2019 IEEE 58th Conference on Decision and Control (CDC)
Palais des Congrès et des Expositions Nice Acropolis
Nice, France, December 11-13, 2019

Approximate information state for partially observed systems

Jayakumar Subramanian and Aditya Mahajan

Abstract— **The standard approach for modeling partially observed systems is to model them as partially observable Markov decision processes (POMDPs) and obtain a dynamic program in terms of a belief state. The belief state formulation works well for planning but is not ideal for online reinforcement learning because the belief state depends on the model and, as such, is not observable when the model is unknown.**

In this paper, we present an alternative notion of an information state for obtaining a dynamic program in partially observed models. In particular, an information state is a sufficient statistic for the current reward which evolves in a controlled Markov manner. We show that such an information state leads to a dynamic programming decomposition. Then we present a notion of an approximate information state and present an approximate dynamic program based on the approximate information state. Approximate information state is defined in terms of properties that can be estimated using sampled trajectories. Therefore, they provide a constructive method for reinforcement learning in partially observed systems. We present one such construction and show that it performs better than the state of the art for three benchmark models.

I. Introduction

The theory of Markov decision processes focuses primarily on systems with full state observation. When systems with partial state observations are considered, they are converted to systems with full state observations by considering the belief state (which is the posterior belief on the state of the system given the history of observations and actions). Although this leads to an explosion in the size of the state space, the resulting value function has a nice property—it is piecewise linear and convex in the belief state [1]—which is exploited to develop efficient algorithms to compute the optimal policy [2], [3]. Thus, for planning, there is little value in studying alternative characterizations of partially observed models.

However, the belief state formulation is not as nice a fit for online reinforcement learning. Part of the difficulty is that the construction of the belief state depends on the system model. So, when the system model is unknown, the belief state cannot be constructed using the observations. Therefore, critic based methods are not directly applicable. There are some results that circumvent this difficulty [4]–[6]. However, many of the recent results suggest that using RNNs (Recurrent Neural Networks [7]) or LSTMs (Long Short Term Memories [8]) for modeling the policy function (actor) and/or the action-value function (critic) works for reinforcement learning in partially observed systems [9]–[14]. In this paper, we present a rigorous theory for planning and learning in partially observed models using the notions of information state and approximate information state. We then present numerical experiments that show that the approximate information state based works well on benchmark models.

II. Model

A general system with partial observations may be represented using the following stochastic input-output model. Consider a system that takes two inputs: a control input $U_t \in \mathcal{U}$ and a stochastic input $W_t \in W$ and generates two outputs: an observation $Y_t \in \mathcal{Y}$ and a real-valued reward R_t. The spaces $\mathcal{W}$, $\mathcal{U}$, and $\mathcal{Y}$ are Banach spaces and the stochastic inputs $(W_1, \dots, W_T)$ are independent random variables defined on a common probability space.

Remark 1 For ease of exposition, we ignore measurability and present our main arguments informally. We assume that $\mathcal{W}$, $\mathcal{U}$, and $\mathcal{Y}$ are finite sets. The arguments can be made rigorous using standard methods [15]. □

Formally, we assume that there are observation functions $\{f_t\}_{t=1}^T$ and reward functions $\{r_t\}_{t=1}^T$ such that

$$Y_{t+1} = f_t(Y_{1:t}, U_{1:t}, W_t) \quad \text{and} \quad R_t = r_t(Y_{1:t}, U_{1:t}, W_t).$$

An agent observes the history $H_t = (Y_{1:t}, U_{1:t-1})$ of observations and control inputs until time t and chooses the control input

$$U_t = \pi_t(H_t)$$

according to some history dependent policy $\pi := \{\pi_t\}_{t=1}^T$. The performance of policy π is given by

$$J(\pi) = \mathbb{E}^\pi\biggl[\sum_{t=1}^T R_t\biggr]. \tag{1}$$

The objective of the agent is to choose a policy π to maximize the expected total reward $J(\pi)$.

A. A dynamic programming decomposition

In this section, we present a dynamic program for (1) which uses the history of observations and actions as state. Such a dynamic program is not efficient for computing the optimal policy but it will serve as a reference for the rest of the analysis.

First consider the dynamic program for computing the value of any policy π. In particular, define the *reward-to-go* function as

$$J_t(h_t; \pi) := \mathbb{E}^\pi\biggl[\sum_{s=t}^T R_s \biggm| H_t = h_t\biggr]. \tag{2}$$

J. Subramanian and A. Mahajan are with the Faculty of Electrical & Computer Engineering, McGill University, Montreal QC H3A 0E9, Canada jayakumar.subramanian@mail.mcgill.ca aditya.mahajan@mcgill.ca

978-1-7281-1398-2/19/$31.00 ©2019 IEEE

From definitions in (1) and (2), we have

$$J(\pi) = \mathbb{E}[J_1(H_1;\pi)].$$

Thus, the dynamic program (3) gives a recursive method to compute $J(\pi)$.

Let $J_{T+1}(h_{T+1};\pi) \coloneqq 0$. Then, the reward to go functions can be computed recursively as follows:

$$\begin{aligned} J_t(h_t;\pi) &\overset{(a)}{=} \mathbb{E}^\pi\biggl[R_t + \mathbb{E}\biggl[\sum_{s=t+1}^{T} R_s \biggm| H_{t+1}\biggr] \biggm| H_t = h_t\biggr] \\ &= \mathbb{E}^\pi\bigl[R_t + J_{t+1}(H_{t+1};\pi) \bigm| H_t = h_t\bigr], \end{aligned} \tag{3}$$

where (a) follows from the towering property of conditional expectation and the fact that $H_t \subseteq H_{t+1}$. Note that $J_t(h_t;\pi)$ only depends on the future policy $(\pi_t,\dots,\pi_T)$ and not on the past policy $(\pi_1,\dots,\pi_{t-1})$.

Now, recursively define the following *value functions*. $V_{T+1}(h_{T+1}) \coloneqq 0$ and for $t \in \{T,\dots,1\}$:

$$Q_t(h_t,u_t) = \mathbb{E}[R_t + V_{t+1}(H_{t+1}) \mid H_t = h_t, U_t = u_t] \tag{4}$$

and

$$V_t(h_t) = \max_{u_t \in \mathcal{U}} Q_t(h_t,u_t). \tag{5}$$

Theorem 1 *A policy $\pi = (\pi_1,\dots,\pi_T)$ is optimal if and only if it satisfies*

$$\pi_t(h_t) \in \arg\max_{u_t\in\mathcal{U}} Q_t(h_t,u_t). \tag{6}$$

PROOF To prove this, we need to show the following:

(C) At any time t, $J_t(h_t,\pi) \le V_t(h_t)$, with equality if and only if $(\pi_t,\pi_{t+1},\dots,\pi_T)$ satisfy (6).

We prove this using backward induction. At $t = T+1$, (C) is satisfied by definition and this forms the basis of induction. We assume that (C) holds for time $t+1$, which is the induction hypothesis. Then for time t, we have from (3),

$$\begin{aligned} J_t(h_t;\pi) &= \mathbb{E}^\pi\bigl[R_t + J_{t+1}(H_{t+1};\pi) \bigm| H_t = h_t\bigr] \\ &\overset{(a)}{\le} \mathbb{E}^\pi\bigl[R_t + V_{t+1}(H_{t+1}) \bigm| H_t = h_t\bigr] \\ &\overset{(b)}{\le} V_t(h_t), \end{aligned}$$

where (a) follows from the induction hypothesis and (b) follows from the definition of the value function (5) and (4). From the induction hypothesis, the equality in (a) is achieved if and only if $\{\pi_s\}_{s>t}$ satisfy (6). From (5), we see that the equality in (b) is achieved if and only if $\pi_t(h_t) \in \arg\max_{u\in\mathcal{U}} Q_t(h_t,u)$, i.e., π_t satisfies (6). Hence, (C) holds at time t. ■

B. Information state and a simplified dynamic program

Let $\mathcal{F}_t = \sigma(H_t)$ denote the filtration generated by the history of observations and control actions.

Definition 1 An information state $\{Z_t\}_{t\ge1}$, $Z_t \in \mathcal{Z}$, is an $\mathcal{F}_t$ adapted process (therefore, there exist functions $\{\vartheta_t\}_{t=1}^T$ such that $Z_t = \vartheta_t(H_t)$) that satisfies the following properties:

(P1) Sufficient for performance evaluation, i.e.,

$$\mathbb{E}[R_t \mid H_t = h_t, U_t = u_t] = \mathbb{E}[R_t \mid Z_t = \vartheta_t(h_t), U_t = u_t].$$

(P2) Sufficient to predict itself, i.e., for any Borel subset A of $\mathcal{Z}$,

$$\mathbb{P}(Z_{t+1} \in A \mid H_t = h_t, U_t = u_t) = \mathbb{P}(Z_{t+1} \in A \mid Z_t = \vartheta_t(h_t), U_t = u_t).$$

There is no restriction on the space $\mathcal{Z}$, although an information state is useful only when the space $\mathcal{Z}$ is "small" in an appropriate sense. We have assumed that the space $\mathcal{Z}$ is time-homogeneous for convenience. In some situations, it may be more convenient to construct an information state which takes values in spaces that are changing with time.

For some models, instead of (P2), it is easier to verify the following stronger conditions:

(P2a) Evolves in a state-like manner, i.e., there exist measurable functions $\{\varphi_t\}_{t=1}^T$ such that

$$Z_{t+1} = \varphi_t(Z_t, Y_{t+1}, U_t).$$

(P2b) Is sufficient for predicting future observations, i.e., for any Borel measurable subset A of $\mathcal{Y}$,

$$\mathbb{P}(Y_{t+1} \in A \mid H_t = h_t, U_t = u_t) = \mathbb{P}(Y_{t+1} \in A \mid Z_t = \vartheta_t(h_t), U_t = u_t).$$

Proposition 1 *(P2a) and (P2b) imply (P2).* □

PROOF For any Borel measurable subset A of $\mathcal{Z}$, we have

$$\begin{aligned} &\mathbb{P}(Z_{t+1} \in A \mid H_t = h_t, U_t = u_t) \\ &\overset{(a)}{=} \sum_{y_{t+1}\in\mathcal{Y}} \mathbb{P}(Y_{t+1} = y_{t+1}, Z_{t+1} \in A \mid H_t = h_t, U_t = u_t) \\ &\overset{(b)}{=} \sum_{y_{t+1}\in\mathcal{Y}} \mathbb{1}\{\varphi_t(\vartheta_t(h_t), y_{t+1}, u_t) \in A\} \\ &\qquad \times \mathbb{P}(Y_{t+1} = y_{t+1} \mid H_t = h_t, U_t = u_t) \\ &\overset{(c)}{=} \sum_{y_{t+1}\in\mathcal{Y}} \mathbb{1}\{\varphi_t(\vartheta_t(h_t), y_{t+1}, u_t) \in A\} \\ &\qquad \times \mathbb{P}(Y_{t+1} = y_{t+1} \mid Z_t = \vartheta_t(h_t), U_t = u_t) \\ &\overset{(d)}{=} \mathbb{P}(Z_{t+1} \in A \mid Z_t = \vartheta_t(h_t), U_t = u_t) \end{aligned}$$

where (a) follows from the law of total probability, (b) follows from (P2a), (c) follows from (P2b) and (d) from the law of total probability. ■

Note that $Z_t = H_t$ is always an information state, so an information state always exists. It is straight-forward to show that if we construct a state space model for the above input-output model, then the belief on the state given the history of observations and controls is an information state. Below we present an example of a non-trivial information state that is much simpler than the belief state.

Example 1 (Machine Maintenance) Consider a machine which can be in one of n ordered states where the first state is the best and the last state is the worst. The production cost increases with the state of the machine. The state evolves in a Markovian manner. At each time, an agent has the option to either run the machine or stop and inspect it for a cost. After inspection, s/he may either repair it (at a cost that depends on the state) or replace it (at a fixed cost). The objective is to identify a maintenance policy to minimize the cost of production, inspection, repair, and replacement.

Let τ denote the time of last inspection and S_τ denote the state of the machine after inspection, repair, or replacement. Then, it can be shown that $(S_\tau, t-\tau)$ is an information state for the system. □

The main feature of an information state is that one can always write a dynamic program based on an information state.

Theorem 2 *Let $\{Z_t\}_{t=1}^{T}$ be an information state. Recursively define value functions $\{\tilde{V}_t\}_{t=1}^{T+1}$, where $\tilde{V}_t \colon Z_t \mapsto \mathbb{R}$ as follows: $\tilde{V}_{T+1}(z_{T+1}) = 0$ and for $t \in \{T, \dots, 1\}$:*

$$\begin{aligned} \tilde{Q}_t(z_t, u_t) &= \mathbb{E}[R_t + \tilde{V}_{t+1}(Z_{t+1}) \mid Z_t = z_t, U_t = u_t] \\ \tilde{V}_t(z_t) &= \max_{u_t \in \mathcal{U}} \tilde{Q}_t(z_t, u_t). \end{aligned} \tag{7}$$

Then, we have the following:

$$Q_t(h_t, u_t) = \tilde{Q}_t(\vartheta_t(h_t), u_t) \text{ and } V_t(h_t) = \tilde{V}_t(\vartheta_t(h_t)). \tag{8}$$

Proof We prove the result by backward induction. By construction, (8) is true at time $T+1$. This forms the basis of induction. Assume that (8) is true at time $t+1$ and consider the system at time t. Then,

$$\begin{aligned} Q_t(h_t, u_t) &= \mathbb{E}[R_t + V_{t+1}(H_{t+1}) \mid H_t = h_t, U_t = u_t] \\ &\overset{(a)}{=} \mathbb{E}[R_t + \tilde{V}_{t+1}(\vartheta_{t+1}(H_{t+1})) \mid H_t = h_t, U_t = u_t] \\ &\overset{(b)}{=} \mathbb{E}[R_t + \tilde{V}_{t+1}(Z_{t+1}) \mid Z_t = \vartheta_t(h_t), U_t = u_t] \\ &\overset{(c)}{=} \tilde{Q}_t(\vartheta_t(h_t), u_t), \end{aligned}$$

where (a) follows from the induction hypothesis, (b) follows from the properties of information state, and (c) follows from the definition of $\tilde{Q}$. This shows that the action-value functions are equal. By maximizing over the actions, we get that the value functions are also equal. ■

Remark 2 In light of Theorem 2, an information state may be viewed as a generalization of the traditional notion of state [16], [17]. Traditionally, the state of an input-output system is sufficient for input-output mapping. In contrast, the information state is sufficient for dynamic programming.

The notion of information state is also related to sufficient statistics for optimal control [18]. However, in contrast to [18], we do not assume a state space model for the underlying system so it is easier to develop reinforcement learning algorithms using our notion of an information state. □

Coming back to Example 1, Theorem 2 shows that we can write a dynamic program for that model using the information state $(S_\tau, t-\tau)$, which takes values in a countable set. This countable state dynamic program is considerably simpler than the standard belief state dynamic program typically used for that model. Another feature of the information state formulation is that the information state $(S_\tau, t-\tau)$ does not depend on the transition probability of the state of the machine or the cost of inspection or repair. Thus, if these model parameters were unknown, we can use a standard reinforcement learning algorithm to find an optimal policy which maps $(S_\tau, t-\tau)$ to current action.

Given these benefits of a good information state, it is natural to consider a data-driven approach to identify an information state. An information state identified from data will not be exact and it is important to understand what is the loss in performance when using an approximate information state. In the next section, we present a notion of approximate information state and bound the approximation error.

III. Approximate information state (AIS)

Roughly speaking, a compression of the history is an approximate information state if it approximately satisfies (P1) and (P2). This intuition can be made precise as follows.

Definition 2 Given positive numbers ε and δ, an (ε, δ)-approximate information state $\{\hat{Z}_t\}_{t=1}^{T}$, where $\hat{Z}_t$ takes values in a in a Polish metric space $(\hat{\mathcal{Z}}, d)$, is an $\mathcal{F}_t$ adapted process (therefore, there exist functions $\{\widehat{\vartheta}_t\}_{t=1}^{T}$ such that $\hat{Z}_t = \widehat{\vartheta}_t(H_t)$) that satisfies the following properties:

(AP1) Sufficient for approximate performance evaluation, i.e.,

$$\begin{aligned} \big|\mathbb{E}[R_t \mid H_t = h_t, U_t = u_t] - \\ \mathbb{E}[R_t \mid \hat{Z}_t = \widehat{\vartheta}_t(h_t), U_t = u_t]\big| \le \varepsilon. \end{aligned}$$

(AP2) Sufficient to predict itself approximately. For any Borel subset A of $\hat{\mathcal{Z}}$ define,

$$\mu_t(A) = \mathbb{P}(\hat{Z}_{t+1} \in A \mid H_t = h_t, U_t = u_t)$$

and

$$\nu_t(A) = \mathbb{P}(\hat{Z}_{t+1} \in A \mid \hat{Z}_t = \widehat{\vartheta}_t(h_t), U_t = u_t).$$

Then,

$$\mathcal{K}(\mu_t, \nu_t) \le \delta,$$

where $\mathcal{K}(\cdot, \cdot)$ denotes the Wasserstein or Kantorovich-Rubinstein distance[1] between two distributions. □

Remark 3 Kantorovich-Rubinstein duality [19] states that for any probability measures μ and ν on $\mathcal{X}$,

$$\mathcal{K}(\mu, \nu) = \sup_{\|f\|_{\text{Lip}} \le 1} \Big| \int_{\mathcal{X}} f d\mu - \int_{\mathcal{X}} f d\nu \Big|$$

where $\|f\|_{\text{Lip}}$ denotes the Lipschitz constant of a function f (with respect to the metric d). This along with (P2) imply

[1] Let $(\mathcal{X}, d)$ be a Polish metric space. For any two probability measures μ, ν on $\mathcal{X}$, the Wasserstein distance between μ and ν is:

$$\mathcal{K}(\mu, \nu) = \inf_{\pi \in \Pi(\mu, \nu)} \int_{\mathcal{X}} d(x, y) d\pi(x, y)$$

where Π represents the product space of the two distributions.

that for a Lipschitz continuous function $\hat{V}\colon \hat{\mathcal{Z}} \to \mathbb{R}$ with Lipschitz constant L_V (with respect to the metric d),

$$\begin{aligned}\big|\mathbb{E}[\hat{V}(\hat{Z}_{t+1}) | H_t = h_t, U_t = u_t] - {}&\\ \mathbb{E}[\hat{V}(\hat{Z}_{t+1}) | \hat{Z}_t = \widehat{\vartheta}_t(h_t), U_t = u_t]\big| &\le L_V \delta.\end{aligned}$$

Our main result is that one can write an approximate dynamic program based on an approximate information state.

Theorem 3 *Let $\{\hat{Z}_t\}_{t=1}^{T}$ be an (ε, δ)-approximate information state. Recursively define value functions $\{\widehat{V}_t\}_{t=1}^{T+1}$, where $\widehat{V}_t\colon \hat{Z}_t \mapsto \mathbb{R}$ as follows: $\widehat{V}_{T+1}(\hat{z}_{T+1}) = 0$ and for $t \in \{T, \dots, 1\}$:*

$$\begin{aligned}\widehat{Q}_t(\hat{z}_t, u_t) &= \mathbb{E}[R_t + \widehat{V}_{t+1}(\hat{Z}_{t+1}) \mid \widehat{Z}_t = \hat{z}_t, U_t = u_t]\\ \widehat{V}_t(\hat{z}_t) &= \max_{u_t \in \mathcal{U}} \widehat{Q}_t(\hat{z}_t, u_t). \end{aligned} \tag{9}$$

Suppose $\widehat{V}_t$ is Lipschitz continuous with Lipschitz constant L_V. Then, we have the following:

$$\begin{aligned}|Q_t(h_t, u_t) - \widehat{Q}_t(\widehat{\vartheta}_t(h_t), u_t)| &\le (T-t)(\varepsilon + L_V\delta) + \varepsilon\\ |V_t(h_t) = \widehat{V}_t(\widehat{\vartheta}_t(h_t))| &\le (T-t)(\varepsilon + L_V\delta) + \varepsilon.\end{aligned} \tag{10}$$

Proof We prove the result by backward induction. By construction, (10) is true at time $T+1$. This forms the basis of induction. Assume that (10) is true at time $t+1$ and consider the system at time t. Let $C = (T-t-1)(\varepsilon + L_V\delta) + \varepsilon$. Then,

$$\begin{aligned}Q_t(h_t, u_t) &= \mathbb{E}[R_t + V_{t+1}(H_{t+1}) \mid H_t = h_t, U_t = u_t]\\ &\overset{(a)}{\le} \mathbb{E}[R_t + \widehat{V}_{t+1}(\widehat{\vartheta}_{t+1}(H_{t+1})) \mid H_t = h_t, U_t = u_t] + C\\ &\overset{(b)}{\le} \Big(\mathbb{E}[R_t \mid \hat{Z}_t = \widehat{\vartheta}_t(h_t), U_t = u_t] + \varepsilon\Big)\\ &\quad + \Big(\mathbb{E}[\hat{V}_{t+1}(\hat{Z}_{t+1}) \mid \hat{Z}_t = \widehat{\vartheta}_t(h_t), U_t = u_t] + L_V\delta\Big) + C\\ &= \widehat{Q}_t(\widehat{\vartheta}_t(h_t), u_t) + (T-t)(\varepsilon + L_V\delta) + \varepsilon.\end{aligned}$$

where (a) follows from the induction hypothesis and (b) follows from (AP1) and Remark 3. The reverse inequality can be proven using a similar argument. By maximizing over actions, we get the relationship between the value functions. ■

Based on Prop. 1, we provide an alternative characterization of an approximate information state. We can replace (AP2) with the following stronger conditions:

(AP2a) Evolves in a state-like manner, i.e., there exist measurable functions $\{\hat{\varphi}_t\}_{t=1}^{T}$ such that $\hat{Z}_{t+1} = \hat{\varphi}_t(\hat{Z}_t, Y_{t+1}, U_t)$. Moreover, these functions are Lipschitz in Y with Lipschitz constant L_U.

(AP2b) Is sufficient for predicting future observations approximately. For any Borel subset A of $\mathcal{Y}$ define, $\mu_t(A) = \mathbb{P}(Y_{t+1} \in A \mid H_t = h_t, U_t = u_t)$ and $\nu_t(A) = \mathbb{P}(Y_{t+1} \in A \mid \hat{Z}_t = \widehat{\vartheta}_t(h_t), U_t = u_t)$. Then, $\mathcal{K}(\mu_t, \nu_t) \le \delta$,

Proposition 2 *If (AP2) is replaced by (AP2a) and (AP2b), the result of Theorem 3 holds with L_V replaced by $L_U L_V$.* □

Proof If (AP2) is replaced by (AP2a) and (AP2b), the statement of Remark 3 holds with L_V replaced by $L_U L_V$. Using this in Theorem 3 gives the desired result. ■

Corollary 1 *Suppose $\{Z_t\}_{t=1}^{T}$ is an information state and $\{\hat{Z}_t\}_{t=1}^{T}$ is an (ε, δ)-approximate information state. Then for any realization h_t of H_t, we have the following:*

$$\begin{aligned}|Q_t(\vartheta_t(h_t), u_t) - \widehat{Q}_t(\widehat{\vartheta}_t(h_t), u_t)| &\le (T-t)(\varepsilon + L_V\delta) + \varepsilon\\ |V_t(\vartheta_t(h_t)) - \widehat{V}_t(\widehat{\vartheta}_t(h_t))| &\le (T-t)(\varepsilon + L_V\delta) + \varepsilon.\end{aligned} \tag{11}$$

Proof The result follows from Theorems 2 and 3. ■

A. Replacing Wasserstein distance by total variation

It is possible to replace the Wasserstein distance in (AP2) by the total variation distance[2] $\|\mu_t - \nu_t,\|_{TV}$. In particular, suppose we define an (ε, δ)-approximate information state as a process that satisfies (AP1) and

$$\|\mu_t - \nu_t\|_{TV} \le \delta.$$

Then there are two ways to bound the approximation error in this case:

1) In Theorem 3, we replace the condition that $\hat{V}_t$ is Lipschitz with the condition that $\hat{V}_t$ is uniformly bounded and L_V denotes the sup-norm of V_t, i.e., $L_V = \|\hat{V}\|_\infty := \sup_{\hat{z} \in \hat{\mathcal{Z}}} |\hat{V}(\hat{z})|$. This result holds because, $\|\mu_t - \nu_t\|_{TV} < \delta$ implies that for any function $\hat{V}_t$:
$$\Big|\int_{\hat{\mathcal{Z}}} \hat{V}_t d\mu_t - \int_{\hat{\mathcal{Z}}} \hat{V}_t d\nu_t\Big| \le \|\hat{V}_t\|_\infty \delta.$$

2) If $\hat{Z}$ is a bounded metric space with diameter D, then [19, Case 6.16] gives:
$$\mathcal{K}(\mu, \nu) \le D\|\mu - \nu\|_{TV}.$$
Thus, the result of Theorem 3 holds with δ replaced by $D\delta$.

B. Remarks and discussion

- The notion of approximate information state is related to predictive state representation (PSR) [5], which predicts a distribution on the future observations given the current history and future actions. Thus, PSR is a state sufficient for input-output models. However, PSR does not predict future rewards, so it is not sufficient for performance evaluation, and therefore, for dynamic programming.
- The notion of information state is also related to bisimulation based equivalence [20], which constructs an equivalence in the belief state that is sufficient for dynamic programming. In principle, the bisimulation equivalence may be relaxed using bisumulation metrics [21] to obtain an approximate information state.

[2] The total variance metric between two probability measures μ and ν is:
$$\|\mu - \nu\|_{TV} := \sup_{f : \|f\|_\infty \le 1} \Big|\int f(x)\mu(dx) - \int f(x)\nu(dx)\Big|.$$

The key difference in our definition of information state is that we do not assume a state space model. So, an approximate information state is a compression of the history and not just a compression of the beliefs. Therefore, it is easier to develop reinforcement learning algorithms based on approximate information state.
- A reinforcement learning algorithm based on properties very similar to our definition of approximate information state was presented in [14]. However, that paper did not include an approximation result similar to Theorem 3 and, therefore, did not provide any performance guarantees.

C. Relationship to state aggregation

Suppose the approximate information state is a compression of an information state Z_t, rather than the history H_t. In particular, there exist measurable functions $\tilde{\vartheta}_t$ such that $\hat{Z}_t = \tilde{\vartheta}_t(Z_t)$. Such a compression is called a state-based (ε, δ) approximate information state if:

1) $\big|\mathbb{E}[R_t \mid Z_t = z_t, U_t = u_t] - \mathbb{E}[R_t \mid \hat{Z}_t = \tilde{\vartheta}_t(z_t), U_t = u_t]\big| \le \varepsilon.$
2) Let $\mu_t(A) = \mathbb{P}(\hat{Z}_{t+1} \in A \mid Z_t = z_t, U_t = u_t)$ and $\nu_t(A) = \mathbb{P}(\hat{Z}_{t+1} \in A \mid \hat{Z}_t = \tilde{\vartheta}_t(z_t), U_t = u_t)$. Then $\mathcal{K}(\mu_t, \nu_t) \le \delta$.

Then, similar to Theorem 3, we can show that:

$$|Q_t(z_t, u_t) - \widehat{Q}_t(\tilde{\vartheta}_t(z_t), u_t)| \le (T - t)(\varepsilon + L_V \delta) + \varepsilon$$
$$|V_t(z_t) = \widehat{V}_t(\tilde{\vartheta}_t(z_t))| \le (T - t)(\varepsilon + L_V \delta) + \varepsilon.$$

These bounds are similar to bounds for aggregating Markov decision processes obtained in [22].

IV. Extension to infinite horizon

In this section, we explain how to extend the notions of information state and approximate information state to infinite horizon discounted reward setup where the performance of a policy is given by

$$J_\infty(\pi) = \mathbb{E}^\pi\Big[\sum_{t=1}^{\infty} \beta^{t-1} R_t\Big],$$

where $\beta \in (0, 1)$ is the discount factor. Such an extension is non-trivial because we do not assume a state space model for the system. So in the infinite horizon case, the history dependent dynamic program (3) cannot be written as the fixed point of a time-homogeneous contractive operator. Nonetheless, we show that when the per-step reward is uniformly bounded, the obvious extensions of the information state to infinite horizon works.

A. Information state for infinite horizon

Definition 3 An $\mathcal{F}_t$-adapted process $\{Z_t\}_{t \ge 1}$ is an information state for infinite horizon if, in addition to satisfying (P1) and (P2), it satisfies the following:

(S) The expectation $\mathbb{E}[R_t | Z_t = \vartheta_t(H_t), U_t = u_t]$ and the transition kernel $\mathbb{P}(Z_{t+1} \in A | Z_t = \vartheta_t(H_t), U_t = u_t)$ are time-homogeneous.

We refer to such a process as time-homogeneous information state. □

In time-homogeneous infinite horizon POMDPs, the belief state is an information state because it satisfies (P1) and (P2) and also satisfies (S).

For any time-homogeneous information state, define the Bellman operator $\mathcal{B}\colon [\mathcal{Z} \to \mathbb{R}] \to [\mathcal{Z} \to \mathbb{R}]$ as follows: for any uniformly bounded function $V\colon \mathcal{Z} \to \mathbb{R}$

$$[\mathcal{B}V](z) = \max_{u \in \mathcal{U}} \mathbb{E}[R_t + \beta V(Z_{t+1}) | Z_t = z, U_t = u]. \quad (12)$$

Because of (S), the expectation on the right hand side does not depend on time. Due to discounting, the operator $\mathcal{B}$ is a contraction and therefore, if the rewards are uniformly bounded, the following fixed point equation has a unique bounded solution:

$$V = \mathcal{B}V. \quad (13)$$

Let V^* be the fixed point and π^* be any policy such that $\pi^*(z)$ achieves the arg max in the right hand side of (12) for $[\mathcal{B}V^*](z)$. Is is easy to see that V^* is the performance of the time homogeneous policy $(\pi^*, \pi^*, \dots)$. However, it is not obvious that V^* equals to the optimal performance J^{OPT} (defined below), because the proof of Theorem 2 relies on backward induction and is not applicable to infinite horizon models. So, we present an alternative proof below.

Theorem 4 *Let $\{Z_t\}_{t \ge 1}$ be a time-homogeneous information state process. Suppose the rewards are uniformly bounded and lie in the interval $[0, M]$. Let V^* be the unique bounded fixed point of the Bellman operator $\mathcal{B}$. Fix a starting time s and let J^{OPT}_s denote the optimal performance from time s onwards, i.e.,*

$$J^{\text{OPT}}_s(h_s) := \max_\pi \mathbb{E}^\pi\Big[\sum_{t=s}^{\infty} \beta^{t-s-1} R_t \;\Big|\; H_s = h_s\Big], \quad (14)$$

where the maximum is over all (possibly randomized) history dependent policies. Then, $J^{\text{OPT}}_s(h_s) = V^*(\vartheta_s(h_s))$. □

Proof Fix a time $T > s$ and let

$$J^{\text{OPT}}_{s,T}(h_s) := \max_\pi \mathbb{E}^\pi\Big[\sum_{t=s}^{T} \beta^{t-s-1} R_t \;\Big|\; H_s = h_s\Big]$$

be the optimal performance for the time interval $[s, T]$. Note that $J^{\text{OPT}}_{s,\infty} = J^{\text{OPT}}_s$.

Let $V^{(0)} = 0$ and iteratively define $V^{(n+1)} = \mathcal{B}V^{(n)}$. From Theorem 1, we know that $J^{\text{OPT}}_{s,T}(h_s) = V^{(T-s)}(\vartheta_s(h_s))$. Now, we consider two directions:

- We first derive a lower bound on $J^{\text{OPT}}_{s,\infty}$. Note that

$$\begin{aligned} J^{\text{OPT}}_{s,\infty}(h_s) &= \max_\pi \mathbb{E}\Big[\sum_{t=s}^{\infty} \beta^{t-s-1} R_t \;\Big|\; H_s = h_s\Big] \\ &\ge \max_\pi \mathbb{E}\Big[\sum_{t=s}^{T} \beta^{t-s-1} R_t \;\Big|\; H_s = h_s\Big] \\ &= J^{\text{OPT}}_{s,T}(h_s) = V^{(T-s)}(\vartheta_s(h_s)). \end{aligned} \quad (15)$$

- Next, we derive an upper bound on $J^{\text{OPT}}_{s,\infty}$. Note that

$$\begin{aligned}
J^{\text{OPT}}_{s,\infty}(h_s) &= \max_{\pi} \mathbb{E}\Bigg[\sum_{t=s}^{\infty} \beta^{t-s-1} R_t \Bigg| H_s = h_s\Bigg] \\
&\le \max_{\pi} \mathbb{E}\Bigg[\sum_{t=s}^{T} \beta^{t-s-1} R_t \Bigg| H_s = h_s\Bigg] + \sum_{t=T+1}^{\infty} \beta^{t-s-1} M \\
&= J^{\text{OPT}}_{s,T}(h_s) + \frac{\beta^T}{1-\beta} M \\
&= V^{(T-s)}(\vartheta_s(h_s)) + \frac{\beta^T}{1-\beta} M. \qquad (16)
\end{aligned}$$

Combining (15) and (16), we get

$$V^{(T-s)}(\vartheta_s(h_s)) \le J^{\text{OPT}}_{s,T}(h_s) \le V^{(T-s)}(\vartheta_s(h_s)) + \frac{\beta^T}{1-\beta} M. \qquad (17)$$

Recall that $\mathcal{B}$ is a contraction. Therefore, $\lim_{T\to\infty} V^{(T-s)} = V^*$. Hence, the result follows from (17) by taking the limit $T \to \infty$. ■

B. Approximate information state for infinite horizon

Definition 4 An $\mathcal{F}_t$-adapted process $\{\hat{Z}_t\}_{t\ge 1}$ is a (ε, δ)-approximate information state for infinite horizon if, in addition to satisfying (AP1) and (AP2), it satisfies the following:

(AS) The expectation $\mathbb{E}[R_t | \hat{Z}_t = \widehat{\vartheta}_t(H_t), U_t = u_t]$ and the transition kernel $\mathbb{P}(\hat{Z}_{t+1} \in A | \hat{Z}_t = \widehat{\vartheta}_t(H_t), U_t = u_t)$ are time-homogeneous.

We refer to such a process as time-homogeneous approximate information state. □

If $\widehat{\mathcal{Z}}$ is a compact subset of the Euclidean space with diameter $d_{\max}$ and the per step reward is bounded by $R_{\max}$, then any $\widehat{\mathcal{Z}}$ valued process satisfies $\varepsilon \le R_{\max}$ and $\delta \le d_{\max}$. Thus, any compression is a $(R_{\max}, d_{\max})$ approximate information state.

As before, define the Bellman operator $\widehat{\mathcal{B}}\colon [\widehat{\mathcal{Z}} \to \mathbb{R}] \to [\widehat{\mathcal{Z}} \to \mathbb{R}]$ as follows: for any uniformly bounded function $V\colon \widehat{\mathcal{Z}} \to \mathbb{R}$,

$$[\widehat{\mathcal{B}}V](\hat{z}) = \max_{u\in\mathcal{U}} \mathbb{E}[R_t + \beta V(\hat{Z}_{t+1}) | \hat{Z}_t = \hat{z}, U_t = u]. \qquad (18)$$

Because of (AS), the expectation on the right hand side does not depend on time. Then, similar to Theorem 4, we can establish the following.

Theorem 5 *Let $\{\hat{Z}_t\}_{t=1}^{\infty}$ be a time-homogeneous (ε, δ)-approximate information state. Suppose the rewards are uniformly bounded. Let $\hat{V}^*$ be the unique bounded fixed point of $V = \widehat{\mathcal{B}}V$. Suppose $\hat{V}^*$ is Lipschitz continuous with Lipschitz constant L_V. Then,*

$$|J^{\text{OPT}}_s(h_s) - V^*(\widehat{\vartheta}_t(h_s))| \le \frac{\varepsilon + \beta L_V \delta}{1-\beta}.$$

PROOF The proof follows by combining ideas from Theorems 3 and 4. ■

V. REINFORCEMENT LEARNING USING APPROXIMATE INFORMATION STATE

In this section, we use an approximate information state to design reinforcement learning algorithms for infinite horizon POMDPs. We split our approach into two steps—a data-driven approach to construct an approximate information state and reinforcement learning using this approximate information state.

A. Constructing an approximate information state

The definition of approximate information state suggests two ways to construct an information state from data: either use $\hat{\vartheta}(h_t)$ to determine an approximate information state that satisfies conditions (AP1) and (AP2) or conditions (AP1), (AP2a), and (AP2b). The first approach is more efficient, but the second is easier to understand. So we first describe the latter and then the former.

Note that training a network requires the control inputs $\{U_t\}_{t\ge 1}$. In this section, we assume that the control and the observations have been generated according to a pure exploration policy. In the next section, we will consider the case when policy is being learned along with the approximate information state.

1) Construction based on (AP1), (AP2a) and (AP2b): We use two function approximators:

- A recurrent neural network (RNN) or its refinements such as LSTM (Long Short-Term Memory) [8] or GRU (Gated Recurrent Unit) [23] with state $C_{t-1} = \hat{Z}_{t-1}$, inputs (Y_t, U_{t-1}) and output $\hat{Z}_t$. We denote this function approximator by ρ.
- A feed forward network with inputs $(\hat{Z}_t, U_t)$ and output $(\tilde{R}_t, \tilde{\nu}_{t+1})$, where $\tilde{R}_t$ is a prediction of the expected reward and $\tilde{\nu}_{t+1}$ is the prediction of ν_{t+1}, the distribution of the next observation Y_{t+1}. We parameterize $\tilde{\nu}_{t+1}$ as multi-variate Gaussain. We denote this function approximator as ψ.

By construction ρ satisfies (AP2a). To minimize the ε in (AP1), we define the loss functions

$$\mathcal{L}_R = \frac{1}{B} \sum_{t=1}^{B} \texttt{smoothL1}(\tilde{R}_t - R_t),$$

where B is the batch size and

$$\texttt{smoothL1}(x) = \begin{cases} \frac{1}{2}x^2 & \text{if } |x| < 1 \\ |x| - \frac{1}{2} & \text{otherwise,} \end{cases}$$

is the standard smooth approximation for L1 loss. To minimize the δ in (AP2), we define the loss function

$$\mathcal{L}_\nu = -\sum_{t=1}^{B-1} \log(\tilde{\nu}_{t+1}(Y_{t+1})),$$

which is the negative log likelihood loss for $\tilde{\nu}_t$ and thus approximates the KL-divergence between μ_t and ν_t. We use the KL-divergence as a surrogate for the Wasserstein distance because: (i) Wasserstein distance is computationally expensive to compute; and (ii) KL-divergence upper bounds

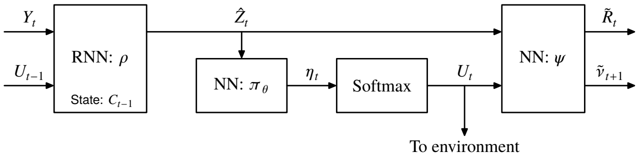


Fig. 1: Neural network based function approximators for RL using AIS.

the total variation (due to Pinsker's inequality), which in turn upper bounds Wasserstein distance for metric spaces with bounded diameter. To train the networks ρ and ψ, we use a weighted combination of these losses to get a single scalar loss:

$$\mathcal{L}_{\rho,\psi} = \lambda \mathcal{L}_R + (1-\lambda)\mathcal{L}_\nu \tag{19}$$

where $\lambda \in [0,1]$ is a hyperparameter.

2) Construction based on (AP1) and (AP2): We use two function approximators:

- A recurrent neural network (RNN) or its refinements such as LSTM (Long Short-Term Memory) [8] or GRU (Gated Recurrent Unit) [23] with state C_{t-1}, inputs (Y_t, U_{t-1}) and output $\hat{Z}_t$. We denote this function approximator by ρ.
- A feed forward network as in the previous case, except that $\tilde{\nu}_{t+1}$ is the prediction of ν_{t+1}, the distribution of the next approximate information state $\hat{Z}_{t+1}$. We denote this function approximator as ψ.

Note that we do not require $C_{t-1} = \hat{Z}_{t-1}$ in this case. This is because to satisfy (AP2), the approximate information state just needs to be a function of the history, and not necessarily evolve in a state-like manner. To minimize ε and δ, we train the networks ρ and ψ using the loss function $\mathcal{L}_{\rho,\psi}$ defined in (19), where $\mathcal{L}_R$ is as before and

$$\mathcal{L}_\nu = -\sum_{t=1}^{B-1} \log(\tilde{\nu}_{t+1}(\hat{Z}_{t+1})). \tag{20}$$

B. Reinforcement learning

In this section, we present an approach to use the approximate information state for reinforcement learning. Let $\pi_\theta : \hat{Z}_t \mapsto \Delta(U_t)$ be a parametrized stochastic policy, where the parameters θ lie in a closed convex set Θ. For example, π_θ could be a feed forward neural network with input $\hat{Z}_t$ and output to be a $|U_t|$ dimensional vector η, which forms the input to a softmax function, that can be written as:

$$\pi_\theta(u|\hat{z}) = \frac{\exp(\tau\eta_u)}{\sum_{w\in\mathcal{U}}\exp(\tau\eta_w)}, \tag{21}$$

where τ is a hyperparameter. In such a policy, θ corresponds to the weights of the network. The basic idea behind policy based reinforcement learning is to get sample path based estimates of the performance gradient $\nabla_\theta J$, which is then used as a gradient loss function for updating the parameters θ using stochastic gradient descent.

An architecture for combining the construction of the approximate information state with reinforcement learning is shown in Fig. 1. In this architecture, we train the networks (ρ, ϕ) and π_θ in parallel using a two time-scale algorithm. In particular, by a slight abuse of notation, let ρ and ψ denote the weights of the corresponding networks. Then,

$$\begin{bmatrix}\rho_{k+1}\\ \psi_{k+1}\end{bmatrix} = \begin{bmatrix}\rho_k\\ \psi_k\end{bmatrix} + a_k \nabla_{\rho,\psi}\mathcal{L}_{\rho,\psi} \text{ and } \theta_{k+1} = \theta_k + b_k \nabla_\theta J(\pi_{\theta_k}),$$

where the learning rates $\{a_k\}_{k\geq 1}$ and $\{b_k\}_{k\geq 1}$ satisfy the standard two time-scale stochastic approximation conditions [24].

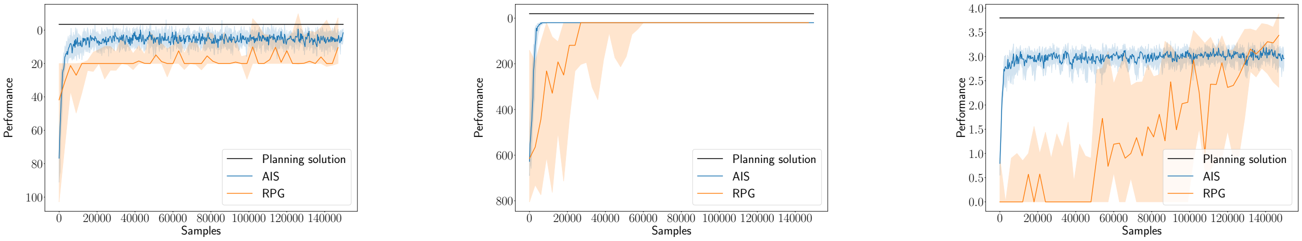
VI. Numerical experiments

In this section, we use the approximate information state based reinforcement learning for three small dimensional POMDP benchmarks: voicemail [25], tiger [2] and 4×4 grid [26]. See [27] for the details of the environments.

We use the approach described in Sec. V-A.2, with the following choices for the networks:

- The ρ network is a two layer recurrent neural network, where the input is one-hot encoded, the first layer is a fully connected layer with 5 neurons and tanh activation and the second layer is a GRU layer with 5 neurons. This network outputs $\hat{Z}_t$ as the state of the GRU cell.
- The ψ network has two parts—one for $\tilde{R}_t$ and one for $\tilde{\nu}_{t+1}$. Both these parts are two layer feedforward neural networks, where the input is an approximate information state and one-hot encoded action, the first layer is a fully connected layer with 5 neurons and tanh activation and the second layer is a fully connected layer with a single neuron for $\tilde{R}_t$ and 5 neurons for $\tilde{\nu}_{t+1}$. This network outputs $\tilde{R}_t$ and the mean for a 5 dimensional multi-variate Gaussian distribution with unit variance as a parameterized distribution for $\tilde{\nu}_{t+1}$.
- The policy network π_θ is a two layer feedforward neural network, where the input is an approximate information state, the first layer is a fully connected layer with 5 neurons and tanh activation and the second layer is a fully connected layer with $|\mathcal{U}|$ neurons. The output of this network are the parameters of a $|\mathcal{U}|$ dimensional softmax distribution.

We train these networks for $\beta = 0.95$, $\lambda = 10/11$, $B = 300$ and $\{a_k\}_{k\geq 1}$ and $\{b_k\}_{k\geq 1}$ chosen according to ADAM(0.1) and ADAM(0.08) [28] respectively. The performance gradients are estimated using REINFORCE [29]. The plots for 500 iterations of the algorithm are shown in Fig. 2. We compare our performance with recurrent policy gradient (RPG) [11] algorithm, which is one of the state of the art algorithms for POMDPs. The data for RPG is taken from [27]. The experiments with RPG used a two layer RNN for the policy function, where the first layer is a recurrent LSTM layer with 20 neurons and the second layer is a fully connected layer with 20 neurons. This outputs parameters for the softmax function to obtain distributions over actions. The RPG implementation also included a history dependent baseline for variance reduction. This was a two layer RNN where the first layer is a recurrent LSTM layer with 20



(a) Voicemail problem

(b) Tiger problem

(c) 4×4 grid problem

Fig. 2: Performance versus samples for all examples. The solid line shows the median value and the shaded region shows the region between the first and third quartiles over 25 runs.

neurons and the second layer is a fully connected layer with a single neuron. In all three examples, our algorithm performed better than or as good as RPG. Part of the reason for better performance is that we could use a much higher learning rate of 0.08 for the policy network π_θ, as compared to the RPG implementation in [27], which experienced instability for learning rates above 0.001.

VII. Conclusion

In this paper, we present a notion of information state for partially observed systems. We show that an information state is sufficient for dynamic programming. We then relax the definition to describe an approximate information state that can be used to identify an approximately optimal policy.

The approximate information state is defined in terms of properties that can be estimated from data, so it can be used to develop sampling based reinforcement learning algorithms. We present one such algorithm and show that it performs better than or comparable to RPG, which is a state of the art reinforcement learning algorithm for POMDPs.

The actor only reinforcement learning algorithm presented in this paper is just a proof of concept. It is straight forward to extend standard critic only and actor-critic algorithms using approximate information state by adding a neural network that approximates action-value functions $\hat{Q}^*(\hat{z}, u)$ and $\hat{Q}^\pi(\hat{z}, a)$ in the architecture in Fig. 1. It will be interesting to evaluate how such extensions perform in practice.

Acknowledgments

We are grateful to Raihan Seraj for providing planning and RPG solutions for all three examples in Fig. 2.

References

[1] R. D. Smallwood and E. J. Sondik, “The optimal control of partially observable markov processes over a finite horizon,” *Operations research*, vol. 21, no. 5, pp. 1071–1088, 1973.

[2] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, “Planning and acting in partially observable stochastic domains,” *Artificial intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[3] G. Shani, J. Pineau, and R. Kaplow, “A survey of point-based POMDP solvers,” *AAMAS*, 2013.

[4] J. Baxter and P. L. Bartlett, “Infinite-horizon policy-gradient estimation,” *JAIR*, vol. 15, pp. 319–350, 2001.

[5] M. L. Littman, R. S. Sutton, and S. P. Singh, “Predictive representations of state,” in *NIPS*, 2002.

[6] A. Hefny, Z. Marinho, W. Sun, S. Srinivasa, and G. Gordon, “Recurrent predictive state policy networks,” *arXiv:1803.01489*, 2018.

[7] D. E. Rumelhart, G. E. Hinton, R. J. Williams *et al.*, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 9, pp. 533–536, 1986.

[8] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[9] B. Bakker, “Reinforcement learning with long short-term memory,” in *NIPS*, 2002.

[10] D. Wierstra, A. Foerster, J. Peters, and J. Schmidhuber, “Solving deep memory POMDPs with recurrent policy gradients,” in *International Conference on Artificial Neural Networks*, 2007.

[11] D. Wierstra, A. Förster, J. Peters, and J. Schmidhuber, “Recurrent policy gradients,” *Logic Journal of the IGPL*, vol. 18, no. 5, pp. 620–634, 2010.

[12] M. Hausknecht and P. Stone, “Deep recurrent Q-learning for partially observable MDPs,” in *2015 AAAI Fall Symposium Series*, 2015.

[13] N. Heess, J. J. Hunt, T. P. Lillicrap, and D. Silver, “Memory-based control with recurrent neural networks,” *arXiv:1512.04455*, 2015.

[14] A. Baisero and C. Amato, “Learning internal state models in partially observable environments;,” *Reinforcement Learning under Partial Observability, NeurIPS Workshop*, 2018.

[15] O. Hernández-Lerma and J. B. Lasserre, *Discrete-time Markov control processes: basic optimality criteria*. Springer, 2012.

[16] A. Nerode, “Linear automaton transformations,” *Proceedings of American Mathematical Society*, vol. 9, pp. 541–544, 1958.

[17] H. S. Witsenhausen, “Some remarks on the concept of state,” in *Directions in Large-Scale Systems*, Y. C. Ho and S. K. Mitter, Eds. Plenum, 1976, pp. 69–75.

[18] C. Striebel, “Sufficient statistics in the optimal control of stochastic systems,” *Journal of Mathematical Analysis and Applications*, vol. 12, pp. 576–592, 1965.

[19] C. Villani, *Optimal transport: Old and New*. Springer, 2008.

[20] P. S. Castro, P. Panangaden, and D. Precup, “Equivalence relations in fully and partially observable markov decision processes,” in *IJCAI*, 2009.

[21] N. Ferns, P. Panangaden, and D. Precup, “Metrics for finite markov decision processes,” in *UAI*, 2004.

[22] D. Bertsekas, “Convergence of discretization procedures in dynamic programming,” *IEEE Trans. Autom. Control*, vol. 20, no. 3, pp. 415–419, 1975.

[23] K. Cho, B. v. M. C. Gulcehre, D. Bahdanau, F. B. H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” in *EMNLP*, 2014.

[24] V. S. Borkar, “Stochastic approximation with two time scales,” *Systems & Control Letters*, vol. 29, no. 5, pp. 291–294, 1997.

[25] J. D. Williams and S. Young, “Partially observable markov decision processes for spoken dialog systems,” *Computer Speech & Language*, vol. 21, no. 2, pp. 393–422, 2007.

[26] A. R. Cassandra, L. P. Kaelbling, and M. L. Littman, “Acting optimally in partially observable stochastic domains,” in *AAAI*, 1994.

[27] R. Seraj, “Learning in the presence of partial observability and concept drifts,” Master’s thesis, McGill University, 2019.

[28] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv:1412.6980*, 2014.

[29] R. J. Williams, “Simple statistical gradient-following algorithms for connectionist reinforcement learning,” *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.