Inverse Reinforcement Learning in Partially Observable Environments

Jaedeug Choi and **Kee-Eung Kim**
Department of Computer Science
Korea Advanced Institute of Science and Technology
Daejeon 305-701, Korea
jdchoi@ai.kaist.ac.kr, kekim@cs.kaist.ac.kr

Abstract

Inverse reinforcement learning (IRL) is the problem of recovering the underlying reward function from the behaviour of an expert. Most of the existing algorithms for IRL assume that the expert's environment is modeled as a Markov decision process (MDP), although they should be able to handle partially observable settings in order to widen the applicability to more realistic scenarios. In this paper, we present an extension of the classical IRL algorithm by Ng and Russell to partially observable environments. We discuss technical issues and challenges, and present the experimental results on some of the benchmark partially observable domains.

1 Introduction

The goal of inverse reinforcement learning (IRL) is to determine the reward function that an agent is optimizing given the agent's behaviour over time [Russell, 1998]. Existing IRL algorithms (*e.g.*, Ng and Russell [2000], Ramanchandran and Amir [2007], and Ziebart *et al.* [2008]) mostly assume that the agent acts in an environment which can be modeled as a Markov decision process (MDP).

Although the MDP assumption provides a good starting point for developing IRL algorithms, the implication is that the agent has access to the true, global state of the environment. The assumption of an omniscient agent is often too strong in practice: even though the agent is assumed to be an expert in the given environment, the agent may be (and often is) making optimal behaviour with a limited sensory capability. Hence, it calls for an extension of IRL algorithms to partially observable environments.

In this paper, we present an IRL algorithm for partially observable environments. Specifically, we assume that the environment is modeled as a partially observable Markov decision process (POMDP), and try to compute the reward function given that the agent follows an optimal policy. Our algorithm is mainly motivated by the classical IRL algorithm by Ng and Russell [2000]. We believe that some of the more recently proposed IRL algorithms could also be extended to handle partially observable environments. The point of this paper is to show the general framework for dealing with partially observable environments, the computational challenges involved in doing so, and some approximation techniques for coping with the challenge.

We believe that our work will prove useful for many problems that could be modeled as POMDPs. For example, the current practice in developing POMDP-based dialogue systems for speech interfaces [Williams and Young, 2007] mostly relies on a labor intensive process: the balance among the reward of a successful dialogue, the penalty of an unsuccessful dialogue, and the cost of information gathering is manually adjusted until a satisfying dialogue policy is obtained from the POMDP. Our work could serve as a foundation for automating the reward specification process (*e.g.*, using the dialogue corpus collected from a wizard-of-oz study) and transferring the reward function to a different dialogue domain (*e.g.*, almost identical reward structure for the travel reservation domain and the appliance control domain).

2 Preliminaries

In this section, we briefly review some of the notations and definitions regarding the models for sequential decision making under uncertainty, namely MDPs and POMDPs. We also briefly overview the IRL algorithm by Ng and Russell [2000] for the sake of presentation of our work.

2.1 MDPs and POMDPs

A Markov decision process (MDP) is defined as a tuple $\langle S, A, T, R, \gamma \rangle$: S is the finite set of states; A is the finite set of actions; T is the state transition function where $T(s, a, s')$ denotes the probability $P(s'|s, a)$ of changing to state s' from state s by taking action a; R is the reward function where $R(s, a)$ denotes the immediate reward of executing action a in state s, whose absolute value is bounded by R_{max}; $\gamma \in [0, 1)$ is the discount factor.

A policy is defined as a mapping $\pi : S \to A$. The value function of policy π is the expected discounted return of executing the policy in the state s:

$$V^\pi(s) = R(s, \pi(s)) + \gamma \sum_{s' \in S} T(s, \pi(s), s') V^\pi(s').$$

Given an MDP, the agent's objective is to find an optimal policy π^* that maximizes the value for all the states, which should satisfy the Bellman equation:

$$V^*(s) = \max_a \left[R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') V^*(s') \right].$$

It is often useful to express the above equation in terms of Q-function: π is an optimal policy if and only if

$$\pi(s) \in \text{argmax}_{a \in A} Q^\pi(s,a)$$

where

$$Q^\pi(s,a) = R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V^\pi(s')$$

The partially observable Markov decision process (POMDP) extends MDP for the agent's noisy observation of the environment. A POMDP is defined as a tuple $\langle S, A, Z, T, O, R, b_0, \gamma \rangle$: S, A, T, R and γ are defined the same as in MDPs; Z is the finite set of observations; O is the observation function where $O(s,a,z)$ denotes the probability $P(z|s,a)$ of perceiving observation z when executing action a and arriving in state s; b_0 is the initial state distribution where $b_0(s)$ denotes the probability of starting in state s.

Since the true state is hidden, the agent has to act based on the history of executed actions and perceived observations. Hence, a policy in POMDP is defined as a mapping from action-observation histories to actions. Since the number of possible histories grows exponentially in the number of time steps, many POMDP algorithms use the concept of *belief*. Formally, the belief b is the probability distribution over the current states, where $b(s)$ denotes the probability that the state is s at the current time step. The belief update for the next time step can be computed from the belief at the current time step: given the action a at the current time step and the observation z at the next time step, the updated belief b_z^a for the next time step is obtained by

$$b_z^a(s') = O(s',a,z) \sum_s T(s,a,s') b(s) / P(z|b,a) \quad (1)$$

where $P(z|b,a) = \sum_{s' \in S} O(s',a,z) \sum_{s \in S} T(s,a,s') b(s)$. Hence, the belief serves as a sufficient statistic for fully summarizing histories, and the policy can be equivalently defined as a mapping from beliefs to actions. Using beliefs, we can view POMDPs as belief-state MDPs, and the value function of an optimal policy satisfies the Bellman equation

$$V^*(b) = \max_a \big[\sum_s b(s) R(s,a) + \gamma \sum_{s',z} T(s,a,s') O(s',a,z) V^*(b_z^a) \big].$$

Alternatively, a policy in POMDP can be represented as a finite state controller (FSC). An FSC policy is defined by a directed graph $\langle \mathcal{N}, \mathcal{E} \rangle$, where each node $n \in \mathcal{N}$ is associated with an action $a \in A$ and has an outgoing edge $e_z \in \mathcal{E}$ per observation $z \in Z$. The policy can be denoted as $\pi = \langle \psi, \eta \rangle$ where ψ is the *action strategy* associating each node n with action $\psi(n) \in A$, and η is the *observation strategy* associating each node n and observation z with a successor node $\eta(n,z) \in \mathcal{N}$.

Given an FSC policy π, the value function V^π is defined over the joint space of FSC nodes and POMDP states: for node n and state s,

$$V^\pi(n,s) = R(s,a) + \gamma \sum_{n',s'} T^{a,os}(\langle n,s \rangle, \langle n',s' \rangle) V^\pi(n',s') \quad (2)$$

where

$$T^{a,os}(\langle n,s \rangle, \langle n',s' \rangle) = T(s,a,s') \sum_{\substack{z \in Z \text{ s.t.} \\ os(z)=n'}} O(s',a,z),$$

with $a = \psi(n)$ and $os(z) = \eta(n,z)$. The value at node n for belief b is calculated by

$$V^\pi(n,b) = \sum_s b(s) V^\pi(n,s), \quad (3)$$

and the starting node for the initial belief b_0 is chosen by $n_0 = \text{argmax}_n V^\pi(n, b_0)$. We can also define Q-function for an FSC policy π:

$$Q^\pi(\langle n,s \rangle, \langle a, os \rangle) = R(s,a) + \gamma \sum_{n',s'} T^{a,os}(\langle n,s \rangle, \langle n',s' \rangle) V^\pi(n',s') \quad (4)$$

which is the expected discounted return of choosing action a at node n and moving to node $os(z)$ upon observation z, and then following policy π. Also the Q-function for node n at belief b is computed by

$$Q^\pi(\langle n,b \rangle, \langle a, os \rangle) = \sum_s b(s) Q^\pi(\langle n,s \rangle, \langle a, os \rangle). \quad (5)$$

With an FSC policy π, we can sort the reachable beliefs into nodes, so that B_n denotes the set of beliefs that are reachable when the current node is n. Note that $|B_n| \geq 1$ for every node n.

2.2 IRL for MDP\R

Formally, the inverse reinforcement learning (IRL) problem in completely-observable Markovian environments is stated as follows: Given an MDP\R $\langle S, A, T, \gamma \rangle$ and an expert's policy π, find the reward function R that makes π an optimal policy for the MDP.

Ng and Russell [2000] present a necessary and sufficient condition for the reward function R of an MDP to guarantee the optimality of π:

$$Q^\pi(s, \pi(s)) \geq Q^\pi(s,a), \quad \forall s, \forall a \quad (6)$$

which states that deviating from the expert's policy should not yield a higher value. This condition can be equivalently reformulated in terms of R:

$$R_{\pi(s)} - R_a + (T_{\pi(s)} - T_a)(I - \gamma T_{\pi(s)})^{-1} R_{\pi(s)} \geq 0, \forall s, \forall a \quad (7)$$

where T_a denotes the $|S| \times |S|$ matrix with the element (s,s') being the transition probability $T(s,a,s')$ and $R_{\pi(s)}$ denotes the $|S|$ vector with the s-th element being the reward $R(s, \pi(s))$.

Given the expert's policy π, which is assumed to be optimal, the reward function is found by solving the optimization problem:

$$\begin{aligned} \text{maximize} \quad & \sum_s \sum_{a \in A \setminus \pi(s)} \big[Q^\pi(s,\pi(s)) - Q^\pi(s,a) \big] - \lambda ||R||_1 \\ \text{subject to} \quad & \text{Constraint (7) and} \\ & |R(s,a)| \leq R_{max}, \forall s, \forall a \end{aligned}$$

where λ is an adjustable weight for the penalty of having too many non-zero entries in the reward function. The idea is to maximize the sum of margins[1] between the expert's policy and all other policies, in the hope that the expert's policy is optimal while favoring the sparseness in the reward function.

[1] We found it more successful to use the sum-of-margins approach than the minimum-of-margins approach in the original paper, since the latter fails when there are multiple optimal policies.

When the expert's policy is not explicitly given but instead the trajectories of the expert's policy in the state and action spaces are available, the algorithm starts with a "base case" random policy π_1. Ideally, the true reward function R should yield $V^*(s) \geq V^{\pi_1}(s)$ since π^* is an optimal policy with respect to R. For a candidate reward function $\hat{R}$, the value of π^* for the starting state s_0 is estimated by the average empirical return:

$$\hat{V}^*(s_0) = \frac{1}{M} \sum_{m=1}^{M} \sum_{t=0}^{T-1} \gamma^t \hat{R}(s_t^m, a_t^m)$$

where $\{s_0^m, s_1^m, \cdots, s_{T-1}^m\}$ and $\{a_0^m, a_1^m, \cdots, a_{T-1}^m\}$ are the T-step state and action sequences in the m-th trajectory of the expert's policy. The values of other policies with candidate reward function $\hat{R}$ are either estimated by sampling trajectories or exactly computed by solving the Bellman equation.

The algorithm iteratively tries to find a better reward function $\hat{R}$, given the set of policies found by the algorithm $\Pi = \{\pi_1, \ldots, \pi_k\}$ up to iteration k, by solving the following optimization problem:

$$\text{maximize } \sum_{\pi \in \Pi} p\left(\hat{V}^*(s_0) - \hat{V}^{\pi}(s_0)\right)$$

where $p(x) = x$ if $x \geq 0$, and $p(x) = 2x$ if $x < 0$. The algorithm then computes a new policy π' that maximizes value function under the new reward function, and add π' to Π. The algorithm continues until it has found a satisfactory reward function.

3 IRL in Partially Observable Environments

In this section, we present our IRL algorithms for partially observable environments. Formally, given an POMDP\R $\langle S, A, Z, T, O, b_0, \gamma \rangle$ and an expert's policy π, we seek the reward function R that makes π an optimal policy for the POMDP. As in IRL for MDP\R, the expert's policy can be given either explicitly in the form of FSC or implicitly by the sampled trajectories of beliefs.

3.1 IRL for POMDP\R from FSC Policies

One of the most natural ways to specify a policy for POMDPs is to use the FSC representation. Hence, when the expert's policy is explicitly given, we will assume that it is represented in the form of an FSC.

We could derive a simple and naive condition for the optimality of the expert's policy, by assuring that the action and observation strategies are optimal at every node, analogous to Equation (6). Given an expert's policy $\pi = \langle \psi, \eta \rangle$,

$$Q^{\pi}(\langle n, b \rangle, \langle \psi(n), \eta(n, \cdot) \rangle) \geq Q^{\pi}(\langle n, b \rangle, \langle a, os \rangle), \quad \forall b \in B_n, \forall a \in A, \forall os \in \mathcal{N}^Z \quad (8)$$

for every node n in π. This inequality states that any policy that deviates from the expert's action and observation strategies should not yield a higher value than that of π. Since it is infeasible to check the value for all possible beliefs, we approximate the condition by comparing the values only for the sampled beliefs reachable by the expert's policy.

However, Equation (8) is not a sufficient condition for the optimality of π: not only we want to compare to the policies with different action and observation strategies, but we also want to take into account other policies that have different number of nodes compared to π, including policies with an infinite number of nodes! Since it is clearly infeasible to include all possible FSC policies into the inequality, we have to resort to a subset of policies that provide a useful feasible region of the reward function. For example, Equation (8) uses the set of $|\mathcal{N}||A||\mathcal{N}|^{|Z|}$ policies that have the same (and possibly fewer) number of nodes as the expert's policy.

A more sound approach to defining the set of policies is to use the set of FSC policies that arise during the dynamic programming update of the expert's policy [Hansen, 1998]. Given the expert's policy π, the dynamic programming update generates $|A||\mathcal{N}|^{|Z|}$ new nodes for all possible action and observation strategies, and these nodes can potentially be a new starting node. This idea comes from the generalized Howards' policy improvement theorem [Howard, 1960]:

Theorem 1 *If an FSC policy is not optimal, dynamic programming update transforms it into an FSC policy with a value function that is as good or better for every belief state and better for some belief state.*

Hence, if the value does not improve for any belief by the dynamic programming update, the expert's policy should be optimal. We should proceed with caution however in the sense that the dynamic programming update does not generate all the necessary nodes to guarantee the optimality of the expert's policy: the nodes in the expert's policy are only those reachable from the starting node n_0, which yields the maximum value at initial belief b_0. Nodes that yield the maximum value at some other belief (*i.e.*, useful) but not reachable from n_0 are not present in the expert's policy. To guarantee the optimality of the expert's policy, we need to include those non-existent but useful nodes in the dynamic programming update, but since there is no way to recover them, we settle for using only the nodes in the expert's policy.

Formally, for the set of newly generated nodes $\mathcal{N}_{new}$, the value function of node $n_{new} \in \mathcal{N}_{new}$ with action strategy of selecting action a and observation strategy os is given by

$$V^{new}(n_{new}, s) = R(s, a) + \gamma \sum_{n', s'} T^{a, os}(\langle n, s \rangle, \langle n', s' \rangle) V^{\pi}(n', s') \quad (9)$$

where V^{π} is the value function of the expert's policy calculated from Equation (2). Note that V^{new} as well as V^{π} are linear in terms of R. Now, the value function of policy π should satisfy

$$V^{\pi}(n, b) \geq V^{new}(n_{new}, b), \ \ \forall b \in B_n, \forall n_{new} \in \mathcal{N}_{new} \quad (10)$$

for every node $n \in \mathcal{N}$ in order to guarantee the optimality of the expert's policy π.

A more computationally efficient way to generate the set of policies is to use the witness theorem [Kaelbling *et al.*, 1998]. Re-stated in terms of FSC policies, we have the following:

Theorem 2 *Let U_a be a nonempty set of useful nodes with the action strategy of choosing action a, and $\mathcal{N}^a$ be the complete set of useful nodes with the action strategy of choosing action a. Then $U_a \neq \mathcal{N}^a$ if and only if there is some node $\tilde{n} \in U_a$, observation z, and $n' \in \mathcal{N}$ for which there is some belief b such that*

$$V^{new}(n_{new}, b) \geq V^{\pi}(n, b)$$

for all $n \in U_a$, where n_{new} is a node that agrees with $\tilde{n}$ in its action and all its successor nodes except for observation z, for which $\eta(n_{new}, z) = n'$.

The witness theorem tells us that if we generate all possible n_{new}'s by changing the successor node of each single observation and check the value for all possible beliefs, we can guarantee the optimality of π. This leads us to the same inequality constraint as Equation (10), except that $\mathcal{N}_{new}$ is defined in a different way. For each action a, we prepare $U_a = \{n \in \mathcal{N} | \psi(n) = a\}$ and use the witness theorem to generate $|\mathcal{N}|^2|Z|$ new nodes. We then use the dynamic programming update to generate $|A_{-\mathcal{N}}||\mathcal{N}|^{|Z|}$ additional nodes, where $A_{-\mathcal{N}}$ is the set of actions that is not selected at any of the nodes in $\mathcal{N}$. Hence, using the witness theorem, we have $|\mathcal{N}_{new}| \leq |\mathcal{N}|^2|Z| + |A_{-\mathcal{N}}||\mathcal{N}|^{|Z|}$, often much fewer compared to those generated by only using the dynamic programming update.

Let us now turn our attention to the complete optimization problem. If we use the optimality constraint in Equation (8), we would like to maximize the sum of margins between the expert's policy and other policies, while making the reward function as sparse as possible:

$$\max \sum_{n\in\mathcal{N}} \sum_{b\in B_n} \sum_{\substack{a\in A\setminus\psi(n) \\ os\in\mathcal{N}^Z\setminus\eta(n,\cdot)}} [V^\pi(\langle n,b\rangle) - Q^\pi(\langle n,b\rangle, \langle a, os\rangle)] - \lambda\|R\|_1$$

$$\text{subject to} \quad \text{Constraints (2), (3), (4), (5), (8), and } |R(s,a)| \leq R_{max}, \forall s, \forall a$$

If we use the dynamic programming update or the witness theorem, the policies other than the expert's policy are captured in n_{new}'s, hence the optimization problem now becomes:

$$\max \sum_{n\in\mathcal{N}} \sum_{b\in B_n} \sum_{n_{new}\in\mathcal{N}_{new}} [V^\pi(n,b) - V^{new}(n_{new}, b)] - \lambda\|R\|_1$$

$$\text{subject to} \quad \text{Constraints (2), (3), (9), (10), and } |R(s,a)| \leq R_{max}, \forall s, \forall a$$

where $\mathcal{N}_{new}$ is the set of nodes generated by the dynamic programming update, or by the witness-based method.

3.2 IRL for POMDP\R from Sampled Trajectories

In many cases, the expert's policy may not be explicitly given, but the set of trajectories may be available instead. Here, we assume that it is the set of T-step belief trajectories, where the m-th trajectory is denoted by $\{b_0^m, b_1^m, \ldots, b_{T-1}^m\}$. If the observation trajectories $\{z_0^m, z_1^m, \ldots, z_{T-1}^m\}$ and the action trajectories $\{a_0^m, a_1^m, \ldots, a_{T-1}^m\}$ are available instead, we can reconstruct the belief trajectories by using the belief update in Equation (1).

In order to derive an IRL algorithm for POMDP\R from the sampled belief trajectories, we make the same assumption as in Ng and Russell [2000] about the reward function

$$R(s,a) = \alpha_1\phi_1(s,a) + \alpha_2\phi_2(s,a) + \cdots + \alpha_d\phi_d(s,a)$$

where $\phi_1, \ldots, \phi_d$ are the known basis functions mapping from $S \times A$ to $\Re$, and the α_i's are the unknown parameters.

Choose a number of beliefs $b_j, j = 1, \ldots, J$.
Choose a random initial policy π_1, and set $\Pi = \{\pi_1\}$.
for $K = 1$ **to** $MaxIter$
  Find $\hat{\alpha}_i$'s by solving the linear program:
    maximize $\sum_{j=1}^{J}\sum_{k=1}^{K} t_{jk}$
    subject to $\hat{V}^\pi(b_j) - V^{\pi_k}(b_j) \geq t_{jk}$, $t_{jk} \geq 0$, $|\hat{\alpha}_i| \leq 1$
  Find π_{K+1} that is optimal w.r.t. the reward function
    $\hat{R}(s,a) = \hat{\alpha}_1\phi_1(s,a) + \hat{\alpha}_2\phi_2(s,a) + \cdots + \hat{\alpha}_d\phi_d(s,a)$
  if $V^{\pi_{K+1}}(b_j) - \hat{V}^\pi(b_j) \leq \epsilon, \forall j$ **then return** $\hat{R}$
  else $\Pi = \Pi \cup \{\pi_{K+1}\}$
return $\hat{R}$

Figure 1: IRL for POMDP\R from the sampled trajectories.

The reward for belief b is then calculated by

$$\begin{aligned} R(b,a) &= \textstyle\sum_s b(s)R(s,a) \\ &= \alpha_1 \textstyle\sum_s b(s)\phi_1(s,a) + \cdots + \alpha_d \sum_s b(s)\phi_d(s,a) \\ &= \alpha_1\phi_1(b,a) + \cdots + \alpha_d\phi_d(b,a) \end{aligned}$$

where $\phi_i(b,a) = \sum_s b(s)\phi_i(s,a)$. We also define $\hat{V}_i^\pi(b_0^m)$ to be the average empirical return of the m-th trajectory when using each basis function as the reward function ($R = \phi_i$)

$$\hat{V}_i^\pi(b_0^m) = \phi_i(b_0^m, a_0^m) + \gamma\phi_i(b_1^m, a_1^m) + \cdots + \gamma^{T-1}\phi_i(b_{T-1}^m, a_{T-1}^m)$$

so that we have

$$\hat{V}^\pi(b_0^m) = \alpha_1\hat{V}_1^\pi(b_0^m) + \cdots + \alpha_d\hat{V}_d^\pi(b_0^m).$$

Noting that $b_0^m = b_0$ for all m, the average empirical return at b_0 is given by

$$\hat{V}^\pi(b_0) = \tfrac{1}{M}\textstyle\sum_{m=1}^{M} \hat{V}^\pi(b_0^m) \tag{11}$$

which is linear in terms of $\alpha_1, \ldots, \alpha_d$.

Given the definitions above, the rest of the derivation is fairly straightforward, and leads to a similar algorithm to that of Ng and Russell [2000], shown in Figure 1. The algorithm iteratively tries to find a reward function parameterized by α_i's that maximizes the sum of the differences between the $\hat{V}^\pi$ of the expert's policy and V^{π_k} of each FSC policy $\pi_k \in \Pi$ found so far. The differences in the values are measured at the representative beliefs b_j's, which are chosen if they appear frequently in the early time steps of the trajectories, ensuring low estimation error in the values. We could use the initial belief b_0 as the only representative belief (similar to Ng and Russell [2000] using only s_0), but we found it more effective in our experiments to include additional beliefs since they often provide a better guidance in the search of the reward function by tightening the feasible region. The average empirical return of the expert's trajectories at the representative belief b_j is computed by

$$\begin{aligned} \hat{V}^\pi(b_j) &= \frac{1}{M_j}\sum_{m=1}^{M} \hat{V}^\pi(b_j^m) = \frac{1}{M_j}\sum_{m=1}^{M}\sum_{i=1}^{d} \alpha_i\hat{V}_i^\pi(b_j^m) \\ &= \frac{1}{M_j}\sum_{m=1}^{M}\sum_{i=1}^{d}\alpha_i \sum_{t=T_j^m}^{T-1} \gamma^{t-T_j^m}\phi_i(b_t^m, a_t^m) \end{aligned} \tag{12}$$

where T_j^m is the first time that b_j is found in m-th trajectory and M_j is the number of trajectories that contain b_j. However, computing $V^{\pi_k}(b_j)$ is not well defined since b_j may be unreachable under π_k. In our work, we use an upperbound approximation $V^{\pi_k}(b_j) \approx \max_n V^{\pi_k}(n, b_j)$, where $V^{\pi_k}(n, b_j)$ is computed by Equation (3).

When $\pi_{K+1} = \pi$, the differences in the value function for all representative beliefs will vanish. Hence, the algorithm terminates when all the differences in the values are below the threshold ϵ, or the iteration number has reached the maximum number of steps $MaxIter$.

4 Experiments

The first set of experiments concerns with the case when the expert's policy is explicitly given using the FSC representation. We chose three POMDP problems for the experiments: Tiger, 1d maze, and 5×5 grid world. The tiger and the 1d maze problems are classic benchmark problems in the POMDP literature, and their specifications can be found in [Cassandra *et al.*, 1994]. The 5×5 grid world problem is inspired by the one in [Ng and Russell, 2000], where the agent can move west, east, north or south, and their effect is assumed to be deterministic. The agent always starts from the north-west corner of the grid and aims to reach the goal in the south-east corner. The current position cannot be observed directly but the presence of the walls can be perceived without any noise when the agent is on the border of the grid. Hence, there are nine observations, eight of them corresponding to eight possible configurations of the nearby walls when on the border, and one corresponding to no wall observation when not on the border. The optimal policies for the problems were computed using PBPI [Ji *et al.*, 2007], and used as the expert's policies.

We experimented with all three approaches in Section 3.1: the Q-function based approach, the DP-update based approach, and the witness theorem based approach. As in the case of IRL for MDP\R, we were able to control the sparseness in the reward function by tuning the penalty weight λ. With the suitable value for λ, all three approaches yielded the same reward function. The estimated reward function is compared to the true reward function in Figure 2 for the tiger and 1d maze problems, and in Figure 3 for the 5×5 grid world problem. We discuss each result below.

In the tiger problem (top of Figure 2), the true reward function is not sparse since every action is associated with a non-zero reward. Since our methods favor sparse reward functions, there is some degree of difference between the true and the learned reward functions, most notably for the listen action, where our methods assign zero reward instead of -1 as in the true reward. However, we can apply the policy-invariant reward transformation [Ng *et al.*, 1999] on the true reward function so that listen action yields zero reward. The learned reward function is very close to the transformed one, as shown in the figure.

In the 1d maze problem (bottom of Figure 2), the expert's policy has three nodes: node n_0 (the starting node) chooses to move right, and changes to node n_1 upon observing *nothing* or to node n_2 upon observing *goal*; node n_1 chooses to move right and always changes to node n_2; node n_2 chooses to move left, and changes to node n_0 upon observing *nothing* or to itself upon observing *goal*. If we follow the expert's policy, the probability of moving right is larger than that of moving left in the left and middle states. Furthermore, the probability of moving right in the middle state is larger than that of moving right in the left state. Hence, the algorithm assigns negative rewards for moving left in the left and middle states, and even a lower reward for the middle state than that for the left state. However, the optimal policy from the POMDP with the learned reward function is the same as the expert's policy.

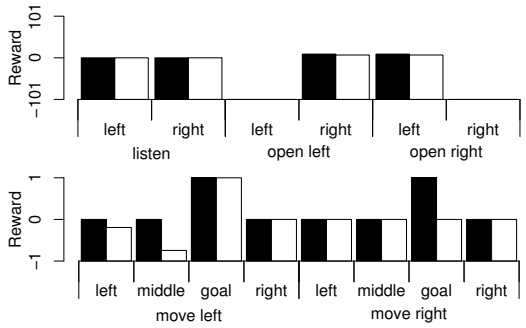

Figure 2: Result of IRL from FSC policies. *Top*: Tiger problem. *Bottom*: 1d maze problem. *Black bars*: The true reward. *White bars*: The estimated reward.

Finally, in the 5×5 grid world problem (Figure 3), the expert's policy is very simple: the agent moves east until it observes the north-east corner, and moves south until it observes the south-east corner. In the figure, the states are numbered sequentially from the start state, so that state 0 corresponds to the north-west corner, state 4 corresponds to the north-east corner, and state 24 corresponds to the south-east corner. Our methods assign positive reward for moving east in the goal state, and zero or negative reward for all other pairs of state and action. Again, even though the learned reward function is different from the true one, it yields the same optimal policy.

The second set of experiments is for the case when the expert's trajectories are given. We experimented on the same set of three problems, with the basis functions chosen as follows: for the tiger and the 1d maze problems, we used the table representation for the reward function, *i.e.*, each pair of state and action as its own basis function; for the 5×5 grid world problem, we prepared six basis functions, five for the states in the south border, and one for all other states. We sampled 5000 belief trajectories each of length 50 time steps. In each iteration K of the algorithm (Figure 1), we evaluated π_{K+1} using the true reward. Figure 4 shows the mean values and the standard deviations of 100 trials on the tiger and the 1d maze problems. For the tiger and the 1d maze problems, our algorithm took an average of 3.91 and 1.97 iterations respectively. The learned reward upon termination yielded the optimal policy in 90% and 98% of the trials for the tiger and the 1d maze problems respectively. The overall average value of the policies was 1.54 for the tiger problem (optimal policy yields 1.93), and 1.19 for the 1d maze problem (optimal policy yields 1.20). We excluded the result on the 5×5 grid

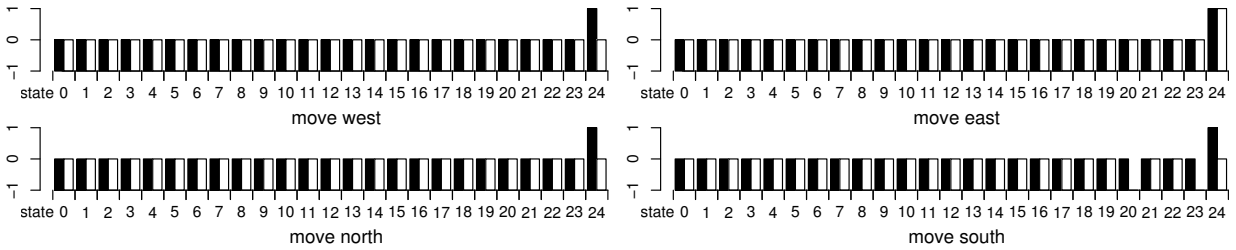

Figure 3: IRL from FSC policy in 5×5 grid world. *Black bars*: The true reward. *White bars*: The estimated reward.

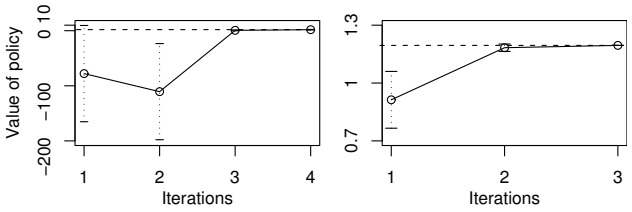

Figure 4: Results of IRL from sampled trajectories. *Left*: Tiger problem. *Right*: 1d maze problem.

world problem in Figure 4, since the optimal reward is found in only 1 iteration for all the trials.

5 Conclusion and Future Work

We presented an IRL framework for dealing with partially observable environments. Experimental results on POMDP benchmark domains show that, in most cases, our algorithm robustly finds solutions close to the true reward functions generating exactly the same optimal policy. The presented work is only a demonstration of how the existing body of work on IRL on MDP\R could be extended to POMDP\R, and more recent IRL techniques as well as some of the IRL-based apprenticeship learning techniques could be similarly extended by following our line of thought.

There are a number of challenges that should be addressed in the future. First, the scalability of our methods should be improved, so that we can handle more realistic problems with large policies, such as dialogue management problems with hundreds of policy nodes. Second, it remains as an open problem whether there exists a sufficient condition for optimality and a tractable method exploiting it. Even we use the witness theorem as a non-sufficient condition, we may end up with a huge number of constraints. We could sample from the exhaustive set of conditions, but the theoretical analyses should be accompanied. Third, some of the issues regarding the convergence to suboptimal rewards could be addressed by more recent techniques, such as Abbeel and Ng [2004] and Ramanchandran and Amir [2007].

6 Acknowledgments

This work was supported by Korea Research Foundation Grant KRF-D00527, and by Defense Acquisition Program Administration and Agency for Defense Development of Korea under contract UD080042AD.

References

[Abbeel and Ng, 2004] Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of ICML*, 2004.

[Cassandra *et al.*, 1994] Anthony R. Cassandra, Leslie Pack Kaelbling, and Michael L. Littman. Acting optimally in partially observable stochastic domains. In *Proceedings of AAAI*, 1994.

[Hansen, 1998] Eric A. Hansen. *Finite-Memory Control of Partially Observable Systems*. PhD thesis, Department of Computer Science, University of Massachusetts at Amherst, 1998.

[Howard, 1960] Ronald A. Howard. *Dynamic Programming and Markov Processes*. MIT Press, Cambridge, MA, 1960.

[Ji *et al.*, 2007] Shihao Ji, Ronald Parr, Hui Li, Xuejun Liao, and Lawrence Carin. Point-based policy iteration. In *Proceedings of AAAI*, 2007.

[Kaelbling *et al.*, 1998] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101, 1998.

[Ng and Russell, 2000] Andrew Y. Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *Proceedings of ICML*, 2000.

[Ng *et al.*, 1999] Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of ICML*, 1999.

[Ramachandran and Amir, 2007] Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *Proceedings of IJCAI*, 2007.

[Russell, 1998] Stuart Russell. Learning agents for uncertain environments (extended abstract). In *Proceedings of COLT*, 1998.

[Williams and Young, 2007] Jason D. Williams and Steve Young. Partially observable Markov decision processes for spoken dialog systems. *Computer Speech and Language*, 27, 2007.

[Ziebart *et al.*, 2008] Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of AAAI*, 2008.