag{30}$$

$$= V_{\mathcal{M}_p}^{\pi^\star}(\boldsymbol{s}) - V_{\mathcal{M}_{\widetilde{q}}^{\widetilde{\pi}^\star}}^{\pi^\star}(\boldsymbol{s}) + V_{\mathcal{M}_{\widetilde{q}}^{\widetilde{\pi}^\star}}^{\pi^\star}(\boldsymbol{s}) - V_{\mathcal{M}_{\widetilde{p}}^{\widetilde{\pi}^\star}}^{\pi^\star}(\boldsymbol{s}) + V_{\mathcal{M}_{\widetilde{q}}^{\widetilde{\pi}^\star}}^{\widetilde{\pi}^\star}(\boldsymbol{s}) - V_{\mathcal{M}_p}^{\widetilde{\pi}^\star}(\boldsymbol{s}) \tag{31}$$

$$\le V_{\mathcal{M}_p}^{\pi^\star}(\boldsymbol{s}) - V_{\mathcal{M}_{\widetilde{q}}^{\widetilde{\pi}^\star}}^{\pi^\star}(\boldsymbol{s}) + V_{\mathcal{M}_{\widetilde{q}}^{\widetilde{\pi}^\star}}^{\pi^\star}(\boldsymbol{s}) - V_{\mathcal{M}_{\widetilde{p}}^{\widetilde{\pi}^\star}}^{\pi^\star}(\boldsymbol{s}) + V_{\mathcal{M}_{\widetilde{p}}^{\widetilde{\pi}^\star}}^{\widetilde{\pi}^\star}(\boldsymbol{s}) - V_{\mathcal{M}_{\widetilde{q}}^{\widetilde{\pi}^\star}}^{\widetilde{\pi}^\star}(\boldsymbol{s}) + V_{\mathcal{M}_{\widetilde{q}}^{\widetilde{\pi}^\star}}^{\widetilde{\pi}^\star}(\boldsymbol{s}) - V_{\mathcal{M}_p}^{\widetilde{\pi}^\star}(\boldsymbol{s}) \tag{32}$$

$$= \underbrace{V_{\mathcal{M}_p}^{\pi^\star}(\boldsymbol{s}) - V_{\mathcal{M}_{\widetilde{q}}^{\widetilde{\pi}^\star}}^{\pi^\star}(\boldsymbol{s})}_{\text{Bnouded by equation 28}} + \underbrace{V_{\mathcal{M}_{\widetilde{q}}^{\widetilde{\pi}^\star}}^{\pi^\star}(\boldsymbol{s}) - V_{\mathcal{M}_{\widetilde{p}}^{\widetilde{\pi}^\star}}^{\pi^\star}(\boldsymbol{s})}_{\text{Bounded by equation 25}} + V_{\mathcal{M}_{\widetilde{p}}^{\widetilde{\pi}^\star}}^{\widetilde{\pi}^\star}(\boldsymbol{s}) - V_{\mathcal{M}_p}^{\widetilde{\pi}^\star}(\boldsymbol{s}) \tag{33}$$

The first inequality is due to the fact that $\widetilde{\pi}^\star$ is the optimal policy of $\mathcal{M}_{\widetilde{p}}^{\widetilde{\pi}^\star}$, and hence

$$V_{\mathcal{M}_{\widetilde{p}}^{\widetilde{\pi}^\star}}^{\widetilde{\pi}^\star}(\boldsymbol{s}) - V_{\mathcal{M}_{\widetilde{p}}^{\widetilde{\pi}^\star}}^{\pi^\star}(\boldsymbol{s}) \ge 0. \tag{34}$$

In the next, we bound the following difference

$$V_{\mathcal{M}_{\widetilde{p}}^{\widetilde{\pi}^\star}}^{\widetilde{\pi}^\star}(\boldsymbol{s}) - V_{\mathcal{M}_p}^{\widetilde{\pi}^\star}(\boldsymbol{s}) \tag{35}$$

To bound the above difference, we further introduce another MDP $\mathcal{M}^{\pi}_{\widetilde{q}_{\min}}$, which is similar to $\mathcal{M}^{\pi}_{\widetilde{q}}$, except that for any $(\boldsymbol{s}_t, \boldsymbol{a}_t) \in \mathcal{U}^{\pi}_k$ with $k \in [1, K]$, we set the distribution of $\widetilde{q}_{\min}$ as

$$\widetilde{q}_{\min}(\boldsymbol{s}_{t+k}|\boldsymbol{s}_t, \boldsymbol{a}_k, \pi) = \underset{\boldsymbol{s}' \in \mathcal{L}^{\pi,k}_{\mathcal{M}^{\pi}_{\widetilde{p}}}(\boldsymbol{s}_t, \boldsymbol{a}_t)}{\arg\min} V^{\pi}_{\mathcal{M}^{\pi}_{\widetilde{q}_{\min}}}(\boldsymbol{s}') \tag{36}$$

Since for any $(\boldsymbol{s}_t, \boldsymbol{a}_t) \in \mathcal{U}^{\widetilde{\pi}^\star}_k$ with $k \in [1, K]$, $\mathcal{L}^{\widetilde{\pi}^\star,k}_{\mathcal{M}_p}(\boldsymbol{s}_t, \boldsymbol{a}_t) \subseteq \mathcal{L}^{\widetilde{\pi}^\star,k}_{\mathcal{M}_{\widehat{p}}}(\boldsymbol{s}_t, \boldsymbol{a}_t)$ holds, then the following holds trivially,

$$V^{\widetilde{\pi}^\star}_{\mathcal{M}^{\widetilde{\pi}^\star}_{\widetilde{q}_{\min}}}(\boldsymbol{s}) \leq V^{\widetilde{\pi}^\star}_{\mathcal{M}_p}(\boldsymbol{s}). \tag{37}$$

Similarly, we define $\mathcal{M}^{\pi}_{\widetilde{p}'}$, such that

$$\forall\, (\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{U}: \quad \widetilde{p}'(\cdot|\boldsymbol{s}, \boldsymbol{a}) = \widetilde{q}_{\min}(\cdot|\boldsymbol{s}, \boldsymbol{a}), \tag{38}$$

$$\text{Otherwise}: \quad \widetilde{p}'(\cdot|\boldsymbol{s}, \boldsymbol{a}) = \widehat{p}(\cdot|\boldsymbol{s}, \boldsymbol{a}). \tag{39}$$

Then, we will have the following holds,

$$V^{\widetilde{\pi}^\star}_{\mathcal{M}^{\widetilde{\pi}^\star}_{\widetilde{p}}}(\boldsymbol{s}) - V^{\widetilde{\pi}^\star}_{\mathcal{M}_p}(\boldsymbol{s}) \tag{40}$$

$$= V^{\widetilde{\pi}^\star}_{\mathcal{M}^{\widetilde{\pi}^\star}_{\widetilde{p}}}(\boldsymbol{s}) - V^{\widetilde{\pi}^\star}_{\mathcal{M}^{\widetilde{\pi}^\star}_{\widetilde{q}_{\min}}}(\boldsymbol{s}) + V^{\widetilde{\pi}^\star}_{\mathcal{M}^{\widetilde{\pi}^\star}_{\widetilde{q}_{\min}}}(\boldsymbol{s}) \tag{41}$$

$$\leq V^{\widetilde{\pi}^\star}_{\mathcal{M}^{\widetilde{\pi}^\star}_{\widetilde{p}'}}(\boldsymbol{s}) - V^{\widetilde{\pi}^\star}_{\mathcal{M}^{\widetilde{\pi}^\star}_{\widetilde{q}_{\min}}}(\boldsymbol{s}) + \underbrace{V^{\widetilde{\pi}^\star}_{\mathcal{M}^{\widetilde{\pi}^\star}_{\widetilde{q}_{\min}}}(\boldsymbol{s}) - V^{\widetilde{\pi}^\star}_{\mathcal{M}_p}(\boldsymbol{s})}_{\leq 0} \tag{42}$$

$$\leq \frac{2\gamma\xi R_{\max}}{(1-\gamma)^2}. \tag{43}$$

The first inequality holds because by definition of LP-MDP, which is the minimizer of the value function. The last inequality holds because the two MDPs has the same transition on those state action in $\mathcal{U}$, whereas for all the other state action pairs, their total variation distance is at most ξ. Hence, we can obtain the result by simply applying the simulation lemma.

Now, we combine the above results and obtain the final bound

$$V^{\pi^\star}_{\mathcal{M}_p}(\boldsymbol{s}) - V^{\widetilde{\pi}^\star}_{\mathcal{M}_p}(\boldsymbol{s}) \tag{44}$$

$$\leq \frac{4\gamma\xi R_{\max}}{(1-\gamma)^2} + \frac{2\rho^{\pi^\star}_{\boldsymbol{s}\to\mathcal{U}^{\widetilde{\pi}^\star}_{-(1:K)}} \mathbb{E}\left[\gamma^{\mathcal{T}^{\pi^\star}_{\boldsymbol{s}\to\mathcal{U}^{\widetilde{\pi}^\star}_{-(1:K)}}}\right] R_{\max}}{1-\gamma}$$

$$+ \sum_{k=1}^{K} \rho^{\pi^\star}_{\boldsymbol{s}\to\mathcal{U}^{\widetilde{\pi}^\star}_k} \mathbb{E}\left[\gamma^{\mathcal{T}^{\pi^\star}_{\boldsymbol{s}\to\mathcal{U}^{\widetilde{\pi}^\star}_k}}\right] \cdot \left(\sum_{i=1}^{k-1} 2\gamma^i R_{\max} + \gamma^k \Delta^{\pi^\star,k}_{p,\widetilde{q}}(\mathcal{U}^{\widetilde{\pi}^\star}_k)\right) \tag{45}$$

□

A.5 Additional Theorems

Theorem 2 (Policy Improvement Theorem for LP-MDP) *Under the tabular case, for any two policies π and π', if $Q^{\pi}_{\mathcal{M}^{\pi}_{\widetilde{p}}}(\boldsymbol{s}, \pi'(\boldsymbol{s})) \geq V^{\pi}_{\mathcal{M}^{\pi}_{\widetilde{p}}}(\boldsymbol{s})$ for any states $\boldsymbol{s} \in \mathcal{S}$, then we will have $V^{\pi'}_{\mathcal{M}^{\pi'}_{\widetilde{p}}}(\boldsymbol{s}) \geq V^{\pi}_{\mathcal{M}^{\pi}_{\widetilde{p}}}(\boldsymbol{s})$ hold for all the states $\boldsymbol{s} \in \mathcal{S}$ if for any $(\boldsymbol{s}_i, \boldsymbol{a}_i), (\boldsymbol{s}_j, \boldsymbol{a}_j) \in \mathcal{U}$, the expected hitting time from $\boldsymbol{s}_i$ to $\boldsymbol{s}_j$ is $+\infty$ for any polices.*

Proof: The only difference between $\mathcal{M}^{\pi'}_{\widetilde{p}}$ and $\mathcal{M}^{\pi}_{\widetilde{p}}$ is how are the transition distributions $\hat{p}(\cdot|\boldsymbol{s}, \boldsymbol{a})$ with $(\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{U}$ modified based on the value functions of policy π' and π. Recall that the definition of $\mathcal{U}$ is

$$\mathcal{U} \coloneqq \{(\boldsymbol{s}, \boldsymbol{a}) \in \mathcal{S} \times \mathcal{A} : d(\boldsymbol{s}, \boldsymbol{a}) \geq \xi\}. \tag{46}$$

Consider the following process of constructing a sequence of MDPs such that $\mathcal{M}_0 = \mathcal{M}^{\pi}_{\widetilde{p}}$ and $\mathcal{M}_{|\mathcal{U}|} = \mathcal{M}^{\pi'}_{\widetilde{p}}$,

1. Initialize $i = 1$, and $\mathcal{U}^{(i)} = \mathcal{U}$;
2. Randomly choose an element $(\boldsymbol{s}^i, \boldsymbol{a}^i) \in \mathcal{U}^{(i)}$ and set $\mathcal{U}^{(i+1)} = \mathcal{U}^{(i)} \setminus \{(\boldsymbol{s}^i, \boldsymbol{a})\}$;
3. Change the transition on $(\boldsymbol{s}, \boldsymbol{a})$ of $\mathcal{M}_i$ to be exactly the same as it in $\mathcal{M}_{\widetilde{p}}^{\pi'}$;
4. Increment i by 1 and repeat the steps 2-4 until $i = |\mathcal{U}|$.

The above process modifies the transition on one state action pair at each iteration. Suppose that the resulting sequence of state action pairs is $\{(\boldsymbol{s}^i, \boldsymbol{a}^i)\}_{i=1}^{|\mathcal{U}|}$ (corresponds to step 2). Since $\mathcal{M}_1$ only modifies the transition on $(\boldsymbol{s}^1, \boldsymbol{a}^1)$ to be same as it in $\mathcal{M}_{\widetilde{p}}^{\pi'}$, and by the assumption that the probability of revisiting state $\boldsymbol{s}^1$ is 0, we have the following hold

$$p\left[\boldsymbol{s}_{t'} = s^1 | \pi, \mathcal{M}_0, \boldsymbol{s}_0\right] = p\left[\boldsymbol{s}_{t'} = s^1 | \pi, \mathcal{M}_1, \boldsymbol{s}_0\right]. \tag{47}$$

The above argument states that, under any policy π, the probability of hitting state $\boldsymbol{s}^1$ at time step t' is the same for $\mathcal{M}_0$ and $\mathcal{M}_1$. This is because the dynamics of $\mathcal{M}_0$ and $\mathcal{M}_1$ are exactly the same before hitting $\boldsymbol{s}^1$. In the same way, we can show that, for any $i \in [1, |\mathcal{U}|]$,

$$p\left[\boldsymbol{s}_{t'} = s^i | \pi, \mathcal{M}_{i-1}, \boldsymbol{s}_0\right] = p\left[\boldsymbol{s}_{t'} = s^i | \pi, \mathcal{M}_i, \boldsymbol{s}_0\right]. \tag{48}$$

We prove the theorem by induction. We first prove the base step that $V_{\mathcal{M}_0}^{\pi}(\boldsymbol{s}_0) \leq V_{\mathcal{M}_1}^{\pi'}(\boldsymbol{s}_0)$ for any $\boldsymbol{s}_0 \in \mathcal{S}$.

$$V_{\mathcal{M}_{\widetilde{p}}^{\pi}}^{\pi}(\boldsymbol{s}_0) = V_{\mathcal{M}_0}^{\pi}(\boldsymbol{s}_0) \tag{49}$$

$$\leq Q_{\mathcal{M}_0}^{\pi}(\boldsymbol{s}_0, \pi'(\boldsymbol{s}_0)) \tag{50}$$

$$= \mathbb{E}_{\boldsymbol{a}_0 \sim \pi'(\boldsymbol{a}_0|\boldsymbol{s}_0), \boldsymbol{s}_1 \sim \mathcal{M}_0|\boldsymbol{s}_0, \boldsymbol{a}_0}\left[r(\boldsymbol{s}_0, \boldsymbol{a}_0) + \gamma V_{\mathcal{M}_0}^{\pi}(\boldsymbol{s}_1)\right] \tag{51}$$

$$\leq \mathbb{E}_{\boldsymbol{a}_0 \sim \pi'(\boldsymbol{a}_0|\boldsymbol{s}_0), \boldsymbol{s}_1 \sim \mathcal{M}_0|\boldsymbol{s}_0, \boldsymbol{a}_0}\left[r(\boldsymbol{s}_0, \boldsymbol{a}_0) + \gamma Q_{\mathcal{M}_0}^{\pi}(\boldsymbol{s}_1, \pi'(\boldsymbol{s}_1))\right] \tag{52}$$

$$= \mathbb{E}_{\boldsymbol{a}_0, \boldsymbol{a}_1 \sim \pi'}\left[r(\boldsymbol{s}_0, \boldsymbol{a}_0) + \gamma r(\boldsymbol{s}_1, \boldsymbol{a}_1) + \gamma^2 V_{\mathcal{M}_0}^{\pi}(\boldsymbol{s}_2)\right] \tag{53}$$

$$\leq \dots \tag{54}$$

$$\leq \mathbb{E}_{\boldsymbol{a}_0, \boldsymbol{a}_1, \dots, \boldsymbol{a}_{t'} \sim \pi'}\left[\sum_{t=0}^{t'} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma^{t'+1} V_{\mathcal{M}_0}^{\pi}(\boldsymbol{s}_{t'+1})\right] \tag{55}$$

$$\overset{(a)}{=} \sum_{t'=0}^{\infty} p\left[\boldsymbol{s}_{t'+1} = s^1 | \pi', \mathcal{M}_0, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_0}\left[\sum_{t=0}^{t'} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma^{t'+1} V_{\mathcal{M}_0}^{\pi}(\boldsymbol{s}_{t'+1}) | \boldsymbol{s}_{t'+1} = \boldsymbol{s}^1\right] \tag{56}$$

$$+ p\left[\boldsymbol{s}_t \neq \boldsymbol{s}^1 \ \forall\, t \geq 0 | \pi', \mathcal{M}_0, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_0}\left[\sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) | \boldsymbol{s}_t \neq \boldsymbol{s}^1 \ \forall\, t \geq 0\right]$$

$$\overset{(b)}{=} \sum_{t'=0}^{\infty} p\left[\boldsymbol{s}_{t'+1} = s^1 | \pi', \mathcal{M}_1, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_1}\left[\sum_{t=0}^{t'} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma^{t'+1} V_{\mathcal{M}_0}^{\pi}(\boldsymbol{s}_{t'+1}) | \boldsymbol{s}_{t'+1} = \boldsymbol{s}^1\right] \tag{57}$$

$$+ p\left[\boldsymbol{s}_t \neq \boldsymbol{s}^1 \ \forall\, t \geq 0 | \pi', \mathcal{M}_1, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_1}\left[\sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) | \boldsymbol{s}_t \neq \boldsymbol{s}^1 \ \forall\, t \geq 0\right]$$

$$\overset{(c)}{\leq} \sum_{t'=0}^{\infty} p\left[\boldsymbol{s}_{t'+1} = s^1 | \pi', \mathcal{M}_1, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_1}\left[\sum_{t=0}^{t'} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma^{t'+1} Q_{\mathcal{M}_0}^{\pi}(\boldsymbol{s}^1, \pi'(\boldsymbol{s}^1)) | \boldsymbol{s}_{t'+1} = \boldsymbol{s}^1\right] \tag{58}$$

$$+ p\left[\boldsymbol{s}_t \neq \boldsymbol{s}^1 \ \forall\, t \geq 0 | \pi', \mathcal{M}_1, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_1}\left[\sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) | \boldsymbol{s}_t \neq \boldsymbol{s}^1 \ \forall\, t \geq 0\right]$$

$$\overset{(d)}{\leq} \sum_{t'=0}^{\infty} p\left[\boldsymbol{s}_{t'+1} = s^1 | \pi', \mathcal{M}_1, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_1}\left[\sum_{t=0}^{t'} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma^{t'+1} Q_{\mathcal{M}_1}^{\pi}(\boldsymbol{s}^1, \pi'(\boldsymbol{s}^1)) | \boldsymbol{s}_{t'+1} = \boldsymbol{s}^1\right] \tag{59}$$

$$+ p\left[\boldsymbol{s}_t \neq \boldsymbol{s}^1 \ \forall\, t \geq 0 | \pi', \mathcal{M}_1, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_1}\left[\sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) | \boldsymbol{s}_t \neq \boldsymbol{s}^1 \ \forall\, t \geq 0\right]$$

$$\overset{(e)}{\leq} \sum_{t'=0}^{\infty} p\left[\boldsymbol{s}_{t'+1} = s^1 | \pi', \mathcal{M}_1, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_1}\left[\sum_{t=0}^{t'} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma^{t'+1} Q^{\pi'}_{\mathcal{M}_1}(\boldsymbol{s}^1, \pi'(\boldsymbol{s}^1)) | \boldsymbol{s}_{t'+1} = \boldsymbol{s}^1\right] \tag{60}$$

$$+ p\left[\boldsymbol{s}_t \neq \boldsymbol{s}^1 \ \forall\, t \geq 0 | \pi', \mathcal{M}_1, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_1}\left[\sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) | \boldsymbol{s}_t \neq \boldsymbol{s}^1 \ \forall\, t \geq 0\right]$$

$$= V^{\pi'}_{\mathcal{M}_1}(\boldsymbol{s}_0). \tag{61}$$

For (a), we simply divide the trajectories into two groups, the first part of equation corresponds to the trajectories that contain state $\boldsymbol{s}^1$, and the second part of the equation corresponds to those trajectories that doesn't contain $\boldsymbol{s}^1$. The equality (b) holds because of equation 47. For (c), we use the assumption that $Q^{\pi}_{\mathcal{M}_0}(\boldsymbol{s}, \pi'(\boldsymbol{s})) \geq V^{\pi}_{\mathcal{M}_0}(\boldsymbol{s})$. In terms of (d), this is because

$$Q^{\pi}_{\mathcal{M}_1}(\boldsymbol{s}^1, \pi(\boldsymbol{s}^1)) = \mathbb{E}_{\boldsymbol{a} \sim \pi' | \boldsymbol{s}^1, \boldsymbol{s}' \sim \mathcal{M}_1 | \boldsymbol{s}^1, \boldsymbol{a}}\left[r(\boldsymbol{s}^1, \boldsymbol{a}) + \gamma V^{\pi}_{\mathcal{M}_1}(\boldsymbol{s}') | \boldsymbol{s}^1\right] \tag{62}$$

$$= \mathbb{E}_{\boldsymbol{a} \sim \pi' | \boldsymbol{s}^1, \boldsymbol{s}' \sim \mathcal{M}_1 | \boldsymbol{s}^1, \boldsymbol{a}}\left[r(\boldsymbol{s}^1, \boldsymbol{a}) + \gamma V^{\pi}_{\mathcal{M}_0}(\boldsymbol{s}') | \boldsymbol{s}^1\right] \tag{63}$$

$$\geq \mathbb{E}_{\boldsymbol{a} \sim \pi' | \boldsymbol{s}^1, \boldsymbol{s}' \sim \mathcal{M}_0 | \boldsymbol{s}^1, \boldsymbol{a}}\left[r(\boldsymbol{s}^1, \boldsymbol{a}) + \gamma V^{\pi}_{\mathcal{M}_0}(\boldsymbol{s}') | \boldsymbol{s}^1\right] \tag{64}$$

$$= Q^{\pi}_{\mathcal{M}_0}(\boldsymbol{s}^1, \pi(\boldsymbol{s}^1)). \tag{65}$$

Lastly, (e) can be obtained by the policy improvement lemma (see Lemma 1),

$$Q^{\pi}_{\mathcal{M}_1}(\boldsymbol{s}^1, \pi(\boldsymbol{s}^1)) = \mathbb{E}_{\boldsymbol{a} \sim \pi' | \boldsymbol{s}^1, \boldsymbol{s}' \sim \mathcal{M}_1 | \boldsymbol{s}^1, \boldsymbol{a}}\left[r(\boldsymbol{s}^1, \boldsymbol{a}) + \gamma V^{\pi}_{\mathcal{M}_0}(\boldsymbol{s}') | \boldsymbol{s}^1\right] \tag{66}$$

$$\leq \mathbb{E}_{\boldsymbol{a} \sim \pi' | \boldsymbol{s}^1, \boldsymbol{s}' \sim \mathcal{M}_1 | \boldsymbol{s}^1, \boldsymbol{a}}\left[r(\boldsymbol{s}^1, \boldsymbol{a}) + \gamma V^{\pi'}_{\mathcal{M}_0}(\boldsymbol{s}') | \boldsymbol{s}^1\right] \tag{67}$$

$$= \mathbb{E}_{\boldsymbol{a} \sim \pi' | \boldsymbol{s}^1, \boldsymbol{s}' \sim \mathcal{M}_1 | \boldsymbol{s}^1, \boldsymbol{a}}\left[r(\boldsymbol{s}^1, \boldsymbol{a}) + \gamma V^{\pi'}_{\mathcal{M}_1}(\boldsymbol{s}') | \boldsymbol{s}^1\right] \tag{68}$$

$$= Q^{\pi'}_{\mathcal{M}_1}(\boldsymbol{s}^1, \pi'(\boldsymbol{s}^1)) \tag{69}$$

For the induction step, let's suppose the following hold for $m \geq 1$,

$$V^{\pi}_{\mathcal{M}^{\pi}_{\tilde{p}}}(\boldsymbol{s}_0) \leq V^{\pi'}_{\mathcal{M}_m}(\boldsymbol{s}_0). \tag{70}$$

Then, it remains to prove that the above statement holds for $m+1$. To show this, we follow the same proof as in the base step.

$$V^{\pi}_{\mathcal{M}^{\pi}_{\tilde{p}}}(\boldsymbol{s}_0) = V^{\pi}_{\mathcal{M}_0}(\boldsymbol{s}_0) \tag{71}$$

$$\overset{(f)}{\leq} \sum_{t'=0}^{\infty} p\left[\boldsymbol{s}_{t'+1} = s^{m+1} | \pi', \mathcal{M}_m, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_m}\left[\sum_{t=0}^{t'} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma^{t'+1} Q^{\pi}_{\mathcal{M}_m}(\boldsymbol{s}^{m+1}, \pi'(\boldsymbol{s}^{m+1})) | \boldsymbol{s}_{t'+1} = \boldsymbol{s}^{m+1}\right] \tag{72}$$

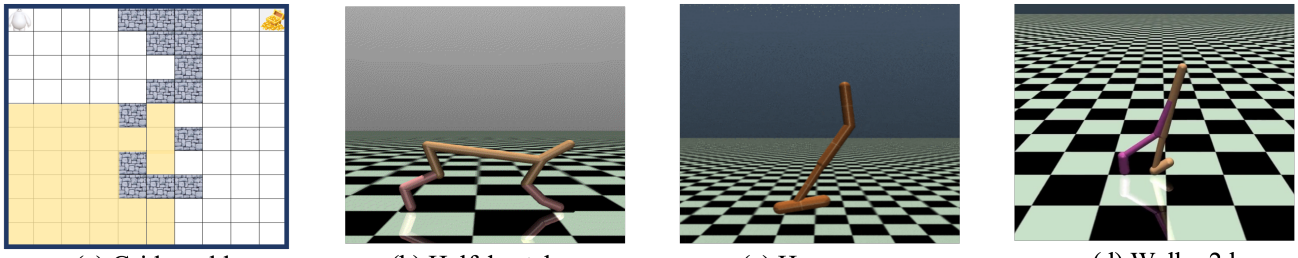
$$+ p\left[\boldsymbol{s}_t \neq \boldsymbol{s}^{m+1} \ \forall\, t \geq 0 | \pi', \mathcal{M}_m, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_m}\left[\sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) | \boldsymbol{s}_t \neq \boldsymbol{s}^{m+1} \ \forall\, t \geq 0\right]$$

$$\overset{(g)}{\leq} \sum_{t'=0}^{\infty} p\left[\boldsymbol{s}_{t'+1} = s^{m+1} | \pi', \mathcal{M}_{m+1}, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_{m+1}}\left[\sum_{t=0}^{t'} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma^{t'+1} Q^{\pi}_{\mathcal{M}_{m+1}}(\boldsymbol{s}^{m+1}, \pi'(\boldsymbol{s}^{m+1})) | \boldsymbol{s}_{t'+1} = \boldsymbol{s}^{m+1}\right]$$

$$+ p\left[\boldsymbol{s}_t \neq \boldsymbol{s}^{m+1} \ \forall\, t \geq 0 | \pi', \mathcal{M}_{m+1}, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_{m+1}}\left[\sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) | \boldsymbol{s}_t \neq \boldsymbol{s}^{m+1} \ \forall\, t \geq 0\right] \tag{73}$$

$$\overset{(h)}{\leq} \sum_{t'=0}^{\infty} p\left[\boldsymbol{s}_{t'+1} = s^{m+1} | \pi', \mathcal{M}_{m+1}, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_{m+1}}\left[\sum_{t=0}^{t'} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma^{t'+1} Q^{\pi'}_{\mathcal{M}_{m+1}}(\boldsymbol{s}^{m+1}, \pi'(\boldsymbol{s}^{m+1})) | \boldsymbol{s}_{t'+1} = \boldsymbol{s}^{m+1}\right]$$

$$+ p\left[\boldsymbol{s}_t \neq \boldsymbol{s}^{m+1} \ \forall\, t \geq 0 | \pi', \mathcal{M}_{m+1}, \boldsymbol{s}_0\right] \mathbb{E}_{\pi', \mathcal{M}_{m+1}}\left[\sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) | \boldsymbol{s}_t \neq \boldsymbol{s}^{m+1} \ \forall\, t \geq 0\right] \tag{74}$$



(a) Grid world (b) Halfcheetah (c) Hopper (d) Walker2d

Figure 9: Visualization of considered tasks. (a) The grid world task is adapted from Eysenbach et al. [2022] with increased difficulty. The agent starts where the baymax is located (top left. The reward for hitting the treasure is +50, +1 for yellow-shaded grids, and +0.5 for other grids. The walls cannot be crossed. For (b), (c) and (d), the tasks come from the D4RL benchmark [Fu et al., 2020].

$$= V^{\pi'}_{\mathcal{M}_{m+1}}(\boldsymbol{s}_0). \tag{75}$$

For the above derivations, step (f) holds by assumption of the induction step. For step (g), this is because

$$Q^{\pi}_{\mathcal{M}_{m+1}}(\boldsymbol{s}^{m+1}, \pi(\boldsymbol{s}^{m+1})) \geq \mathbb{E}_{\boldsymbol{a}\sim\pi'|\boldsymbol{s}^{m+1}, \boldsymbol{s}'\sim\mathcal{M}_m|\boldsymbol{s}^{m+1},\boldsymbol{a}}\left[r(\boldsymbol{s}^{m+1}, \boldsymbol{a}) + \gamma V^{\pi}_{\mathcal{M}_m}(\boldsymbol{s}')|\boldsymbol{s}^{m+1}\right] \tag{76}$$

$$= Q^{\pi}_{\mathcal{M}_m}(\boldsymbol{s}^{m+1}, \pi(\boldsymbol{s}^{m+1})). \tag{77}$$

The last step (h) holds because of the policy improvement lemma as well as the assumption that the hitting time from the states in $\mathcal{U}$ to each other is infinite. Hence, the dynamics after state $\boldsymbol{s}^{m+1}$ is the same as in $\mathcal{M}_0$, which concludes the proof.

□

Lemma 1 (Policy Improvement Lemma) *For any MDP $\mathcal{M}$, and any two policies π and π' and define,*

$$Q^{\pi}_{\mathcal{M}}(\boldsymbol{s}, \pi'(\boldsymbol{s})) \coloneqq \mathbb{E}_{\boldsymbol{a}\sim\pi'(\cdot|\boldsymbol{s})}\left[Q^{\pi}_{\mathcal{M}}(\boldsymbol{s}, \boldsymbol{a})\right]. \tag{78}$$

if $Q^{\pi}_{\mathcal{M}}(\boldsymbol{s}, \pi'(\boldsymbol{s})) \geq V^{\pi}_{\mathcal{M}}(\boldsymbol{s})$ for all $\boldsymbol{s} \in \mathcal{S}$, then $V^{\pi'}_{\mathcal{M}}(\boldsymbol{s}) \geq V^{\pi}_{\mathcal{M}}(\boldsymbol{s})$ for all $\boldsymbol{s} \in \mathcal{S}$.

B Computational Cost

Suppose that we use n particles for approximating the (continuous) lookahead set, then the size of the (approximated) lookahead set will grow exponentially in the lookahead steps k. Although, in our experiments, we show that a small lookahead step is sufficient to obtain significantly improved empirical results, we would like to extend the discussion on the computational cost as well as potential algorithms for reducing the computational cost.

The naive implementation will incur a computational complexity of $\mathcal{O}(n^k)$ for computing the LP-MDP for each state. However, this cannot be improved as we need to check if all the states in the k-step lookahead set are certain under the current policy. However, the number of queries of the value functions can be improved by adopting the branch and bound algorithm, though the worst case computational complexity is still $\mathcal{O}(n^k)$. In our experiments, it takes roughly several hours for training using one TPU-V3-8 for both our method and MOReL-SAC.

C Experimental Details

In this section, we introduce the details about the experiments. Our implementation is based on JAX [Bradbury et al., 2018] and Acme [Hoffman et al., 2020]. We train our models using TPU-V3-8. Our code will be released after cleanup.

C.1 Gridworld Experiments

For the gridworld experiments, we adopt the original implementation of MnM [Eysenbach et al., 2022] in https://github.com/ben-eysenbach/mnm/blob/main/experiments.ipynb.

To obtain the expert policy, we train the MnM for $1,500$ episodes where each each episode lasts for 200 steps. This gives us a replay buffer $((\boldsymbol{s}_t, \boldsymbol{a}_t, r_t, \boldsymbol{s}_{t+1}))$ of size $300,000$. Then, for our experiments, we randomly sample 20% of transitions from the replay buffer to serves as the offline training data. For training the dynamics model, since the environment is discrete, we simply fit the transition probability on each state action pair by

$$\widehat{p}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t) = \frac{n_{(\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{s}_{t+1})}}{n_{(\boldsymbol{s}_t, \boldsymbol{a}_t)}}, \tag{79}$$

where $n_{(\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{s}_{t+1})}$ and $n_{(\boldsymbol{s}_t, \boldsymbol{a}_t)}$ denotes the number of occurrence of $(\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{s}_{t+1})$ and $(\boldsymbol{s}_t, \boldsymbol{a}_t)$ in the offline dataset.

To learn the policy, we fit the Q-value function by minimizing the TD loss for $2,000$ episodes using the fitted dynamics model. The learning rate is set to be $1e-2$. Since the maximum reward is 50, we set the reward penalty to be -50. We present the code for implementing the gridworld environment in the next.

```
# Dynamics implementation

MAP = "ooooxxxooo\\" + \
      "ooooxxxooo\\" + \
      "ooooooxooo\\" + \
      "ooooxxxooo\\" + \
      "oooxooooo\\" + \
      "ooooooxooo\\" + \
      "oooxooooo\\" + \
      "ooooxxxooo\\" + \
      "oooooooooo\\" + \
      "oooooooooo"

A_DELTA = np.array([
                    [-1, 0],
                    [0, 1],
                    [1, 0],
                    [0, -1]
                    ])

WALLS = np.zeros((size, size))
p_s_given_sa = np.zeros([num_states, num_actions, num_states])

def ij_to_index(i, j):
  return i * size + j

def index_to_ij(index):
  return (index // size, index % size)

# Construct the map
def build_walls(data, grid_map, noise):
  canvas = data.copy()
  nrows, ncols = canvas.shape
  symbols = grid_map.split('\\')
  for r in range(nrows):
    for c in range(ncols):
      if symbols[r][c] == 'x':
        canvas[r][c] = 1
  return canvas

# the deterministic dynamics.
WALLS = build_walls(WALLS, MAP)
def dynamics(index, a):
  i, j = index_to_ij(index)
  (i, j) = (i, j) + A_DELTA[a]
  (i, j) = np.clip((i, j), 0, size - 1)
  if WALLS[i, j]:
    i, j = index_to_ij(index)
  index2 = ij_to_index(i, j)
  return index2

# compute the stochastic dynamics of the gridworld.
for index, a in itertools.product(range(num_states), range(num_actions)):
  if index < num_states - 1:
    index2 = dynamics(index, (a+2) % 4)
    p_s_given_sa[index, a, index2] += noise
    index2 = dynamics(index, a)
    p_s_given_sa[index, a, index2] += (1 - noise)
  else:
    p_s_given_sa[index, a, index] = 1.0
```

C.2 D4RL Experiments

For the D4RL experiments, we refer to the implementation of MOReL in https://github.com/aravindr93/mjrl/tree/v2/projects/morel for implementing MOReL-SAC and our method. For SAC, we adopt the implementation provided in Acme [Hoffman et al., 2020] and the default hyperparameters, except for the target entropy terms. For the target entropy terms, we adopt -3 for both `Halfcheetah` and `Walker2D`, and -1 for `Hopper`. We train the SAC for 1M steps.

For dynamics models, we implement it in the following way,

```
class DeterministicMLP(hk.Module):
  """MLP with DeterministicMLP outputs."""

  def __init__(
      self,
      output_size: int,
      hidden_sizes: Sequence[int],
      *,
      activation=jax.nn.swish,
      name: Optional[str] = None,
  ):
    super().__init__(name=name)
    self.output_size = output_size
    w_init = hk.initializers.VarianceScaling(1.0, 'fan_in', 'truncated_normal')
    self.mlp = hk.nets.MLP(
        hidden_sizes, w_init=w_init, activation=activation, activate_final=True)
    self.mean = hk.Linear(
        self.output_size, w_init=w_init, name='mean')

  def __call__(self, x):
    h = self.mlp(x)
    mean = self.mean(h)
    return mean
```

We determine the hyperparameters by tuning the performance of MOReL-SAC such that it can achieve a reasonable performance. Then, we adopt the same hyperparameters and only add the lookahead mechanism to the MOReL-SAC for implementing our method, which ensures that the comparison is fair.

The configurations on tasks for MOReL-SAC are summarized in the following tables. To be noted, most hyperparameters are the same across different tasks, and the only differences are in the pessimism coefficient and learning rates. For training the dynamics models, we stop the training once the loss doesn't improve for 20 epochs.

For the pessimism coefficient, we use it in the same way as suggested by MOReL [Kidambi et al., 2020]. Specifically, we compute the largest discrepancy measure on the training data (i.e., offline data) as our base unit, i.e.,

$$\max_{(\boldsymbol{s},\boldsymbol{a})\in\mathcal{D}_{\text{offline}},i\neq j} \|f_{\boldsymbol{\theta}_i}(\boldsymbol{s},\boldsymbol{a}) - f_{\boldsymbol{\theta}_j}(\boldsymbol{s},\boldsymbol{a})\|_2. \tag{80}$$

Suppose that the pessimism coefficient is c, then the threshold of the "uncertainess" of a state action pair $(\boldsymbol{s},\boldsymbol{a})$ (i.e., the cutoff for determining if the state action pair is in $\mathcal{U}$ or not) is computed by

$$c\cdot\max_{(\boldsymbol{s},\boldsymbol{a})\in\mathcal{D}_{\text{offline}},i\neq j} \|f_{\boldsymbol{\theta}_i}(\boldsymbol{s},\boldsymbol{a}) - f_{\boldsymbol{\theta}_j}(\boldsymbol{s},\boldsymbol{a})\|_2. \tag{81}$$

Parameter	Hopper			
	random	medium	medium-replay	medium-expert
Model learning rate	1e-3	1e-3	1e-3	1e-3
Optimizer	Adam	Adam	Adam	Adam
Hidden dim.	512	512	512	512
Number of Layers	4	4	4	4
Activation	Swish	Swish	Swish	Swish
Batch size	256	256	256	256
Pessimism coefficient	1.0	1.5	1.5	1.0
Reward penalty	0	0	0	0
Model horizon	700	700	700	700
Size of ensemble	4	4	4	4

Table 2: The configurations for the `Hopper` task.

Parameter	Halfcheetah			
	random	medium	medium-replay	medium-expert
Model learning rate	5e-4	1e-3	5e-4	5e-4
Optimizer	Adam	Adam	Adam	Adam
Hidden dim.	1024	1024	1024	1024
Number of Layers	4	4	4	4
Activation	Swish	Swish	Swish	Swish
Batch size	256	256	256	256
Pessimism coefficient	1.5	1.5	1.5	1.5
Reward penalty	-500	-500	-500	-500
Model horizon	1000	1000	1000	1000
Size of ensemble	10	10	10	10

Table 3: The configurations for the `Halfcheetah` task.

Parameter	Walker2D			
	random	medium	medium-replay	medium-expert
Model learning rate	5e-4	1e-3	1e-3	1e-3
Optimizer	Adam	Adam	Adam	Adam
Hidden dim.	1024	1024	1024	1024
Number of Layers	4	4	4	4
Activation	Swish	Swish	Swish	Swish
Batch size	256	256	256	256
Pessimism coefficient	2.0	1.0	1.0	1.5
Reward penalty	0	0	0	0
Model horizon	700	700	700	700
Size of ensemble	4	4	4	4

Table 4: The configurations for the `Walker2D` task.