Learning the Difference between Partially Observable Dynamical Systems

Sami Zhioua[1], Doina Precup[1], François Laviolette[2], and Josée Desharnais[2]

[1] School of Computer Science, McGill University, QC, Canada
[2] Department of Computer Science and Software Engineering, Laval University, QC, Canada

Abstract. We propose a new approach for estimating the difference between two partially observable dynamical systems. We assume that one can interact with the systems by performing actions and receiving observations. The key idea is to define a Markov Decision Process (MDP) based on the systems to be compared, in such a way that the optimal value of the MDP initial state can be interpreted as a divergence (or dissimilarity) between the systems. This dissimilarity can then be estimated by reinforcement learning methods. Moreover, the optimal policy will contain information about the actions which most distinguish the systems. Empirical results show that this approach is useful in detecting both big and small differences, as well as in comparing systems with different internal structure.

1 Introduction

When building or learning models of dynamical systems, one is often confronted with the problem of choosing between two models, or assessing whether a given model is faithful to the original system. In such scenarios, a notion of behavioral distance (or divergence) between dynamical systems can be very helpful. For example, when learning a model of a Partially Observable Markov Decision Process (POMDP), one might make assumptions about the number of internal states, then learn the parameters of the system from data. It would be very useful for the learning process to have a notion of how close the learned model is to the true system. Moreover, an algorithm that can point out what parts of the learned model are most different from the original would be very useful, as it would suggest where the learned model can be improved. The results of such an algorithm can be used to guide both the process of selecting the structure of a new model, as well as the data acquisition. An even greater need for determining differences arises if the systems under consideration are different in structure (e.g., a first and a second-order Markov model, or a POMDP and a predictive state representation). In this case, simple inspection of the models cannot determine if they behave similarly. Instead, we need a way of assessing dissimilarity based purely on data, independently of the model structure.

In this paper, we develop a novel approach to this problem, inspired by ideas from probabilistic verification [1]. The key idea is to define a Markov Decision Process (MDP) whose states and actions are built based on the processes that we want to compare. The optimal value function of this MDP will be interpreted as a divergence (or distance) between the processes. No knowledge of the original processes is needed; instead, we only need the ability to interact with them. The optimal policy in the MDP

W. Buntine et al. (Eds.): ECML PKDD 2009, Part II, LNAI 5782, pp. 664–677, 2009.

will point out the actions which distinguish the systems most. This is very useful both in learning applications, as well as in knowledge transfer, as we discuss in detail later.

The paper is structured as follows. Section 2 presents the problem and notation. Section 3 presents the core ideas of our approach. Section 4 establishes the theoretical properties of the proposed approach. We show that the MDP we define has a positive value function, with strict inequality if and only if the two dynamical systems under consideration are different. We also discuss the sample complexity of our approach. In Section 5, we illustrate the behavior of the algorithm on several standard POMDP examples from the literature. We show that the algorithm can both identify when the systems are different, as well as quantify the extent of the difference. Section 6 contains a discussion and related work. Finally, in Section 7 we conclude and present avenues for future work.

2 Background

An MDP is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is a set of actions, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \longrightarrow [0, 1]$ is a transition function such that $\mathcal{T}(s, a, s')$ represents the probability of making a transition from s to s' after action a, and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \longrightarrow \mathbb{R}$ is a reward function such that $\mathcal{R}(s, a)$ is the reward obtained when action a is performed in state s.

We will use MDPs to establish the differences between two stochastic, partially observable systems. We make the assumption that it is possible to interact with the systems by choosing actions a_t at discrete time steps and receiving in response observations o_t. We also assume that there is a well-defined (but possibly unknown) probability distribution $Pr(o_{t+1}|a_1 o_1 \ldots a_t o_t), \forall t$. The most common model for such systems in the AI literature is the Partially Observable Markov Decision Process (POMDP) (see [2] for a survey). However, this formulation is more general and it includes, for example, higher-order Markov systems, as well as systems with internal structure represented as a finite state machine.

The notion of equivalence or similarity between probabilistic processes has been studied extensively in the literature on verification of concurrent processes. In their pioneering work [3], Larsen and Skou have defined a notion of *tests*, which consist conceptually of logical formulas defined based on a given grammar. The formulas are defined over the observations produced by the systems, and give a characterization of what statements can made true in the two processes. Two processes are considered equivalent if they accept all tests with equal probabilities. Different grammars for generating tests give rise to different notions of equivalence between processes. The notion of equivalence can be relaxed by using *metrics*, which represent quantitative measures of similarity between processes. Metrics are often defined based on the complexity of the grammar needed to distinguish the processes.

In the AI literature, a similar idea is captured by predictive state representations (PSRs) [4], which characterize the "state" of a dynamical system by the set of conditional probabilities of different observation sequences that can be observed the current history, given different sequences of actions. In some cases, PSRs can provide a compact representation of the dynamical system. However, from the point of view of this

paper, the main idea we want to emphasize is the use of tests, and their associated probabilities.

A *test* is a sequence of actions of the form $a_1 a_2 \ldots a_n$ where $a_1, a_2, \ldots, a_n \in \mathcal{A}$. Running a test on a dynamical system produces a sequence of observations of the form $o_1 o_2 \ldots o_n$ where $o_1, o_2, \ldots, o_n \in \mathcal{O}$. The set of all such sequences will be denoted $\mathcal{O}^*$. Under our definition of dynamical system, any test induces a well-defined probability distribution over observations sequences:

$$P(o_1 \ldots o_n | a_1 \ldots a_n) = P(o_1|a_1)P(o_2|a_1, o_1, a_2) \ldots P(o_n|a_1, o_1, \ldots a_{n-1} o_{n-1} a_n).$$

Definition 1. Two dynamical systems with probability distributions P_1 and P_2 are *trace equivalent* if and only if, for any test $a_1 \ldots a_n$, and for any sequence of observations $o_1 \ldots o_n$,

$$P_1(o_1 \ldots o_n | a_1 \ldots a_n) = P_2(o_1 \ldots o_n | a_1 \ldots a_n).$$

3 Quantifying the Difference between Dynamical Systems

Let P_1 and P_2 be two processes that we are interested in comparing (for ease of notation, we will denote a system by its associated probability distributions). We are interested in determining tests which maximize the difference between the two systems. Our strategy will be two-fold. First, we will break down the action sequence of a test into steps, and consider one action at a time. Second, we will establish a reward function which emphasizes the differences between processes: if the same action causes *different* observations in the two processes, it should receive a high reward, otherwise, it should receive a low reward. However, because the systems under consideration are stochastic, different observations may be obtained in the two systems even if they are identical. The likelihood of this occurrence is higher if the probability distributions associated with the systems have high entropy. In order to correctly account for this possibility, we introduce a third and a fourth processes, called respectively P_{1c} and P_{2c}, which are simply clones of the original systems P_1 and P_2 (see Fig.1). There will be a high reward if P_1 and P_2 differ for some action, but this reward can be cancelled if P_1 and P_{1c} differ and/or P_2 and P_{2c} differ. The intuition is that in this case, the difference we observed may well be due to the inherent stochasticity in the processes and not to an actual behavioral difference between them.

The intuition for our approach is inspired by the well-known Kullback-Leibler (KL) divergence [5]. A divergence between two dynamical systems could be defined as the maximum divergence between the probability distributions over observation sequences generated by all tests run in the systems. The KL divergence is a good way of measuring the similarity of two probability distributions. For two distributions P_1 and P_2, it is defined as:

$$\mathrm{KL}(P_1 \| P_2) := \mathrm{E}_{h \sim P_1} \ln \frac{1}{P_2(h)} - \mathrm{E}_{h \sim P_1} \ln \frac{1}{P_1(h)} \quad (1)$$

Unfortunately, because of the high number of possible tests (especially large systems), computing the maximum value over all Kullback-Leibler divergences is not tractable. Nevertheless, there is an analogy between our intuition and the KL divergence. The first

term in the formula can be considered as comparing the two processes of interest, P_2 and P_1. The second term accounts for the entropy of the first process, P_1, for a given test. However, the KL divergence is not symmetrical, whereas for our purpose, the two processes should be treated in the same way. Hence, we would like a divergence which is closer in spirit to the "symmetrized" KL divergence, $KL(P_1|P_2) + KL(P_2|P_1)$. Also, instead of striving to compute the KL divergence, we will compute a different divergence, which will be more efficient. We are now ready to detail our divergence computation.

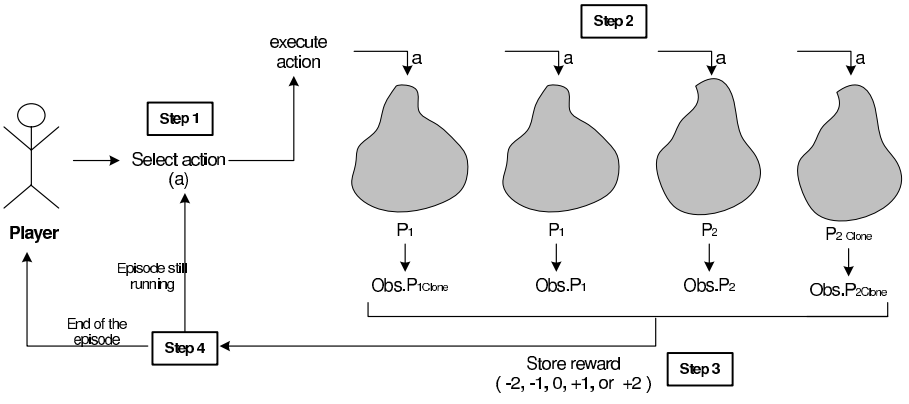

Fig. 1. Sketch of the divergence computation

Figure 1 depicts the setup that we will use. The interaction with the processes will consist of the following steps:

Initialize the four processes (P_{1c}, P_1, P_2, and P_{2c})
Repeat:

Step 1 : Choose an action a.
Step 2 : Execute a on P_{1c}, P_1, P_2. and P_{2c}.
Step 3 : Compute the reward by adding the following components:
- a reward of $(+2)$ for different observations between P_1 and P_2
- a (-1) reward for different observations between P_1 and P_{1c}
- a (-1) reward for different observations between P_2 and P_{2c}.

In other words, if $Obs.P_i$ denotes the observation obtained in system i, the immediate (stochastic) reward is given by the formula:

$$R := 2\ (Obs.P_1 \neq Obs.P_2) - (Obs.P_1 \neq Obs.P_{1c}) - (Obs.P_2 \neq Obs.P_{2c}) \quad (2)$$

where 0 and 1 are used as both truth values and numbers.

until either $Obs.P_1 \neq Obs.P_2$, or the episode reaches a certain horizon.

We will now show that this interaction gives rise to an MDP, whose value is 0 if and only if the two processes have the same probability distribution over observation sequences for all tests.

4 MDP Formulation and Theoretical Properties

Definition 2. Given two dynamical systems P_1 and P_2 with the same sets of actions and observations, $\mathcal{A}$ and $\mathcal{O}$, we define an MDP $\mathcal{M}$ as follows:

- The state space is the set $S := \{(\mathcal{AO})^*\} \cup \{Dead\}$, with an initial state $s_0 = \epsilon$ (corresponding to the empty action-observation sequence).
- The set of actions is $\mathcal{A}$.
- The transition function is defined as:

$$\mathcal{T}(s,a,s') := \begin{cases} P_1(o|s,a)P_2(o|s,a) & \text{if } s' = sao \\ 1 - \sum_{o \in \mathcal{O}} P_1(o|s,a)P_2(o|s,a) & \text{if } s' = Dead \\ 0 & \text{otherwise} \end{cases}$$

- The reward function is defined as in Equation (2)

The MDP states correspond to action-observation sequences $a_1 o_1 a_2 o_2 \ldots$. When the observations obtained in P_1 and P_2 do not coincide, the episode stops. In the MDP, this corresponds to a transition to the $Dead$ state. Note that for a state $s = a_1 o_1 \ldots a_n o_n \in S$, the notation $P_1(o|s,a) = P_1(o|a_1 \ldots a_n, o_1 \ldots o_n, a)$ represents the probability of observing o after executing action a, following the history $a_1 o_1 \ldots a_n o_n$.

We will now show that this MDP has optimal value function equal to 0 if and only if the two systems are identical. Moreover, the value function is never negative; therefore, it defines a divergence notion. First, we will show the following lemma:

Lemma 1. *The expected value of the reward in the MDP given by Def. 2 is:*

$$\mathcal{R}(s,a) = \sum_{o \in \mathcal{O}} (P_1(o|s,a) - P_2(o|s,a))^2$$

Proof. Suppose we have n possible observations. Let $P_1(o_i|s,a)$ and $P_2(o_i|s,a)$ be the probabilities of a designated observation o_i in the two processes. We analyze the reward by cases:

- W.p. $P_1(o_i|s,a)P_2(o_i|s,a)$, the same observation o_i is emitted in the two original processes. In this case, the MDP continues to a state sao_i. The rewards that can be obtained are:
  - 0, w.p. $P_1(o_i|s,a)P_2(o_i|s,a)P_1(o_i|s,a)P_2(o_i|s,a)$ (i.e. if both clones also produce o_i)
  - -1 w.p. $P_1(o_i|s,a)P_2(o_i|s,a)[P_1(o_i|s,a)(1 - P_2(o_i|s,a)) + P_2(o_i|s,a)(1 - P_1(o_i|s,a))]$ (i.e. one clone emits o_i, the other emits a different observation)
  - -2 w.p. $P_1(o_i|s,a)P_2(o_i|s,a)[(1 - P_1(o_i|s,a))(1 - P_2(o_i|s,a))]$ (i.e. neither clone emits o_i)

The total expected reward in this case will then be:

$$-P_1(o_i|s,a)P_2(o_i|s,a)[(1-P_1(o_i|s,a))+(1-P_2(o_i|s,a))]$$

– W.p. $P_1(o_i|s,a)P_2(o_j|s,a)$, different observations o_i and o_j are emitted in the two original processes. In this case, the MDP goes to the $Dead$ state. The rewards obtained are:
  - $+2$, w.p. $P_1(o_i|s,a)P_2(o_j|s,a)P_1(o_i|s,a)P_2(o_j|s,a)$ (the clones have identical observations with the original systems)
  - $+1$, w.p. $P_1(o_i|s,a)P_2(o_j|s,a)[(1-P_1(o_i|s,a))P_2(o_j|s,a)+)P_1(o_i|s,a)(1-P_2(o_j|s,a))]$ (one clone emits an observation identical to the one produced by the original system, the other clone emits a different observation)
  - 0, w.p. $P_1(o_i|s,a)P_2(o_j|s,a)[(1-P_1(o_i|s,a))(1-P_2(o_j|s,a))]$ (both clones have different emissions than the originals)

  The total expected reward in this case will be:

$$P_1(o_i|s,a)P_2(o_j|s,a)[P_1(o_i|s,a)+P_2(o_j|s,a)]$$

Now, we can write the immediate expected reward as:

$$\begin{aligned}
\mathcal{R}(s,a) &= \sum_i \sum_{j\neq i} P_1(o_i|s,a)P_2(o_j|s,a)[P_1(o_i|s,a)+P_2(o_j|s,a)] \\
&\quad - \sum_i P_1(o_i|s,a)P_2(o_i|s,a)[(1-P_1(o_i|s,a))+(1-P_2(o_i|s,a))] \\
&= \sum_i \sum_{j\neq i} P_1(o_i|s,a)P_2(o_j|s,a)[P_1(o_i|s,a)+P_2(o_j|s,a)] \\
&\quad + \sum_i P_1(o_i|s,a)P_2(o_i|s,a)[P_1(o_i|s,a)+P_2(o_i|s,a)] \\
&\quad - \sum_i P_1(o_i|s,a)P_2(o_i|s,a)[P_1(o_i|s,a)+P_2(o_i|s,a)] \\
&\quad - \sum_i P_1(o_i|s,a)P_2(o_i|s,a)[(1-P_1(o_i|s,a))+(1-P_2(o_i|s,a))] \\
&= \sum_i \sum_{j} P_1(o_i|s,a)P_2(o_j|s,a)[P_1(o_i|s,a)+P_2(o_j|s,a)] \\
&\quad - 2\sum_i P_1(o_i|s,a)P_2(o_i|s,a) \\
&= \sum_i (P_1(o_i|s,a))^2 + \sum_j (P_2(o_j|s,a))^2 - 2\sum_i P_1(o_i|s,a)P_2(o_i|s,a) \\
&= \sum_i P_1(o_i|s,a)(P_1(o_i|s,a)-P_2(o_i|s,a)) \\
&\quad + \sum_i P_2(o_i|s,a)(P_2(o_i|s,a)-P_1(o_i|s,a)) \\
&= \sum_{o\in\mathcal{O}} (P_1(o|s,a)-P_2(o|s,a))^2
\end{aligned}$$

which concludes the proof. □

We note that the MDP $\mathcal{M}$ has a very special form: it is a tree, with no loops. Using Lemma 1, and the Bellman equation for policy evaluation (see, e.g. [6] for a detailed description) we have:

$$\begin{aligned}
V^{\pi}(s_0) &= \sum_{a_1 \in \mathcal{A}} \pi(a_1|s_0) \sum_{o_1 \in \mathcal{O}} \left[(P_1(o_1|s,a_1) - P_2(o_1|s,a_1))^2 \right. \\
&+ \left. \gamma P_1(o_1|s,a_1) P_2(o_1|s,a_1) V^{\pi}(sa_1o_1) \right] \\
&= \sum_{a_1 \in \mathcal{A}} \pi(a_1|s_0) \sum_{o_1 \in \mathcal{O}} \left[(P_1(o_1|s,a_1) - P_2(o_1|s,a_1))^2 \right] \\
&+ \gamma \sum_{a_1 \in \mathcal{A}} \pi(a_1|s_0) \sum_{o_1 \in \mathcal{O}} \mathcal{T}(s,a_1,sa_1o_1) \sum_{a_2 \in \mathcal{A}} \pi(a_2|sa_1o_1) \\
&\quad \sum_{o_2 \in \mathcal{O}} (P_1(o_2|sa_1o_1,a_2) - P_2(o_2|sa_1o_1,a_2))^2 \\
&+ \gamma^2 \sum_{a_1 \in \mathcal{A}} \pi(a_1|s_0) \sum_{o_1 \in \mathcal{O}} \mathcal{T}(s,a_1,sa_1o_1) \sum_{a_2 \in \mathcal{A}} \pi(a_2|sa_1o_1) \\
&\quad \sum_{o_2 \in \mathcal{O}} \mathcal{T}(sa_1o_1,a_2,sa_1o_1a_2o_2) V^{\pi}(sa_1o_1a_2o_2) \\
&= \ldots
\end{aligned}$$

Since all rewards are strictly positive, and 0 only if the two systems are identical, it is now intuitively clear that all policies will have value 0 if and only if P_1 and P_2 are identical, and strictly positive value otherwise. More formally:

Theorem 1. *Let $\mathcal{M}$ be the MDP induced by P_1 and P_2. If the discount factor $\gamma < 1$ or the size of MDP $|\mathcal{M}| < \infty$ then the optimal value $V^{\star}(s_0) \geq 0$, and $V^{\star}(s_0) = 0$ if, and only if, P_1 and P_2 are trace equivalent.*

The proof is in the appendix.

Note that the size of the MDP increases exponentially with the desired horizon (or depth). However, the computation of the divergence can still be done quite efficiently. In particular, in order to obtain sample complexity bounds, we can use the sparse sampling result of Kearns, Mansour and Ng [7]; they show that the optimal value function of an MDP can be approximated within a desired degree of accuracy ϵ, for all states, in an amount of time which is independent on the number of states in the MDP. The running time still depends on the number of actions; however, in many tasks of interest the number of actions is small, while the number of observations is very large. This would cause the number of states in $\mathcal{M}$ to explode, but the number of samples will still depend mainly on the desired accuracy parameters, the discount factor γ and the maximum reward in the MDP (which in our case is 2).

Moreover, if the two systems are very different, intuitively many sequences will be able to witness this discrepancy. As a result, it may become clear with much fewer samples than this theoretical bound that the systems are different. In the case when we are only interested in the number of samples needed to detect a difference, if one exists, Monte Carlo estimation, along with Hoeffding bounds, can be used to establish tighter bounds. We will now study the speed of the algorithm empirically.

5 Experimental Results

We experimented with this approach on several benchmarks from Tony Cassandra's POMDP repository [8] as well as the hand washing domain [9], which models a real-time system to assist persons with dementia during hand washing. The goal was to see whether this approach can successfully detect differences between models, and to assess the speed with which these can be detected. The experimentation consists in comparing each POMDP with itself and then with slightly modified (noisy) versions of itself. We experimented with introducing noise both in the transition dynamics as well as in the observation probabilities. In both cases, we studied two different kinds of changes: inserting one single important modification, or inserting small perturbations in several places in the model.

The single modification is generated as follows. We choose at random an action a, an internal state of the POMDP and two other internal states reachable through a; then we swap the probabilities of the two next-state transitions. A similar approach is used when perturbing the observation distributions. In the second case, for each state-action pair (s, a), we choose uniformly randomly a number $\delta \in [0, 1]$ and a state in the POMDP. Then, we increase the corresponding transition probability via a by δ. For the remaining states we decrease the transition probabilities by $\delta/(|S| - 1)$. An analogous transformation is applied when we modify the observation distribution.

For each setting, we compare the modified POMDP with the original version. We always perform 10 independent runs, each one consisting of 10^6 episodes. The other systems used for comparison are as follows:

- P_2 : a single modification in the internal dynamics
- P_3 : a single modification in the observation probabilities
- P_4 : small perturbations are inserted throughout in the internal dynamics
- P_5 : small perturbations are inserted throughout in the observation probabilities.

Figure 2 presents some representative learning curves for the value of the divergence. We picked for visualization domains which exhibit typical behavior, as we do not have

Table 1. Summary of empirical results on different POMDPs

	♯ of states	♯ of actions	♯ of observations	Divergence ratios P_2	P_3	P_4	P_5
DataSet4.3	4	4	20	1.35*	2*	2.51*	4.73*
4x3 Maze	11	4	18	1.02	1.023	1.4*	1.99*
Cheese Maze	11	4	7	1.28*	1.01	1.83*	1.72*
Tiger	2	3	6	1.01	1.64*	1.29*	1.77*
Shuttle Docking	8	3	10	1.21*	2.16*	12.26*	9.2*
Paint	4	4	6	1.17	1.16	1.78*	4.79*
MiniHall	13	3	9	4.79*	3.03*	2.74*	3.15*
HallWay	60	5	21	1.0	1.001	1.24*	1.32*
Aloha	30	9	3	1.01	1.37*	2.91*	3.06*
Hand Washing	180	6	6	1.62*	1.56*	1.66*	1.63*

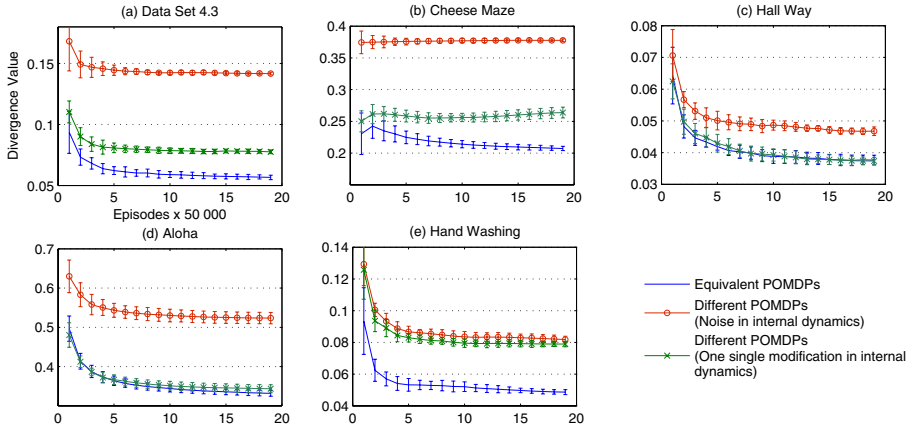

Fig. 2. Learning curves for the divergence values

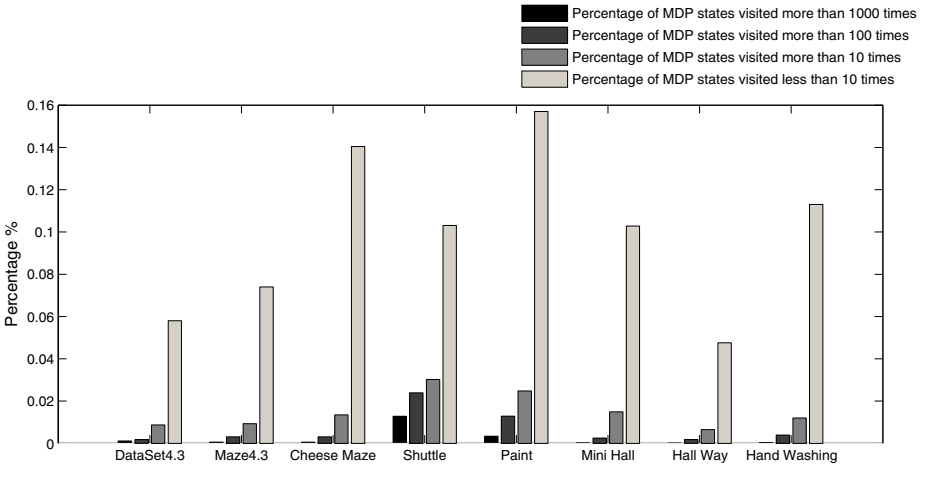

Fig. 3. Statistics on the visited MDP states

space to include the curves for all the benchmarks. As can be seen, in most cases the differences between models are detected very quickly, and the learning curves are significantly different throughout (the bars on the graphs represent standard deviations over the runs). This suggest that one would not even need to run this approach to convergence, if a significant change is detected early on. There are three problems, Hallway, Tiger and Aloha, in which the change in the internal dynamics is not detected correctly. This is because, as it turns out, this one change is not significant enough to affect the probability distribution over observations.

From this figure, one can notice that the magnitude of the divergence values differs from task to task. Hence, in order to decide if two systems are indeed identical or not, we divide the obtained divergence by the divergence that would be obtained by comparing

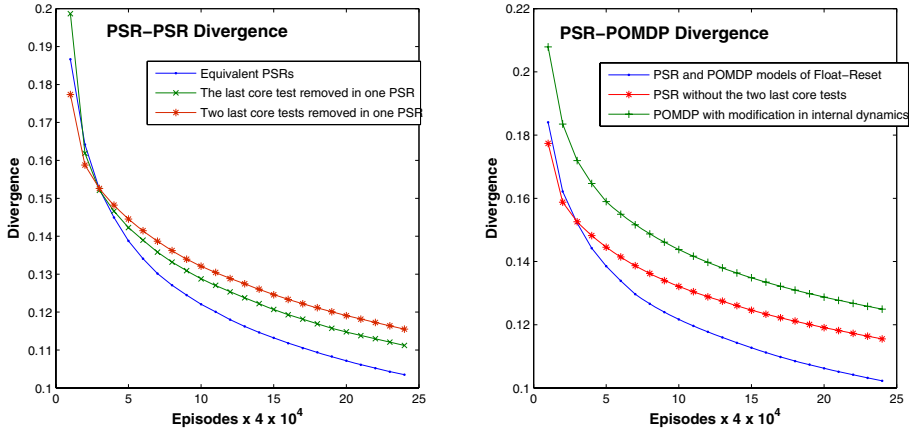

Fig. 4. Comparing (a) two PSRs and (b) a PSR and a POMDP

the system to itself. Hence, two identical systems would be expected to have a ratio of 1, while two very different systems would have a large ratio. Table 1 summarizes the results of the experimentation. The first four columns describe the names of the domains. For each benchmark, we include the number of states, the number of actions, and the number of observations. The remaining columns indicate the divergence ratios. The $*$ symbols indicate statistically significant differences (according to a bootstrap test). From the table, it is clear that almost all differences are detected successfully.

Figure 3 presents statistics on the state visitation frequencies in the MDP. One of the main concerns a priori regarding our method is that the number of states in the MDP that we use is very large (increases exponentially with the size of the horizon). However, because of the special tree structure and because of the particular way we set up the interaction, we expect that only very few states will be visited. Hence, in our implementation, we only maintain values, in a hash table, for states that are actually encountered. For each benchmark, we computed the percentages of states that are visited less than 10 times, between 10 and 100 times, between 100 and 1000 ties, and more than 1000 times. The results confirm the fact that indeed, very few states are ever visited (between 10% and 30% total). A really tiny percentage of states are visited many times. If we stopped the process as soon as significant differences are detected, these percentages would be even lower.

Finally, the last plot (Figure 4) shows the divergence measures computed for predictive state representation (PSR) models. We use the standard float-reset problem from the literature. In the left graph, we compare a PSR representation with itself, against a PSR with one less core test, and against a PSR with two less core tests. In the right graph, we compare PSR and POMDP representations of float-reset. First, we compare the PSR representation with an equivalent POMDP, then against a modified POMDP representation, and finally we compared the original POMDP representation against a PSR with two fewer core tests. The graph shows the potential of our method to detect differences even representations which are not POMDPs.

6 Discussion

Several distance and divergence notions for probabilistic systems have been proposed in the literature. In particular, trace equivalence is a fairly weak notion, and the verification community often relies instead on distance metrics based on bisimulation, a much stronger equivalence notion. Desharnais et al. [10] define a metric for Labelled Markov Processes, which is inspired by the logical characterization of bisimulation due to the same authors. A similar piece of work by Ferns et al. [11] presents a metric for measuring the similarity of states in Markov Decision Processes, again based on the notion of bisimulation for MDPs. For our purposes, bisimulation is too strong, as it essentially establishes a behavior equivalence between systems which requires that they be expressed in the same form. Our definition instead lets us compare systems which may have very different structure (e.g. POMDPs and PSRs). The divergence that we propose is also more efficient to compute and easier to approximate than bisimulation-based metrics. Note that our algorithm generalizes to a family of equivalence notions, called k-moment equivalence (see [1] for details); these equivalences are stronger than trace equivalence but weaker than bisimulation.

In his work on automatic speech recognition, Rabiner developed two distance measures for Hidden Markov Models (HMMs). One is based on a state permutation function [12] and the other is inspired by relative entropy [13]. The connection to KL divergence through relative entropy is interesting, but our setting involves actions, which are chosen actively by the learning algorithm in order to probe the differences in the systems.

The group on Foundations of Software Engineering at Microsoft Research developed a line of research which represents the problem of model-based testing as a game between a tester and the implementation under test. Blass et al. [14] model the game as a negative non-discounted MDP whose goal is to reach final states with minimum total cost. The final states are states in which shared resources are freed, i.e., where new tests can be initiated. Their algorithmic approaches are all offline: all the states are known in advance and the MDP that they set up is solved using a value iteration algorithm. Veanes et al. [15] use instead an online approach for testing through reinforcement learning. The objective of the learning is to achieve the best coverage possible within fixed resource constraints. In each episode the algorithm runs one test case until either a non-conformance between the model (specification) and the implementation under test occurs, or the maximum number of steps has been reached. The cost, or negative reward, associated to an action is proportional to the number of times that actions has been chosen from that state. In both these offline and online approaches, the goal of the learning task is to achieve the best possible coverage. In our work, however, we defined a reward model which is based on the non-conformance between the two processes we compare. We speculate that our approach focuses the computation on more useful paths, though an empirical comparison between these approaches would be useful.

The approach we propose has great potential for two important applications: learning dynamical systems from data, and knowledge transfer. In both cases, the advantage of our method is that it provides an optimal policy, in addition to the divergence. This is very useful information, focusing on the action sequence which distinguishes the systems most.

In the case of learning, it would be natural to take this action sequence and try to refine the model around it (e.g. by adding internal states, in the case of a POMDP, or by adding tests, in the case of a PSR). Alternatively, this action sequence can be played preferentially, along with exploratory paths starting from it. This would provide additional data to improve model parameter estimates, in such a way as to make this critical action path more similar (behaviorally) in the two systems.

In the case of knowledge transfer, the goal is to decide if a policy learned in one system can be applied successfully in a different system. This problem can be formalized as policy evaluation in our MDP, which is very simple and efficient to solve. Hence, the divergence we propose could be used, for example, to search a library of small dynamical systems for one whose optimal policy can be applied successfully in a larger system. We are currently investigating this line of research.

7 Conclusions and Future Work

We presented an approach for estimating the differences between partially observable dynamical systems, based on trying out actions and observing their behavior. Our algorithm is applicable even if the models of the two systems are not available, provided that we are allowed to interact with them. Moreover, the approach allows us to pinpoint the action sequences which most distinguish the systems. Our empirical results are very encouraging, but more practical experience with this approach is needed in the future. One nice aspect is that the algorithm only depends on the number of actions and observations, not on the internal structure of the dynamical system. Moreover, as illustrated above, differences are identified quickly and typically with few interactions with the environment. This is due to the fact that we use reinforcement learning methods, which quickly focus on differences that matter. An early detection mechanism will be implemented in future work.

One important direction for future work is scaling up this approach for systems with many actions and/or observations. In this case, function approximation techniques, or methods by which one expresses interest in a limited set of observations, will need to be used.

Acknowledgements

This work was supported in part by NSERC and FQRNT. The authors thank Prakash Panangaden for helpful discussions.

References

1. Desharnais, J., Laviolette, F., Zhioua, S.: Testing probabilistic equivalence through reinforcement learning. In: Arun-Kumar, S., Garg, N. (eds.) FSTTCS 2006. LNCS, vol. 4337, pp. 236–247. Springer, Heidelberg (2006)
2. Kaelbling, L.P., Littman, M.L., Cassandra, A.R.: Planning and acting in partially observable stochastic domains. Artificial Intelligence 101, 99–134 (1998)

3. Larsen, K.G., Skou, A.: Bisimulation through probabilistic testing. Inf. Comput. 94, 1–28 (1991)
4. Singh, S., James, M., Rudary, M.: Predictive state representations: a new theory for modeling dynamical systems. In: The 20th Conference on Uncertainty in Artificial Intelligence, Banff, Canada, pp. 512–519 (2004)
5. Cover, T.M., Thomas, J.A.: 12. In: Elements of Information Theory. Wiley, Chichester (1991)
6. Sutton, R.S., Barto, A.G.: Introduction to Reinforcement Learning. MIT Press, Cambridge (1998)
7. Kearns, M.J., Mansour, Y., Ng, A.Y.: A sparse sampling algorithm for near-optimal planning in large markov decision processes. Machine Learning 49, 193–208 (2002)
8. Cassandra, T.: Tony's POMDP Page (2009), `http://www.cs.brown.edu/research/ai/pomdp/`
9. Hoey, J., Bertoldi, A., Poupart, P., Mihailidis, A.: Assisting persons with dementia during handwashing using a partially observable markov decision process. In: The 5th International Conference on Computer Vision Systems, Bielefeld, March 21-24 (2007)
10. Desharnais, J., Gupta, V., Jagadeesan, R., Panangaden, P.: Metrics for labeled Markov processes. Theoretical Computer Science 318, 323–354 (2004)
11. Ferns, N., Castro, P., Panangaden, P., Precup, D.: Methods for computing state similarity in markov decision processes. In: Proceedings of the 22nd Conference on Uncertainty in Artificial intelligence, Cambridge, MA, USA, July 13 - 16 (2006)
12. Levinson, S., Rabiner, L.: An introduction to the application of the theory of probabilistic functions of a return process to automatic speech recognition. The Bell System Technical Journal 62, 1035–1074 (1983)
13. Juang, B., Rabiner, L.: A probabilistic distance measure for hidden return models. AT&T Technical Journal 62, 391–408 (1985)
14. Blass, A., Gurevich, Y., Nachmanson, L., Veanes, M.: Play to test. Technical report, Microsoft Research (2005)
15. Veanes, M., Roy, P., Campbell, C.: Online testing with reinforcement learning. In: Havelund, K., Núñez, M., Roşu, G., Wolff, B. (eds.) FATES 2006 and RV 2006. LNCS, vol. 4262, pp. 240–253. Springer, Heidelberg (2006)

Appendix: Proof of Theorem 1

We need the following three lemmas.

Lemma 2. *Let P_1 and P_2 two dynamical systems. Then the following are equivalent:*

(i) *P_1 and P_2 are trace equivalent*
(ii) $\forall a \in \mathcal{A}, s \in \mathcal{S}, \quad \mathcal{R}(s,a) = 0$
(iii) $\forall \pi, \quad V^{\pi}(s_0) = 0.$

Proof. $(i) \Rightarrow (ii)$. If P_1 and P_2 are trace equivalent, then

$$\forall a \in Act, o \in \mathcal{O}, s \in S, P_1(o|s,a) = P_2(o|s,a) \Rightarrow \quad \mathcal{R}(s,a) = 0 \quad \text{(by Lemma 1)}$$

$(ii) \Rightarrow (i)$. Immediate from Lemma 1. $(ii) \Rightarrow (iii)$ and $(iii) \Rightarrow (ii)$ follow from the definition of V^{π} given in Section 4: all policies have value greater than or equal to 0, with equality occurring only if all the squared differences in transition probabilities are 0. □

Now we need we need to link the value of the optimal policy to policies acting to finite depth.

Lemma 3. *For every policy π that prescribes actions only for states up to a maximum depth H,*

$$\exists \pi' \text{ such that } V^{\pi}(s_0) \leq V^{\pi'}(s_0),$$

where π' coincides with π up to depth H.

Proof. The result follows from Lemma 1 and the value function unrolling computation in Section 4. □

Lemma 4. *Let $\mathcal{M}$ be the MDP induced by P_1 and P_2. If the discount factor $\gamma < 1$ or the size of MDP $|\mathcal{M}| < \infty$ then $V^{\star}(s_0) \geq V^{\pi}(s_0)$ for any policy π.*

Proof. If $|\mathcal{M}| < \infty$, since $\mathcal{M}$ has a tree structure, the result is a direct consequence of Lemma 3. Otherwise, it is sufficient to show that

$$\forall \epsilon > 0 \ \ \forall \pi \ \ \exists \pi' \ \text{ such that } \ |V^{\pi'}(s_0) - V^{\pi}(s_0)| < \epsilon.$$

Because of Lemma 3, w.l.o.g., we may suppose n to be large enough to satisfy

$$\sum_{i=n+1}^{\infty} \gamma^i < \epsilon.$$

Since on each episode, the reward signal is between (-2) and $(+2)$, it is easy to see that any policy π' of $\mathcal{M}$ that coincides with π on $\mathcal{M}'$ will have the desired property. □

Theorem 1. *Let $\mathcal{M}$ be the MDP induced by P_1 and P_2. If the discount factor $\gamma < 1$ or the size of MDP $|\mathcal{M}| < \infty$ then the optimal value $V^{\star}(s_0) \geq 0$, and $V^{\star}(s_0) = 0$ if, and only if, P_1 and P_2 are trace equivalent.*

Proof. If P_1 and P_2 are trace equivalent, then Theorem 2 implies that $V^{*}(s_0) = 0$. If they are not trace equivalent, then there exists at least one policy π such that $V^{\pi}(s_0) > 0$. Finally, by Lemma 4, we can conclude that $V^{*}(s_0) > 0$. □