Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling (ICAPS 2022)

Task-Guided Inverse Reinforcement Learning under Partial Information

Franck Djeumou[1], **Murat Cubuktepe**[1], **Craig Lennon**[2], **and Ufuk Topcu**[1]

[1] The University of Texas at Austin
[2] Army Research Laboratory
fdjeumou@utexas.edu, mcubuktepe@utexas.edu, craig.t.lennon.civ@mail.mil, utopcu@utexas.edu

Abstract

We study the problem of inverse reinforcement learning (IRL), where the learning agent recovers a reward function using expert demonstrations. Most of the existing IRL techniques make the often unrealistic assumption that the agent has access to full information about the environment. We remove this assumption by developing an algorithm for IRL in partially observable Markov decision processes (POMDPs). The algorithm addresses several limitations of existing techniques that do not take the *information asymmetry* between the expert and the learner into account. First, it adopts causal entropy as the measure of the likelihood of the expert demonstrations as opposed to entropy in most existing IRL techniques, and avoids a common source of algorithmic complexity. Second, it incorporates task specifications expressed in temporal logic into IRL. Such specifications may be interpreted as side information available to the learner a priori in addition to the demonstrations and may reduce the information asymmetry. Nevertheless, the resulting formulation is still nonconvex due to the intrinsic nonconvexity of the so-called *forward problem*, i.e., computing an optimal policy given a reward function, in POMDPs. We address this nonconvexity through sequential convex programming and introduce several extensions to solve the forward problem in a scalable manner. This scalability allows computing policies that incorporate memory at the expense of added computational cost yet also outperform memoryless policies. We demonstrate that, even with severely limited data, the algorithm learns reward functions and policies that satisfy the task and induce a similar behavior to the expert by leveraging the side information and incorporating memory into the policy.

Introduction

Inverse reinforcement learning (IRL) is a technique that recovers a reward function using expert demonstrations and learns a policy inducing a similar behavior to the expert's. IRL techniques have found a wide range of applications (Abbeel, Coates, and Ng 2010; Kitani et al. 2012; Hadfield-Menell et al. 2016; Dragan and Srinivasa 2013; Finn, Levine, and Abbeel 2016). The majority of the work has focused on Markov decision processes (MDPs), assuming that the learning agent can fully observe the state of the environment and the expert's demonstrations (Abbeel, Coates, and Ng 2010; Ziebart et al. 2008; Zhou, Bloem, and Bambos 2017; Ziebart, Bagnell, and Dey 2010; Hadfield-Menell et al. 2016; Finn, Levine, and Abbeel 2016). Often, in reality, the learning agent will not have such full observation. For example, a robot will never know everything about its environment (Ong et al. 2009; Bai, Hsu, and Lee 2014; Zhang et al. 2017) and may not observe the internal states of a human with whom it works (Akash et al. 2019; Liu and Datta 2012). Such information limitations violate the intrinsic assumptions made in existing IRL techniques.

We study IRL in partially observable Markov decision processes (POMDPs), a widely used model for decision-making under imperfect information. The partial observability brings three key challenges in IRL. The first two challenges are related to the so-called *information asymmetry* between the expert and the learner. First, the expert typically has access to full information about the environment, while the learner has only a partial view of the expert's demonstrations. Second, even in the hypothetical case in which the actual reward function is known to the learner, the learner's optimal policy under limited information may not yield the same behavior as the expert due to information asymmetry.

The third challenge is due to the computational complexity of policy synthesis in POMDPs. Many standard IRL techniques rely on a subroutine that solves the so-called *forward problem*, i.e., computing an optimal policy for a given reward. Solving the forward problem for POMDPs is significantly more challenging than MDPs, both theoretically and practically. Optimal policies for POMDPs may require infinite memory of observations (Madani, Hanks, and Condon 1999), whereas memoryless policies are enough for MDPs.

An additional limitation in existing IRL techniques is due to the limited expressivity and often impracticability of state-based reward functions in representing complex tasks (Littman et al. 2017). For example, it will be tremendously difficult to define a merely state-based reward function to describe requirements such as "do not steer off the road while reaching the target location and coming back to home" or "monitor multiple locations with a certain order." On the other hand, such requirements can be concisely and precisely specified in temporal logic (Baier and Katoen 2008; Pnueli 1977). Recent work has demonstrated the utility of incorporating temporal logic specifications into IRL in MDPs (Memarian et al. 2020; Wen, Papusha, and Topcu

2017). In this work, we address these challenges and limitations in IRL techniques by studying the problem:

> **Task-guided IRL:** Given a POMDP, a *task specification* φ expressed in temporal logic, and a set of expert demonstrations, learn a policy along with the underlying reward function that maximizes the *causal entropy* of the induced stochastic process, induces a behavior similar to the expert's, and ensures satisfaction of φ.

We highlight two parts of the problem statement. Using *causal entropy* as an optimization criterion results in a least-committal policy that induces a behavior obtaining the same accumulated reward as the expert's demonstrations while making no additional assumptions about the demonstrations. Given the task requirements, the *task specifications* guide the learning process by describing the feasible behaviors and allowing to learn performant policies with respect to the task requirements. Such specifications can also be interpreted as side information available to the learner a priori in addition to the demonstrations and partially alleviates the information asymmetry between the expert and the learner.

Most existing work on IRL relies on *entropy* as a measure of the likelihood of the demonstrations, yet, when applied to stochastic MDPs, has to deal with nonconvex optimization problems (Ziebart et al. 2008; Ziebart, Bagnell, and Dey 2010). On the other hand, IRL techniques that adopt *causal entropy* as the measure of likelihood enjoy formulations based on convex optimization (Zhou, Bloem, and Bambos 2017; Ziebart, Bagnell, and Dey 2010, 2013). We show similar algorithmic benefits in maximum-causal-entropy IRL carry over from MDPs to POMDPs.

A major difference between MDPs and POMDPs in maximum-causal-entropy IRL is, though, due to the intrinsic nonconvexity of policy synthesis in POMDPs, which yields a formulation of the task-guided IRL problem as a nonconvex optimization. It is known that this nonconvexity severely limits the scalability for synthesis in POMDPs. We develop an algorithm that solves the resulting nonconvex problem in a scalable manner by adapting sequential convex programming (SCP) (Yuan 2015; Mao et al. 2018). The algorithm is iterative. In each iteration, it linearizes the underlying nonconvex problem around the solution from the previous iteration. The algorithm introduces several extensions, among which a verification step not present in existing SCP schemes. We show that it computes a sound and locally optimal solution to the task-guided IRL problem.

The algorithm scales to POMDPs with tens of thousands of states as opposed to tens of states in the existing work, e.g., belief-based or off-the-shelf nonconvex optimization solvers. In POMDPs, *finite-memory* policies that are functions of the history of the observations outperform memoryless policies (Yu and Bertsekas 2008). Computing a finite-memory policy for a POMDP is equivalent to computing a memoryless policy on a larger product POMDP (Junges et al. 2018). On the other hand, existing IRL techniques on POMDPs cannot effectively utilize memory, as they do not scale to large POMDPs. We leverage the scalability of our algorithm to compute performant policies that incorporate memory using finite-state controllers (Meuleau et al. 1999; Amato, Bernstein, and Zilberstein 2010).

We demonstrate the applicability of the approach through several examples. We show that, without task specifications, the developed algorithm can compute more performant policies than a straight adaptation of the original GAIL (Ho and Ermon 2016) to POMDPs. Then, we demonstrate that by incorporating task specifications into the IRL procedure, the learned reward function and policy accurately describe the behavior of the expert while outperforming the policy obtained without the task specifications. Additionally, we show that incorporating memory into the learning agent's policy leads to more performant policies. We also show that with more limited data, the performance gap becomes more prominent between the learned policies with and without using task specifications. Finally, we demonstrate the scalability of our approach for solving the *forward problem* through extensive comparisons with several state-of-the-art POMDP solvers and show that on larger POMDPs, the algorithm can compute more performant policies in significantly less time.

Related work. The closest work to ours is by Choi and Kim (2011), where they extend classical maximum-margin-based IRL techniques for MDPs to POMDPs. However, even on MDPs, maximum-margin-based approaches cannot resolve the ambiguity caused by suboptimal demonstrations, and they work well when there is a single reward function that is clearly better than alternatives (Osa et al. 2018). In contrast, we adopt causal entropy that has been shown (Osa et al. 2018; Ziebart, Bagnell, and Dey 2010) to alleviate these limitations on MDPs. Besides, Choi and Kim (2011) rely on efficient off-the-shelf solvers to the forward problem. Instead, this paper also develops an algorithm that outperforms off-the-shelf solvers and can scale to POMDPs that are orders of magnitude larger compared to the examples in Choi and Kim (2011). Further, Choi and Kim (2011) do not incorporate task specifications in their formulations.

Prior work tackled the ill-posed IRL problem using maximum margin formulations (Ratliff, Bagnell, and Zinkevich 2006; Abbeel and Ng 2004; Ng, Russell et al. 2000), or probabilistic models to compute the likelihood of expert demonstrations (Ramachandran and Amir 2007; Ziebart et al. 2008; Ziebart, Bagnell, and Dey 2010; Zhou, Bloem, and Bambos 2017; Finn, Levine, and Abbeel 2016; Ho and Ermon 2016). Besides, the idea of using side information to guide and augment IRL has been explored in recent work (Papusha, Wen, and Topcu 2018; Wen, Papusha, and Topcu 2017; Memarian et al. 2020). However, these IRL techniques are only applicable to MDPs as opposed to POMDPs.

IRL under some restricted notion of partial information has been studied in prior work. Boularias, Krömer, and Peters (2012) consider the setting where the features of the reward function are partially specified. Kitani et al. (2012); Bogert and Doshi (2014) consider IRL problems from partially observable demonstrations and use the hidden Markov decision process framework as a solution. Therefore, all these approaches consider a particular case of POMDPs. We also note that none of these methods incorporate side information into IRL and do not provide guarantees on the performance of the policy with respect to a task specification.

Background

Notation. We denote the set of nonnegative real numbers by $\mathbb{R}_+$, the set of all probability distributions over a finite or countably infinite set $\mathcal{X}$ by $\mathrm{Distr}(\mathcal{X})$, the set of all (infinite or empty) sequences $x_0, x_1, \ldots, x_\infty$ with $x_i \in \mathcal{X}$ by $(\mathcal{X})^*$ for some set $\mathcal{X}$, and the expectation of a function g of jointly distributed random variables X and Y by $\mathbb{E}_{X,Y}[g(X,Y)]$.

POMDPs. A partially observable Markov decision process (POMDP) is a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{Z}, \mathcal{P}, \mathcal{O}, \mathcal{R}, \mu_0, \gamma)$, with finite sets $\mathcal{S}$, $\mathcal{A}$ and $\mathcal{Z}$ denoting the set of states, actions, and observations, respectively, a transition function $\mathcal{P} : \mathcal{S} \times \mathcal{A} \mapsto \mathrm{Distr}(\mathcal{S})$, an observation function $\mathcal{O} : \mathcal{S} \mapsto \mathrm{Distr}(\mathcal{Z})$, a reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}_+$, an initial state of distribution $\mu_0 \in \mathrm{Distr}(\mathcal{S})$, and a discount factor $\gamma \in (0,1)$. We denote $\mathcal{P}(s'|s,\alpha)$ as the probability of transitioning to state s' after an action α is selected in state s, and $\mathcal{O}(z|s)$ is the probability of observing $z \in \mathcal{Z}$ in state s.

Policies. An observation-based policy $\sigma : (\mathcal{Z}\times\mathcal{A})^* \times \mathcal{Z} \mapsto \mathrm{Distr}(\mathcal{A})$ for a POMDP $\mathcal{M}$ maps a sequence of observations and actions to a distribution over actions. A M-*finite-state controller* (M-FSC) consists of a finite set of memory states of size M and two functions. The *action mapping* $\eta(n,z)$ takes a FSC memory state n and an observation $z \in \mathcal{Z}$, and returns a distribution over the POMDP actions. The *memory update* $\delta(n,z,\alpha)$ returns a distribution over memory states and is a function of the action α selected by η. An FSC induces an observation-based policy by following a joint execution of these two functions upon a trace of the POMDP. *Memoryless FSCs, denoted by* $\sigma\colon \mathcal{Z} \to \mathrm{Distr}(\mathcal{A})$*, are observation-based policies, where* $\sigma_{z,\alpha}$ *is the probability of taking the action* α *given solely observation* z*.*

Remark 1 (REDUCTION TO MEMORYLESS POLICIES)**.** *In the remainder of the paper, for ease of notation, we synthesize optimal* M*-FSCs for POMDPs (so-called forward problem) by computing memoryless policies* σ *on theoretically-justified larger POMDPs obtained from the so-called product of the memory update* δ *and the original POMDPs. Indeed, Junges et al. (2018) provide product POMDPs, whose sizes grow polynomially with the size of the domain of* δ*.*

Causal Entropy in POMDPs. For a POMDP $\mathcal{M}$, a policy σ induces the stochastic processes $S^\sigma_{0:\infty} := (S^\sigma_0, \ldots, S^\sigma_\infty)$, $A^\sigma_{0:\infty} := (A^\sigma_0, \ldots, A^\sigma_\infty)$, and $Z^\sigma_{0:\infty} := (Z^\sigma_0, \ldots, Z^\sigma_\infty)$. At each time index t, the random variables S^σ_t, A^σ_t, and Z^σ_t take values $s_t \in \mathcal{S}$, $\alpha_t \in \mathcal{A}$, and $z_t \in \mathcal{Z}$, respectively.

We consider the infinite time horizon setting and define the discounted causal entropy as (Zhou, Bloem, and Bambos 2017): $H^\gamma_\sigma := \sum_{t=0}^\infty \gamma^t \mathbb{E}_{A^\sigma_t, S^\sigma_t}[-\log \mathbb{P}(A^\sigma_t | S^\sigma_t)]$.

Remark 2. *The* entropy *of POMDPs (or MDPs) involves the* future policy decisions *(Ziebart et al. 2008), i.e.,* $S^\sigma_{t+1:T}$*, at a time index* t*, as opposed to the causal entropy in POMDPs (or MDPs). Thus, Ziebart et al. (2008) show that the problem of computing a policy that maximizes the entropy is nonconvex, even in MDPs. Inverse reinforcement learning techniques that maximize the entropy of the policy rely on approximations or assume that the transition function of the MDP is deterministic. On the other hand, computing a policy that maximizes the causal entropy can be formulated as a convex optimization problem in MDPs (Ziebart, Bagnell, and Dey 2010; Zhou, Bloem, and Bambos 2017).*

LTL Specifications. We use general linear temporal logic (LTL) to express complex task specifications on the POMDP $\mathcal{M}$. Given a set AP of atomic propositions, i.e., Boolean variables with truth values for a given state s or observation z, LTL formulae are constructed inductively as following: $\varphi := \mathrm{true} \mid a \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \mathbf{X}\varphi \mid \varphi_1 \mathbf{U} \varphi_2$, where $a \in \mathrm{AP}$, φ, φ_1, and φ_2 are LTL formulae, $\neg$ and $\wedge$ are the logic negation and conjunction, and $\mathbf{X}$ and $\mathbf{U}$ are the *next* and *until* temporal operators. Besides, temporal operators such as *always* (**G**) and *eventually* (**F**) are derived as $\mathbf{F}\varphi := \mathrm{true}\mathbf{U}\varphi$ and $\mathbf{G}\varphi := \neg\mathbf{F}\neg\varphi$. A detailed description of the syntax and semantics of LTL is beyond the scope of this paper and can be found in Pnueli (1977); Baier and Katoen (2008).

$\Pr^\sigma_\mathcal{M}(\varphi)$ *denotes the probability of satisfying the LTL formula* φ *when following the policy* σ *on the POMDP* $\mathcal{M}$*.*

Formal Problem Statement

In this section, we formulate the problem of task-guided inverse reinforcement learning (IRL) in POMDPs. Given a POMDP $\mathcal{M}$ with an *unknown* reward function $\mathcal{R}$, we seek to learn a reward function $\mathcal{R}$ along with an underlying policy σ that induces a behavior similar to the expert demonstrations.

In the remainder of the paper, we assume that the expert can have either full observability or partial observability. We define an expert trajectory on the POMDP $\mathcal{M}$ as the perceived observation and executed action sequence $\tau = \{(z_0, \alpha_0), (z_1, \alpha_1), \ldots, (z_T, \alpha_T)\}$, where $z_i \in \mathcal{Z}$ and $\alpha_i \in \mathcal{A}$ for all $i \in \{0, \ldots, T\}$, and T denotes the length of the trajectory. Similarly to Choi and Kim (2011), we assume given or we can construct from τ (Bayesian inference), the belief trajectory $b^\tau = \{b_0 := \mu_0, \ldots, b_T\}$, where $b_i(s)$ is the probability of being at state s at time index i. In the following, we assume that we are given a set of belief trajectories $\mathcal{D} = \{b^{\tau_1}, \ldots, b^{\tau_N}\}$ from trajectories $\tau_1, \ldots, \tau_N$, where N denotes the total number of underlying trajectories.

We build on the traditional encoding of the reward function as $\mathcal{R}(s,\alpha) := \sum_{k=1}^d \theta_k \phi_k(s,\alpha) = \theta^{\mathrm{T}} \phi(s,\alpha)$, where $\phi : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}^d$ is a known vector of basis functions with components referred to as *feature functions*, $\theta \in \mathbb{R}^d$ is an unknown weight vector characterizing the importance of each feature, and d is the number of features.

Specifically, we seek for a weight θ defining $\mathcal{R}$ and a policy σ such that its discounted feature expectation R^ϕ_σ matches an empirical discounted feature expectation $\bar{R}^\phi$ of the expert demonstration $\mathcal{D}$. That is, we have that $R^\phi_\sigma = \bar{R}^\phi$, where $R^\phi_\sigma := \sum_{t=0}^\infty \gamma^t \mathbb{E}_{S^\sigma_t, A^\sigma_t}[\phi(S^\sigma_t, A^\sigma_t)|\sigma]$ and the empirical mean $\bar{R}^\phi = \frac{1}{N} \sum_{b^\tau \in \mathcal{D}} \sum_{b_i \in b^\tau} \gamma^i \sum_{s \in \mathcal{S}} b_i(s) \phi(s, \alpha_i)$.

However, there may be infinitely many reward functions and policies that can satisfy the feature matching condition. Thus, to resolve the policy ambiguity, we seek for a policy σ that also maximizes the discounted causal entropy H^γ_σ.

Problem 1. *Given a reward-free POMDP* $\mathcal{M}$*, a demonstration set* $\mathcal{D}$*, and a feature* ϕ*, compute a policy* σ *and weight* θ

such that (a) $\sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{S_t^\sigma, A_t^\sigma}[\phi(S_t^\sigma, A_t^\sigma)|\sigma] = \bar{R}^\phi$*; (b) The causal entropy* H_σ^γ *is maximized by* σ.

Additionally, we seek to incorporate, if available, a priori high-level side information on the underlying expert task.

Problem 2. *Given a temporal logic formula* φ *and* $\lambda \geq 0$*, compute a policy* σ *and weight* θ *such that the constraints (a) and (b) in Problem 1 are satisfied, and* $\Pr_{\mathcal{M}}^\sigma(\varphi) \geq \lambda$.

We note that λ specifies the threshold of satisfaction of φ since binary constraint satisfaction does not make sense under partial observability and stochasticity in the model.

Nonconvex Formulation for IRL in POMDPs

In this section, we formulate the IRL problem as a nonconvex optimization problem. Then, we utilize a *Lagrangian relaxation* of the nonconvex problem as a part of our solution approach. We recall the reader (see Remark 1) that we compute M-FSC for POMDPs by computing memoryless policies σ on larger product POMDPs.

Substituting Visitation Counts. We eliminate the (infinite) time dependency in H_σ^γ and the feature matching constraint by a substitution of variables involving the policy-induced *discounted state visitation count* $\mu_\sigma^\gamma : \mathcal{S} \mapsto \mathbb{R}_+$ and *state-action visitation count* $\nu_\sigma^\gamma : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}_+$. For a policy σ, state s, and action α, the discounted visitation counts are defined by $\mu_\sigma^\gamma(s) := \mathbb{E}_{S_t}[\sum_{t=1}^{\infty} \gamma^t 1_{\{S_t=s\}}|\sigma]$ and $\nu_\sigma^\gamma(s,\alpha) := \mathbb{E}_{A_t,S_t}[\sum_{t=1}^{\infty} \gamma^t 1_{\{S_t=s,A_t=\alpha\}}|\sigma]$, where $1_{\{\cdot\}}$ is the indicator function. Further, $\nu_\sigma^\gamma(s,\alpha) = \pi_{s,\alpha}\mu_\sigma^\gamma(s)$, where $\pi_{s,\alpha} = \mathbb{P}[A_t = a|S_t = s]$ is a state-based policy.

We first provide a concave expression for the discounted causal entropy H_σ^γ as a function of μ_σ^γ and ν_σ^γ:

$$\begin{aligned} H_\sigma^\gamma &:= \sum\nolimits_{t=0}^{\infty} \gamma^t \mathbb{E}_{S_t^\sigma, A_t^\sigma}[-\log(\pi_{s_t,\alpha_t})] \\ &= \sum\nolimits_{t=0}^{\infty} \sum\nolimits_{(s,\alpha)\in\mathcal{S}\times\mathcal{A}} -(\log \pi_{s,\alpha})\pi_{s,\alpha}\gamma^t \mathbb{P}[S_t^\sigma = s] \\ &= \sum\nolimits_{(s,\alpha)\in\mathcal{S}\times\mathcal{A}} -(\log \pi_{s,\alpha})\pi_{s,\alpha}\mu_\sigma^\gamma(s) \\ &= \sum\nolimits_{(s,\alpha)\in\mathcal{S}\times\mathcal{A}} -\log \frac{\nu_\sigma^\gamma(s,\alpha)}{\mu_\sigma^\gamma(s)} \nu_\sigma^\gamma(s,\alpha), \end{aligned} \tag{1}$$

where the first equality is due to the definition of the discounted causal entropy H_σ^γ, the second equality obtained by expanding the expectation. The third and fourth equalities follow by the definition of the state visitation count μ_σ^γ, and the state-action visitation count ν_σ^γ. Next, we obtain a *linear* expression in ν_σ^γ for the discounted feature expectation R_ϕ^σ as:

$$\begin{aligned} R_\sigma^\phi &= \sum_{t=0}^{\infty} \sum_{(s,\alpha)\in\mathcal{S}\times\mathcal{A}} \phi(s,\alpha)\gamma^t \mathbb{P}[S_t^\sigma = s, A_t^\sigma = \alpha] \\ &= \sum_{(s,\alpha)\in\mathcal{S}\times\mathcal{A}} \phi(s,\alpha)\nu_\sigma^\gamma(s,\alpha), \end{aligned} \tag{2}$$

where the second equality is obtained by the definition of the visitation count ν_σ^γ. The following *nonconvex* constraint in $\mu_\sigma^\gamma(s)$ and $\sigma_{z,\alpha}$ ensures observation-based policies:

$$\nu_\sigma^\gamma(s,\alpha) = \mu_\sigma^\gamma(s) \sum\nolimits_{z\in\mathcal{Z}} \mathcal{O}(z|s)\sigma_{z,\alpha}. \tag{3}$$

Algorithm 1: Compute the weight vector θ and policy σ solution of the Lagrangian relaxation of the IRL problem.

Input: Feature expectation $\bar{R}^\phi$ from $\mathcal{D}$, initial weight θ^0, step size $\eta : \mathbb{N} \mapsto \mathbb{R}^+$, and (if available) a priori side information φ and $\lambda \in [0,1]$ imposing $\Pr_{\mathcal{M}}^\sigma(\varphi) \geq \lambda$.

1: $\sigma^0 \leftarrow$ uniform policy ▷ Initialize uniform policy
2: **for** $k = 1, 2, \ldots,$ **do** ▷ Compute θ via gradient descent
3: $\sigma^k \leftarrow \texttt{SCPForward}(\theta^k, \sigma^{k-1}, \varphi, \lambda)$ ▷ Solve the *forward problem* (5)–(7) with optional φ and λ
4: $\theta^{k+1} \leftarrow \theta^k - \eta(k)\big(R_{\sigma^k}^\phi - \bar{R}^\phi\big)$ ▷ Gradient step
5: **end for**
6: **return** σ^k, θ^k

Finally, the variables for the discounted visitation counts must satisfy the so-called *Bellman flow constraint* to ensure that the policy is well-defined. For each state $s \in \mathcal{S}$,

$$\mu_\sigma^\gamma(s) = \mu_0(s) + \gamma \sum_{s'\in\mathcal{S}} \sum_{\alpha\in\mathcal{A}} \mathcal{P}(s|s',\alpha)\nu_\sigma^\gamma(s',\alpha). \tag{4}$$

Lagrangian Relaxation of Feature Matching Constraint. Computing a policy σ that satisfies the feature matching constraint $R_\sigma^\phi = \bar{R}^\phi$ might be infeasible due to $\bar{R}^\phi$ being an empirical estimate from the finite set of demonstrations $\mathcal{D}$. Additionally, the feature matching constraint might also be infeasible due to the information asymmetry between the expert and the learner, e.g., the expert has full observation.

We build on a *Lagrangian relaxation* to incorporate the feature matching constraints into the objective of the forward problem, similar as other IRL algorithms in the literature. Specifically, we introduce $\theta \in \mathbb{R}^d$ as the dual variables of the relaxed problem. The desired weight vector θ and policy σ of Problem 1 and Problem 2 are the solutions of $\min_\theta f(\theta) := \max_\sigma H_\sigma^\gamma + \theta^{\mathrm{T}}(R_\sigma^\phi - \bar{R}^\phi)$. Algorithm 1 updates the reward weights by using gradient descent. To this end, the algorithm computes the gradient $\nabla f(\theta^k) = R_{\sigma^k}^\phi - \bar{R}^\phi$, where $\sigma^k = \arg\max_\sigma H_\sigma^\gamma + (\theta^k)^{\mathrm{T}}(R_\sigma^\phi - \bar{R}^\phi)$. In the following, we refer to the problem of computing such σ^k given θ^k as the *forward problem*, and we develop the algorithm `SCPForward`, presented in next section, to solve it in an efficient and scalable manner while incorporating high-level task specifications to guide the learning.

Nonconvex Formulation of the Forward Problem. Given a weight vector θ^k, we take advantage of the obtained substitution by the expected visitation counts to formulate the *forward problem* associated to Problem 1 as

$$\begin{aligned} \underset{\mu_\sigma^\gamma, \nu_\sigma^\gamma, \sigma}{\text{maximize}} \quad & \sum_{(s,\alpha)\in\mathcal{S}\times\mathcal{A}} -\log \frac{\nu_\sigma^\gamma(s,\alpha)}{\mu_\sigma^\gamma(s)} \nu_\sigma^\gamma(s,\alpha) \\ & + \sum_{(s,\alpha)\in\mathcal{S}\times\mathcal{A}} (\theta^k)^{\mathrm{T}} \phi(s,\alpha)\nu_\sigma^\gamma(s,\alpha), \end{aligned} \tag{5}$$

$$\text{subject to} \quad (3) - (4),$$

$$\forall (s,\alpha) \in \mathcal{S}\times\mathcal{A}, \;\; \mu_\sigma^\gamma(s) \geq 0, \;\; \nu_\sigma^\gamma(s,\alpha) \geq 0, \tag{6}$$

$$\forall (s,\alpha) \in \mathcal{S}\times\mathcal{A}, \;\; \mu_\sigma^\gamma(s) = \sum\nolimits_{\alpha\in\mathcal{A}} \nu_\sigma^\gamma(s,\alpha), \tag{7}$$

where the source of nonconvexity is from (3), and we remove the constant $-(\theta^k)^{\mathrm{T}}\bar{R}^{\phi}$ from the cost function.

Sequential Convex Programming Formulation

We develop `SCPForward`, adapting a sequential convex programming (SCP) scheme to efficiently solve the nonconvex *forward problem* (5)–(7). `SCPForward` involves a *verification step* to compute sound policies and visitation counts, which is not present in the existing SCP schemes. Additionally, we describe in the next section how to take advantage of high-level task specification (Problem 2) through slight modifications of the obtained optimization problem solved by `SCPForward`.

Linearizing Nonconvex Problem

`SCPForward` iteratively linearizes the nonconvex constraints in (3) around a previous solution. However, the linearization may result in an infeasible or unbounded linear subproblem (Mao et al. 2018). We first add *slack variables* to the linearized constraints to ensure feasibility. The linearized problem may not accurately approximate the nonconvex problem if the solutions to this problem deviate significantly from the previous solution. Thus, we utilize trust region constraints (Mao et al. 2018) to ensure that the linearization is accurate to the nonconvex problem. At each iteration, we introduce a *verification step* to ensure that the computed policy and visitation counts are not just approximations but actually satisfy the nonconvex policy constraint (3), improves the realized cost function over past iterations, and satisfy the temporal logic specifications, if available.

Linearizing Nonconvex Constraints and Adding Slack Variables. We linearize the nonconvex constraint (3), which is quadratic in $\mu_\sigma^\gamma(s)$ and $\sigma_{z,\alpha}$, around the previously computed solution denoted by $\hat{\sigma}$, $\mu_{\hat{\sigma}}^\gamma$, and $\nu_{\hat{\sigma}}^\gamma$. However, the linearized constraints may be infeasible. We alleviate this drawback by adding *slack variables* $k_{s,\alpha} \in \mathbb{R}$ for $(s,\alpha) \in \mathcal{S} \times \mathcal{A}$, which results in the affine constraint:

$$\nu_\sigma^\gamma(s,\alpha) + k_{s,\alpha} = \mu_{\hat{\sigma}}^\gamma(s) \sum\nolimits_{z\in\mathcal{Z}} \mathcal{O}(z|s)\sigma_{z,\alpha} + \big(\mu_\sigma^\gamma(s) - \mu_{\hat{\sigma}}^\gamma(s)\big) \sum\nolimits_{z\in\mathcal{Z}} \mathcal{O}(z|s)\hat{\sigma}_{z,\alpha}. \tag{8}$$

Trust Region Constraints. The linearization may be inaccurate if the solution deviates significantly from the previous solution. We add following *trust region* constraints to alleviate this drawback:

$$\forall (z,\alpha) \in \mathcal{Z}\times\mathcal{A}, \quad \hat{\sigma}_{z,\alpha}/\rho \le \sigma_{z,\alpha} \le \hat{\sigma}_{z,\alpha}\rho, \tag{9}$$

where ρ is the size of the trust region to restrict the set of allowed policies in the linearized problem. We augment the cost function in (5) with the term $-\beta\sum_{(s,\alpha)\in\mathcal{S}\times\mathcal{A}} k_{s,\alpha}$ to ensure that we minimize the violation of the linearized constraints, where β is a large positive constant.

Linearized Problem. Finally, by differentiating $x \mapsto x\log x$ and $y \mapsto x\log(x/y)$, we obtain the coefficients required to linearize the convex causal entropy cost function in (5). Thus, we obtain the following linear program (LP):

$$\begin{aligned}\underset{\mu_\sigma^\gamma,\nu_\sigma^\gamma,\sigma}{\text{maximize}} \; & \sum\nolimits_{(s,\alpha)\in\mathcal{S}\times\mathcal{A}} -\Bigg(\beta k_{s,\alpha} - \Big(\frac{\nu_{\hat{\sigma}}^\gamma(s,\alpha)}{\mu_{\hat{\sigma}}^\gamma(s)}\Big)\mu_\sigma^\gamma(s) \\ & + \Big(\log\frac{\nu_{\hat{\sigma}}^\gamma(s,\alpha)}{\mu_{\hat{\sigma}}^\gamma(s)} + 1\Big)\nu_\sigma^\gamma(s,\alpha)\Bigg) \\ & + \sum\nolimits_{(s,\alpha)\in\mathcal{S}\times\mathcal{A}} (\theta^k)^{\mathrm{T}}\phi(s,\alpha)\nu_\sigma^\gamma(s,\alpha) \qquad (10) \\ \text{subject to} \quad & (4), (6)-(9).\end{aligned}$$

Verification Step. After each iteration, the linearization might be inaccurate, i.e, the resulting policy $\tilde{\sigma}$ and *potentially inaccurate* visitation counts $\tilde{\nu}_{\tilde{\sigma}}^\gamma, \tilde{\mu}_{\tilde{\sigma}}^\gamma$ might not be feasible to the nonconvex policy constraint (3). As a consequence of the potential infeasibility, the currently attained (linearized) optimal cost might significantly differ from the *realized cost* by the feasible visiation counts for the $\tilde{\sigma}$. Additionally, existing SCP schemes linearizes the nonconvex problem around the previously inaccurate solutions for $\tilde{\nu}_{\tilde{\sigma}}^\gamma$, and $\tilde{\mu}_{\tilde{\sigma}}^\gamma$, further propagating the inaccuracy. The proposed *verification step* solves these issues. Given the computed policy $\tilde{\sigma}$, `SCPForward` computes the *unique and sound* solution for the visitation count $\mu_{\tilde{\sigma}}^\gamma$ by solving the corresponding *Bellman flow* constraints:

$$\mu_{\tilde{\sigma}}^\gamma(s) = \mu_0(s) + \gamma \sum_{s'\in\mathcal{S}}\sum_{\alpha\in\mathcal{A}} \mathcal{P}(s|s',\alpha)\mu_{\tilde{\sigma}}^\gamma(s') \sum_{z\in\mathcal{Z}} \mathcal{O}(z|s)\tilde{\sigma}_{z,\alpha}, \tag{11}$$

for all $s \in \mathcal{S}$, and where $\mu_{\tilde{\sigma}}^\gamma \ge 0$ is the only variable of the linear program. Then, `SCPForward` computes $\nu_{\tilde{\sigma}}^\gamma(s,\alpha) = \mu_{\tilde{\sigma}}^\gamma(s')\sum_{z\in\mathcal{Z}}\mathcal{O}(z|s)\tilde{\sigma}_{z,\alpha}$ and the *realized cost* cost at the current iteration is defined by

$$\begin{aligned}\mathrm{C}(\tilde{\sigma},\theta^k) = \; & \sum\nolimits_{(s,\alpha)\in\mathcal{S}\times\mathcal{A}} -\log\frac{\nu_{\tilde{\sigma}}^\gamma(s,\alpha)}{\mu_{\tilde{\sigma}}^\gamma}\nu_{\tilde{\sigma}}^\gamma(s,\alpha) \\ & + \sum\nolimits_{(s,\alpha)\in\mathcal{S}\times\mathcal{A}} (\theta^k)^{\mathrm{T}}\phi(s,\alpha)\nu_\sigma^\gamma(s,\alpha), \qquad (12)\end{aligned}$$

where we assume $0\log 0 = 0$. Finally, if the realized cost $\mathrm{C}(\tilde{\sigma},\theta^k)$ does not improve over the previous cost $\mathrm{C}(\hat{\sigma},\theta^k)$, the verification step rejects the obtained policy $\tilde{\sigma}$, contracts the trust region and `SCPForward` iterates with the previous solutions $\hat{\sigma}$, $\mu_{\hat{\sigma}}^\gamma$, and $\nu_{\hat{\sigma}}^\gamma$. Otherwise, the linearization is sufficiently accurate, the trust region is expanded, and `SCPForward` iterates with $\tilde{\sigma}$, $\mu_{\tilde{\sigma}}^\gamma$ and $\nu_{\tilde{\sigma}}^\gamma$. *By incorporating this verification step, we ensure that* `SCPForward` *always linearizes the nonconvex optimization problem around a solution that satisfies the nonconvex constraint* (3).

Incorporating High-Level Task Specifications

Given high-level side information on the agent tasks as the LTL formula φ, we first compute the product of the POMDP and the ω-automaton representing φ to find the set $\mathcal{T} \subseteq \mathcal{S}$ of states, called target or reach states, satisfying φ with probability 1 by using standard graph-based algorithms as a part of preprocessing step. We refer the reader to Baier and Katoen (2008) for a detailed introduction on how LTL specifications can be reduced to reachability specifications given by $\mathcal{T}$.

Algorithm 2: `SCPForward`: Linear programming-based algorithm to solve the *forward problem* (5)–(7), i.e., compute a policy σ^k that maximizes the causal entropy, enforces the feature matching constraint, and satisfies the specifications, if available.

Input: Current weight estimate θ^k, current best policy $\hat{\sigma}$, side information φ and λ, trust region $\rho > 1$, penalization coefficients β, $\beta^{\text{sp}} \geq 0$, constant ρ_0 to expand or contract trust region, and a threshold ρ_{lim} for trust region contraction.

1: Find $\mu^{\gamma}_{\hat{\sigma}}$ via linear constraint (11) and $\nu^{\gamma}_{\hat{\sigma}} = \mu^{\gamma}_{\hat{\sigma}}(s') \sum_{z \in \mathcal{Z}} \mathcal{O}(z|s)\hat{\sigma}_{z,\alpha}$, given $\hat{\sigma}$ ▷ Realized visitation counts

2: Find $\mu^{\text{sp}}_{\hat{\sigma}}$ via linear constraint (13) with $\nu^{\text{sp}}_{\hat{\sigma}} = \mu^{\text{sp}}_{\hat{\sigma}}(s') \sum_{z \in \mathcal{Z}} \mathcal{O}(z|s)\hat{\sigma}_{z,\alpha}$, given $\hat{\sigma}$ ▷ If φ is available

3: Compute the realized cost $\mathrm{C}(\hat{\sigma}, \theta^k) \leftarrow (12) + \mathrm{C}^{\text{sp}}_{\hat{\sigma}}$, given $\hat{\sigma}$ ▷ Add specifications' violation

4: **while** $\rho > \rho_{\text{lim}}$ **do** ▷ Trust region threshold

5: Find optimal $\tilde{\sigma}$ to the augmented LP (10) via $\hat{\sigma}$, $\mu^{\gamma}_{\hat{\sigma}}$, $\nu^{\gamma}_{\hat{\sigma}}$, $\mu^{\text{sp}}_{\hat{\sigma}}$, $\nu^{\text{sp}}_{\hat{\sigma}}$ ▷ We augment the LP with constraints (6), (7), (8), (13), and (spec) induced by $\mu^{\text{sp}}_{\sigma}, \nu^{\text{sp}}_{\sigma}$, and by adding $-\beta \sum_{(s,\alpha) \in \mathcal{S} \times \mathcal{A}} k^{\text{sp}}_{s,\alpha} - \beta^{\text{sp}}\Gamma^{\text{sp}}$ to the cost (10).

6: Compute the realized $\mu^{\gamma}_{\tilde{\sigma}}$, $\nu^{\gamma}_{\tilde{\sigma}}$,$\mu^{\text{sp}}_{\tilde{\sigma}}$, $\nu^{\text{sp}}_{\tilde{\sigma}}$, and $\mathrm{C}(\tilde{\sigma}, \theta^k)$ via $\tilde{\sigma}$ as in lines 1–3

7: $\{\hat{\sigma} \leftarrow \tilde{\sigma}; \rho \leftarrow \rho\rho_0\}$ if $\mathrm{C}(\tilde{\sigma}, \theta^k) \geq \mathrm{C}(\hat{\sigma}, \theta^k)$ else $\{\rho \leftarrow \rho/\rho_0\}$ ▷ Verification step

8: **end while**

9: **return** $\sigma^k := \hat{\sigma}$

As a consequence, the probability of satisfying φ is the sum of the probability of reaching the target states $s \in \mathcal{T}$, which are given by the *undiscounted state visitation count* μ^{sp}_{σ}. That is, $\Pr^{\sigma}_{\mathcal{M}}(\varphi) = \sum_{s \in \mathcal{T}} \mu^{\text{sp}}_{\sigma}(s)$. Unless $\gamma = 1$, $\mu^{\text{sp}}_{\sigma} \neq \mu^{\gamma}_{\sigma}$. Thus, we introduce new variables $\mu^{\text{sp}}_{\sigma}, \nu^{\text{sp}}_{\sigma}$, and the adequate constraints in the linearized problem (10).

Incorporating Undiscounted Visitation Variables to Linearized Problem. We append new constraints, similar to (6), (7), and (8), into the linearized problem (10), where the variables $\mu^{\gamma}_{\sigma}, \nu^{\gamma}_{\sigma}, k_{s,\alpha}, \mu^{\gamma}_{\hat{\sigma}}, \nu^{\gamma}_{\hat{\sigma}}$ are replaced by $\mu^{\text{sp}}_{\sigma}, \nu^{\text{sp}}_{\sigma}$, $k^{\text{sp}}_{s,\alpha}, \mu^{\text{sp}}_{\hat{\sigma}}, \nu^{\text{sp}}_{\hat{\sigma}}$, respectively. Further, we add the constraint

$$\mu^{\text{sp}}_{\sigma}(s) = \mu_0(s) + \sum_{s' \in \mathcal{S} \setminus \mathcal{T}} \sum_{\alpha \in \mathcal{A}} \mathcal{P}(s|s', \alpha)\nu^{\text{sp}}_{\sigma}(s', \alpha), \quad (13)$$

which is a modification of the *Bellman flow constraints* such that $\mu^{\text{sp}}_{\sigma}(s)$ for all $s \in \mathcal{T}$ only counts transitions from non-target states. Finally, we penalize the introduced slack variables for feasibility of the linearization by augmenting the cost function with the term $-\beta \sum_{(s,\alpha) \in \mathcal{S} \times \mathcal{A}} k^{\text{sp}}_{s,\alpha}$.

Relaxing Specification Constraints. We add the constraint $(\text{spec}) := \sum_{s \in \mathcal{T}} \mu^{\text{sp}}_{\sigma}(s) + \Gamma^{\text{sp}} \geq \lambda$ to the linearized problem, where $\Gamma^{\text{sp}} \geq 0$ is a slack variable ensuring the linearized problem is always feasible. We augment the cost function with $-\beta^{\text{sp}}\Gamma^{\text{sp}}$ to penalize violating φ, where β^{sp} is a hyperparameter positive constant .

Updating Verification Step. We modify the previously-introduced realized cost $\mathrm{C}(\tilde{\sigma}, \theta^k)$ to penalize if the obtained policy does not satisfy the specification φ. This cost also accounts for the linearization inaccuracy of the new policy constraint due to σ, μ^{sp}_{σ}, and ν^{sp}_{σ}. At each iteration, `SCPForward` computes the accurate $\mu^{\text{sp}}_{\tilde{\sigma}}$ of current policy $\tilde{\sigma}$ through solving a feasibility LP with constraints given by the *modified Bellman flow constraints* (13). Then, it augments $\mathrm{C}^{\text{sp}}_{\tilde{\sigma}} = \min\{0, (\sum_{s \in \mathcal{T}} \mu^{\text{sp}}_{\tilde{\sigma}}(s) - \lambda)\beta^{\text{sp}}\}$ to the realized cost to take the specification constraints into account.

Numerical Experiments

We evaluate the proposed IRL algorithm on several POMDP instances, from Junges, Jansen, and Seshia (2021). We first compare our IRL algorithm with a straightforward variant of GAIL (Ho and Ermon 2016) adapted for POMDPs. Then, we provide some results on the data-efficiency of the approach when taking advantage of side information. Finally, we demonstrate the scalability of the routine `SCPForward` for solving the *forward* problem through comparisons with state-of-the-art solvers such as `SolvePOMDP` (Walraven and Spaan 2017), `SARSOP` (Kurniawati, Hsu, and Lee 2008), `PRISM-POMDP` (Norman, Parker, and Zou 2017). We consider throughout this section the hyperparameters $\beta = 1e^3$, $\beta^{\text{sp}} = 10$, $\rho = 1.01$, $\rho_0 = 1.5$, $\rho_{\text{lim}} = 1e^{-4}$, and $\gamma = 0.999$. Besides, we provide in Djeumou et al. (2021) additional details on the key results of this paper and the experiments, e.g., preprocessing steps such as the product POMDP with M-FSC or computing reachability specifications from LTL specifications.

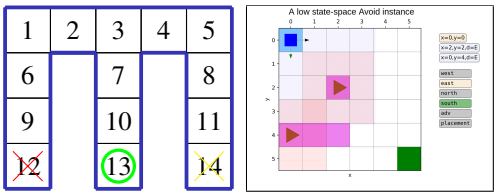

Figure 1: Some examples from the benchmark set. From left to right, we have the *Maze* and *Avoid*, respectively.

Benchmark Set. The POMDP instances are as follows. *Evade* is a turn-based game where the agent must reach a destination without being intercepted by a faster player. In *Avoid*, the agent must avoid being detected by two other moving players following certain preset, yet unknown routes. In *Intercept*, the agent must intercept another player who is trying to exit a gridworld. In *Rocks*, the agents must sample at least one good rock over the several rocks without any failures. Finally, in *Maze*, the agent must exit a maze as fast as possible while avoiding trap states.

Variants of Learned Policies and Experts. We refer to four types of policies. The type of policy depends on whether it uses side information from a temporal specifica-

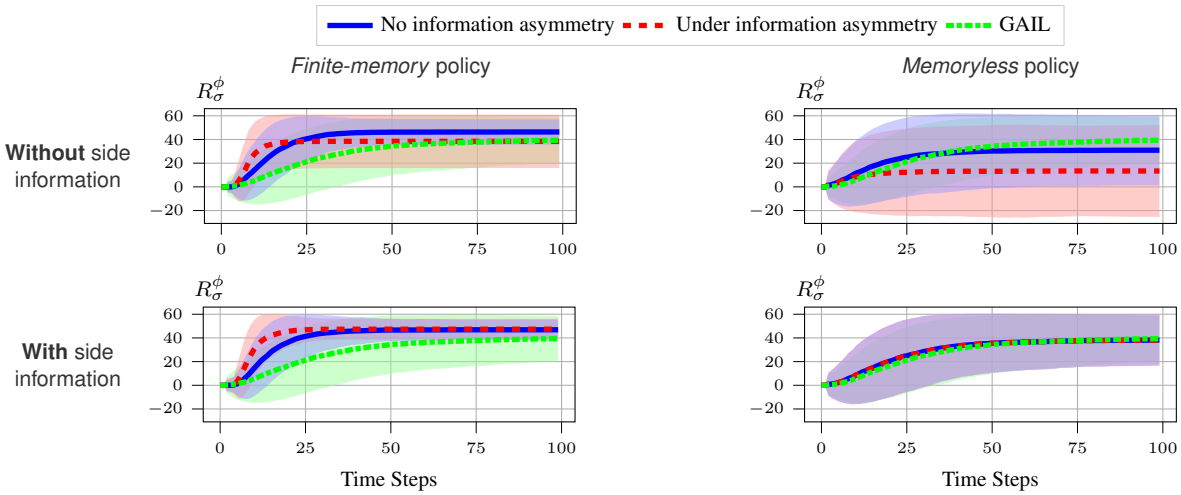

Figure 2: Representative results on the Maze example showing the reward of the policies under the true reward function (R_σ^ϕ) versus the time steps. Compare the two rows: The policies in the top row that do not utilize side information suffer a performance drop under information asymmetry. On the other hand, in the bottom row, the performance of policies incorporating side information into learning does not decrease under information asymmetry. Compare the two columns: The performance of the finite-memory policies in the left column is significantly better than memoryless policies. Except for the memoryless policies without side information, our algorithm outperforms GAIL. The expert reward on the MDP is 48.22, while 47.83 on POMDP.

tion φ or not, and whether it uses a memory size $\mathrm{M} = 1$ or $\mathrm{M} = 10$. We also consider two types of experts. The first expert has full information about the environment and computes an optimal policy in the underlying MDP. The second expert has partial observation and computes a locally optimal policy in the POMDP with a memory size of $\mathrm{M} = 15$. Recall that the agent always has partial information. Therefore, the first type of expert corresponds to having information asymmetry between the learning agent and expert. *Besides, we consider as a baseline a variant of GAIL where we learn the policy on the MDP without side information, and extend it to POMDPs via an offline computation of the belief in the states. Doing so provides a significant advantage to GAIL since we learn on the MDP. We do not compare with Choi and Kim (2011) as explained in the related work.*

We discuss the effect of side information and memory in the policies. While we detail only on the *Maze* example, where the agent must exit a maze as fast as possible, we observe similar patterns for other examples. We give detailed results for the other examples in Djeumou et al. (2021).

Maze Example

The POMDP $\mathcal{M}$ is specified by $\mathcal{S} = \{s_1, \ldots, s_{14}\}$ corresponding to the cell labels in Figure 1. An agent in the maze only observes whether or not there is a wall (in blue) in a neighboring cell. That is, the set of observations is $\mathcal{O} = \{o_1, \ldots, o_6, o_7\}$. For example, o_1 corresponds to observing west and north walls (s_1), o_2 to north and south walls (s_2, s_4), and o_5 to east and west walls ($s_6, s_7, s_8, s_9, s_{10}, s_{11}$). The observations o_6 and o_7 denote the target state (s_{13}) and bad states(s_{12}, s_{14}). The transition model is stochastic with a probability of slipping $p = 0.1$. Further, the states s_{13} and s_{14} lead to the end of the simulation (trapping states).

In the IRL experiments, we consider three feature functions. We penalize taking more steps with $\phi^{\text{time}}(s, \alpha) = -1$ for all s, α. We provide a positive reward when reaching s_{13} with $\phi^{\text{target}}(s, \alpha) = 1$ if $s = s_{13}$ and $\phi^{\text{target}}(s, \alpha) = 0$ otherwise. We penalize bad states s_{12} and s_{14} with $\phi^{\text{bad}}(s, \alpha) = -1$ if $s = s_{12}$ or $s = s_{14}$, and $\phi^{\text{bad}}(s, \alpha) = 0$ otherwise. *Finally, we have the LTL formula* $\varphi = \boldsymbol{G} \neg \text{bad}$ *as the task specification, where* bad *is an atomic proposition that is true if the current state* $s = s_{12}$ *or* $s = s_{14}$. *We constrain the learned policy to satisfy* $\Pr^{\sigma}_{\mathcal{M}}(\boldsymbol{G} \neg \text{bad}) \geq 0.9$.

Side Information Alleviates the Information Asymmetry. Figure 2 shows that if there is an information asymmetry between the learning agent and the expert, the policies that do not utilize side information suffer a significant performance drop. The policies that do not incorporate side information into learning obtain a lower performance by 57% under information asymmetry, as shown in the top row of Figure 2. On the other hand, as seen in the bottom row of Figure 2, the performance of the policies that use side information is almost unaffected by the information asymmetry.

Memory Leads to More Performant Policies. The results in Figure 2 demonstrate that incorporating memory into the policies improves the performance, i.e., the attained reward, in all examples, both in solving the forward problem and learning policies from expert demonstrations. Incorporating memory partially alleviates the effects of information asymmetry, as the performance of the finite-memory policy decreases by 18% under information asymmetry as opposed to 57% for the memoryless policy.

We see that in Table 1, incorporating memory into policy on the Maze and Rocks benchmarks, allows `SCPForward` to compute policies that are almost optimal, evidenced by

Problem	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{S}\times\mathcal{O}\rvert$	$\lvert\mathcal{O}\rvert$	SCPForward R^{ϕ}_{σ}	SCPForward Time (s)	SARSOP R^{ϕ}_{σ}	SARSOP Time (s)	SolvePOMDP R^{ϕ}_{σ}	SolvePOMDP Time (s)
Maze	17	162	11	39.24	**0.1**	**47.83**	0.24	47.83	0.33
Maze(10-FSC)	161	2891	101	46.32	2.04	NA	NA	NA	NA
Rock	550	4643	67	19.68	12.2	**19.83**	**0.05**	–	–
Rock(5-FSC)	2746	41759	331	19.82	97.84	NA	NA	NA	NA
Intercept	1321	5021	1025	**19.83**	**10.28**	**19.83**	13.71	–	–
Intercept	1321	7041	1025	**19.81**	**13.18**	**19.81**	81.19	–	–
Evade	2081	16761	1089	**96.79**	**26.25**	95.28	3600	–	–
Evade	36361	341121	18383	**94.97**	**3600**	–	–	–	–
Avoid	2241	8833	1956	**9.86**	**14.63**	**9.86**	210.47	–	–
Avoid	19797	62133	3164	**9.72**	**3503**	–	–	–	–

Table 1: Results for the benchmarks. On larger benchmarks (e.g., Evade and Avoid), the method we developed can compute a locally optimal policy. We set the time-out to 3600 seconds. An empty cell (denoted by $-$) represents the solver failed to compute any policy before the time-out, while NA refers to not applicable due to the approach being based on belief updates.

obtaining almost the same reward as the solver SARSOP.

Side Information Improves Data Efficiency and Performance. Figure 3 shows that even on a low data regime, learning with task specifications achieves significantly better performance than without the task specifications.

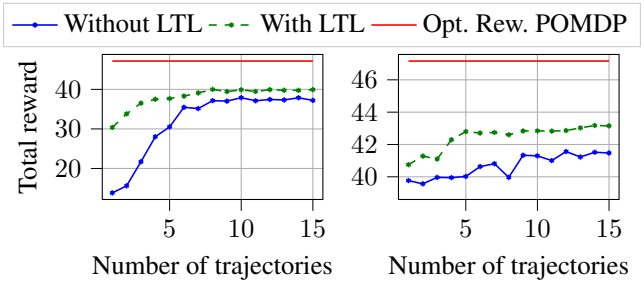

Figure 3: We show the data efficiency of the proposed approach through the total reward obtained by the learned policies as a function of the number of expert demonstrations (No information asymmetry). The figure on the left shows the performance of learning memoryless policies, while the figure on the right shows the performance of a 5-FSC.

Besides, in a more complicated environment such as Avoid, Figure 4 shows that task specifications are crucial to hope even to learn the task. We refer the reader to Djeumou et al. (2021) for the details of the experiment.

SCPForward Yields Better Scalability

We highlight three observations regarding the scalability of SCPForward. First, the results in Table 1 show that only SARSOP is competitive with SCPForward on larger POMDPs. SolvePOMDP runs out of time in all but the smallest benchmarks, and PrismPOMDP runs out of memory in all benchmarks. Most of these approaches are based on updating a belief over the states, which for a large state space can become extremely computationally expensive.

Second, in the benchmarks with smaller state spaces, e.g., *Maze* and *Rock*, SARSOP can compute policies that yield

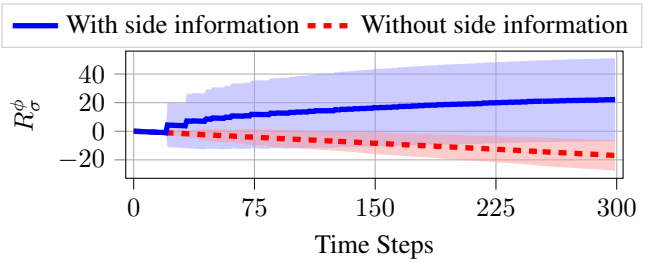

Figure 4: Results on the Avoid example show that side information can help to crucially improve the performance.

better performance in less time. This is due to the efficiency of belief-based approaches on small-size problems. On the other hand, SARSOP does not scale to larger POMDPs with a larger number of states and observations. For example, by increasing the number of transitions in *Intercept* benchmark from 5021 to 7041, the computation time for SARSOP increases by 516%. On the other hand, the increase of the computation time of SCPForward is only 28%.

Third, on the largest benchmarks, including tens of thousands of states and observations, SARSOP fails to compute any policy before time-out, while SCPForward found a solution. Finally, we also note that SCPForward can also compute a policy that maximizes the causal entropy and satisfies an LTL specification, unlike SARSOP.

Conclusion

We develop an algorithm for inverse reinforcement learning under partial observation. The algorithm assumes known transition and observation functions of the POMDP. We empirically demonstrate that by incorporating task specifications into the learning process, we can alleviate the information asymmetry between the expert and the agent and learn policies that yield similar performance to the expert. Further, we show that our routine SCPForward to solve the forward problem outperforms state-of-the-art POMDP solvers on instances with a large number of states and observations.

Acknowledgments

This work was supported in part by ARL W911NF2020132 and NSF 1652113.

References

Abbeel, P.; Coates, A.; and Ng, A. Y. 2010. Autonomous Helicopter Aerobatics Through Apprenticeship Learning. *The International Journal of Robotics Research*, 29(13): 1608–1639.

Abbeel, P.; and Ng, A. Y. 2004. Apprenticeship Learning via Inverse Reinforcement Learning. In *Proceedings of the 21st International Conference on Machine Learning*.

Akash, K.; Polson, K.; Reid, T.; and Jain, N. 2019. Improving Human-Machine Collaboration Through Transparency-based Feedback–Part I: Human Trust and Workload Model. *IFAC-PapersOnLine*, 51(34): 315–321.

Amato, C.; Bernstein, D. S.; and Zilberstein, S. 2010. Optimizing Fixed-Size Stochastic Controllers for POMDPs and Decentralized POMDPs. *AAMAS*, 21: 293–320.

Bai, H.; Hsu, D.; and Lee, W. S. 2014. Integrated Perception and Planning in the Continuous Space: A POMDP Approach. *The International Journal of Robotics Research*, 33: 1288–1302.

Baier, C.; and Katoen, J.-P. 2008. *Principles of Model Checking*. The MIT Press.

Bogert, K. D.; and Doshi, P. 2014. Multi-robot inverse reinforcement learning under occlusion with interactions. In *AAMAS*.

Boularias, A.; Krömer, O.; and Peters, J. 2012. Structured Apprenticeship Learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 227–242.

Choi, J.; and Kim, K.-E. 2011. Inverse Reinforcement Learning in Partially Observable Environments. *Journal of Machine Learning Research*, 12: 691–730.

Djeumou, F.; Cubuktepe, M.; Lennon, C. T.; and Topcu, U. 2021. Task-Guided Inverse Reinforcement Learning Under Partial Information. *ArXiv*, abs/2105.14073.

Dragan, A. D.; and Srinivasa, S. S. 2013. A Policy-Blending Formalism for Shared Control. *The International Journal of Robotics Research*, 32(7): 790–805.

Finn, C.; Levine, S.; and Abbeel, P. 2016. Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization. In *International conference on machine learning*, 49–58.

Hadfield-Menell, D.; Russell, S. J.; Abbeel, P.; and Dragan, A. D. 2016. Cooperative Inverse Reinforcement Learning. In *NIPS*.

Ho, J.; and Ermon, S. 2016. Generative Adversarial Imitation Learning. In *NIPS*.

Junges, S.; Jansen, N.; and Seshia, S. A. 2021. Enforcing Almost-Sure Reachability in POMDPs. In *CAV*.

Junges, S.; Jansen, N.; Wimmer, R.; Quatmann, T.; Winterer, L.; Katoen, J.-P.; and Becker, B. 2018. Finite-State Controllers of POMDPs using Parameter Synthesis. In *UAI*.

Kitani, K. M.; Ziebart, B. D.; Bagnell, J. A.; and Hebert, M. 2012. Activity Forecasting. In *European Conference on Computer Vision*, 201–214.

Kurniawati, H.; Hsu, D.; and Lee, W. S. 2008. Sarsop: Efficient Point-Based POMDP Planning by Approximating Optimally Reachable Belief Spaces. In *Robotics: Science and systems*.

Littman, M. L.; Topcu, U.; Fu, J.; Isbell, C. L.; Wen, M.; and MacGlashan, J. 2017. Environment-Independent Task Specifications via GLTL. *ArXiv*, abs/1704.04341.

Liu, X.; and Datta, A. 2012. Modeling Context Aware Dynamic Trust Using Hidden Markov Model. In *AAAI*.

Madani, O.; Hanks, S.; and Condon, A. 1999. On the Undecidability of Probabilistic Planning and Infinite-Horizon Partially Observable Markov Decision Problems. In *AAAI*.

Mao, Y.; Szmuk, M.; Xu, X.; and Açikmese, B. 2018. Successive Convexification: A Superlinearly Convergent Algorithm for Non-convex Optimal Control Problems. *arXiv: Optimization and Control*, abs/1804.06539.

Memarian, F.; Xu, Z.; Wu, B.; Wen, M.; and Topcu, U. 2020. Active Task-Inference-Guided Deep Inverse Reinforcement Learning. In *2020 59th IEEE CDC*, 1932–1938.

Meuleau, N.; Kim, K.-E.; Kaelbling, L. P.; and Cassandra, A. R. 1999. Solving POMDPs by searching the space of finite policies. In *UAI*, 417–426.

Ng, A. Y.; Russell, S. J.; et al. 2000. Algorithms for Inverse Reinforcement Learning. In *Proceedings of the 17th International Conference on Machine Learning*.

Norman, G.; Parker, D.; and Zou, X. 2017. Verification and Control of Partially Observable Probabilistic Systems. *Real-Time Systems*, 53: 354–402.

Ong, S. C.; Png, S. W.; Hsu, D.; and Lee, W. S. 2009. POMDPs for Robotic Tasks with Mixed Observability. In *Robotics: Science and Systems*, volume 5.

Osa, T.; Pajarinen, J.; Neumann, G.; Bagnell, J.; Abbeel, P.; and Peters, J. 2018. An Algorithmic Perspective on Imitation Learning. *Foundations and Trends in Robotics*, 7: 1–179.

Papusha, I.; Wen, M.; and Topcu, U. 2018. Inverse Optimal Control with Regular Language Specifications. In *2018 Annual American Control Conference (ACC)*, 770–777.

Pnueli, A. 1977. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*, 46–57.

Ramachandran, D.; and Amir, E. 2007. Bayesian Inverse Reinforcement Learning. In *IJCAI*, volume 7, 2586–2591.

Ratliff, N. D.; Bagnell, J. A.; and Zinkevich, M. A. 2006. Maximum Margin Planning. In *Proceedings of the 23rd International Conference on Machine Learning*, 729–736.

Walraven, E.; and Spaan, M. 2017. Accelerated Vector Pruning for Optimal POMDP Solvers. In *AAAI*.

Wen, M.; Papusha, I.; and Topcu, U. 2017. Learning from Demonstrations with High-Level Side Information. In *IJCAI*.

Yu, H.; and Bertsekas, D. P. 2008. On Near Optimality of the Set of Finite-State Controllers for Average Cost POMDP. *Mathematics of Operations Research*, 33: 1–11.

Yuan, Y.-x. 2015. Recent Advances in Trust Region Algorithms. *Mathematical Programming*, 151(1): 249–281.

Zhang, S.; Sinapov, J.; Wei, S.; and Stone, P. 2017. Robot Behavioral Exploration and Multimodal Perception using POMDPs. In *AAAI Spring Symposium*.

Zhou, Z.; Bloem, M.; and Bambos, N. 2017. Infinite Time Horizon Maximum Causal Entropy Inverse Reinforcement Learning. *IEEE Transactions on Automatic Control*, 63: 2787–2802.

Ziebart, B. D.; Bagnell, J. A.; and Dey, A. K. 2010. Modeling Interaction via the Principle of Maximum Causal Entropy. In *ICML*.

Ziebart, B. D.; Bagnell, J. A.; and Dey, A. K. 2013. The Principle of Maximum Causal Entropy for Estimating Interacting Processes. *IEEE Transactions on Information Theory*, 59: 1966–1980.

Ziebart, B. D.; Maas, A. L.; Bagnell, J. A.; and Dey, A. K. 2008. Maximum Entropy Inverse Reinforcement Learning. In *AAAI*.