24th European Conference on Artificial Intelligence - ECAI 2020
Santiago de Compostela, Spain

Accounting for Observer's Partial Observability in Stochastic Goal Recognition Design

Messing with the Marauder's Map

Christabel Wayllace[1], Sarah Keren[2], Avigdor Gal[3], Erez Karpas[4], William Yeoh[5], Shlomo Zilberstein[6]

Abstract. Motivated by security applications, where agent intentions are unknown, actions may have stochastic outcomes, and an observer may have an obfuscated view due to low sensor resolution, we introduce partially-observable states and unobservable actions into a stochastic goal recognition design framework. The proposed model is accompanied by a method for calculating the expected maximal number of steps before the goal of an agent is revealed and a new sensor refinement modification that can be applied to enhance goal recognition. A preliminary empirical evaluation on a range of benchmark applications shows the effectiveness of our approach.

1 Introduction

Goal recognition design (GRD) [7] is an offline task of redesigning (either physical or virtual) environments, with the aim of efficient online goal recognition [3, 13, 16]. In the past few years, researchers have investigated the GRD problem under a varying sets of assumptions [8, 9, 10, 19, 18].

The ability to recognize an agent's goal depends to a large extent, on the ability of an observer to monitor agent behavior. Motivated, among other needs, by security applications such as airport monitoring, where agent intended movements are unknown to an observer and its current location is partially impaired due to low sensor (*e.g.*, GPS) resolution. Therefore, we envision an environment where, while an agent is fully informed, an observer has access only to a partially-observable environment where several nearby agent states may be indistinguishable. In such a setting, changes in observations are the only indication of activities performed by an agent.

In this work, we extend the GRD's state-of-the-art to account for partial observability (of an observer) in stochastic environments. The new model, which we call *Partially-Observable Stochastic GRD* (POS-GRD), assumes that agent's actions are no longer observable and agent's states are partially observable. Moreover, agent activities may have a stochastic outcome. For example, an agent may attempt to pass through doors that are locked at times.

Example 1. *To illustrate the setting of this work, we present an example of a typical real-world security monitoring applications, positioned in the wizarding world of Harry Potter, where magic is the latest technology. Potter is back at the Hogwarts School of Witchcraft and Wizardry. He is tasked with establishing a security system that can detect as early as possible anyone who enters the school from the main entrance, heading towards Professor McGonagall's office. Upon entering the building, a stochastic staircase chamber is used. Therefore, a person aiming to move to a certain part of the school may find herself at a different location. For example, when heading to the dining hall one may find herself at the hallway. Figure 1(bottom) provides a partial view of Hogwarts, illustrating the example by depicting locations as nodes and transitions (and their probabilities) as edges.*

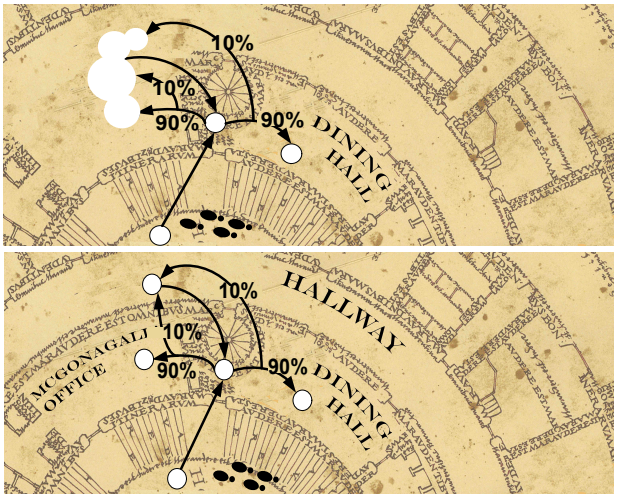


Figure 1. Marauder's Map Before (top) and After (bottom) Potter's Modification Spell (partial view)

A GRD task is constructed of two subtasks, namely (1) analyzing a goal recognition environment using an efficacy measure and (2) improving an environment by applying a set of design modifications. Two efficacy measures were introduced in the literature. *Worst case distinctiveness* (*wcd*) [7] captures the maximum number of steps an agent can take without revealing its goal. *Expected case distinctiveness* (*ecd*) [18] weighs the possible goals based on their likelihood of being the true goal. As for the second subtask, multiple redesign methods were proposed, including action removal and sensor refinement [10], which decreases the degree of observation uncertainty on tokens produced by agent actions.

Example 1. (cont.) *To illustrate environment improvement, consider*

[1] Washington University in St. Louis, USA, email: cwayllace@wustl.edu
[2] Center for Research on Computation and Society, Harvard University, USA, email: skeren@seas.harvard.edu
[3] Technion – Israel Institute of Technology, Israel, email: avigal@technion.ac.il
[4] Technion – Israel Institute of Technology, Israel, email: karpase@technion.ac.il
[5] Washington University in St. Louis, USA, email: wyeoh@wustl.edu
[6] University of Massachusetts, Amherst, USA, email: shlomo@cs.umass.edu

once more our security expert, Harry Potter, who intends to use a magical artifact called the Marauder's Map that, just like a real time Location System, reveals the whereabout of all witches and wizards at Hogwarts. The map can show where witches and wizards are, but due to some dark magic (that muggles sometimes tend to associate with poor system maintenance) its resolution has been reduced and some places, like the hallway and Professor McGonagall's office are no longer distinguishable (Figure 1(top)).

Potter can cast exactly one *modification spell to improve the resolution of some part of the map. Knowing the stochastic nature of the stairs, Potter realizes that his best choice is to cast the spell to create separate observations to the hallway and Professor McGonagall's office (Figure 1(bottom)). This will guarantee that anyone ending up at the hallway and heads back to the entrance has the intention of reaching Professor McGonagall's office, and such a recognition can occur after that person takes at most two actions.*

In this work we offer solutions to both subtasks in the new problem setting of POS-GRD. First, we make use of *Markov Decision Processes* (MDPs), augmented with goal information, to compute efficiently *wcd* in a partially-observable stochastic environment. Then, we introduce a new environment modification that refines sensors over states (rather than actions), as an efficient environment redesign mechanism, in addition to the use of action removal modification, which has been introduced for several GRD models before.

The contribution of this work is threefold. First, we present in Section 3 a model of a new variant of the GRD problem that was not considered before. POS-GRD supports decision making in environments where observers have only partial observability and actions may have stochastic outcomes. This setting removes some restrictions that exist in the literature on the nature of observability in an environment and provides a framework that is coupled with many realistic assumptions. The model is given in a general form that does not require agent optimality. Our second contribution (presented in Section 4) involves a novel approach of integrating partial observability into an augmented MDP, proposed by Wayllace *et al.* [18], to compute *wcd* efficiently. The use of augmented MDPs is tailored for agents that use optimal policies. Finally, our third contribution (also presented in Section 4) involves a new effective environment modification (sensor refinement over states) to reduce *wcd*.

Our preliminary empirical evaluation (Section 5) shows promise when it comes to the effectiveness of the proposed sensor refinement modification in reducing *wcd* in partially-observable stochastic settings.

Section 2 introduces the principles of a stochastic short path markov decision process, Section 6 discusses related work and Section 7 concludes the paper.

2 Background

A *Stochastic Shortest Path Markov Decision Process* (SSP-MDP) [11] is represented as a tuple $\langle \mathbf{S}, s_0, \mathbf{A}, \mathbf{T}, \mathbf{C}, g \rangle$. It consists of a set of states $\mathbf{S}$; a start state $s_0 \in \mathbf{S}$; a set of actions $\mathbf{A}$; a transition function $\mathbf{T} : \mathbf{S} \times \mathbf{A} \times \mathbf{S} \to [0, 1]$ that gives the probability $T(s, a, s')$ of transitioning from state s to s' when action a is executed; a cost function $\mathbf{C} : \mathbf{S} \times \mathbf{A} \times \mathbf{S} \to \mathbb{R}$ that gives the cost $C(s, a, s')$ of executing action a in state s and arriving in state s'; and a set of goal states $g \subseteq \mathbf{S}$. The goal states are terminal, that is, $T(s, a, s) = 1$ and $C(s, a, s) = 0$ for all goal states $s \in g$ and actions $a \in \mathbf{A}$.

An SSP-MDP (MDP hereinafter) must also satisfy the following two conditions: (1) *proper policy* existence: there is a mapping from states to actions with which an agent can reach a goal state from any state with probability 1; (2) *improper policy* cost: improper policies incur a cost of ∞ at states from which a goal cannot be reached with probability 1.

Solving an MDP involves finding an optimal policy π^*, *i.e.*, a mapping of states to actions, with the smallest expected cost. We use the term *optimal actions* to refer to actions in an optimal policy. While a policy maps every state to an action, a *partial policy* maps a *subset* of states to an action — that is, a partial policy $\pi : \mathbf{S} \to \mathbf{A} \cup \{\perp\}$ maps each state to either an action or to $\perp$, denoting it is undefined for that state. In what follows, when referring to partial policies, we shall assume proper policies only. We denote the set of states on which a partial policy $\hat{\pi}$ is defined by $S_{\hat{\pi}} := \{s \in \mathbf{S} \mid \hat{\pi}(s) \neq \perp\}$.

Given a policy π and a starting state s_0, $V_\pi(s_0) = \sum_{s' \in \mathbf{S}} T(s, \pi(s), s') \big[C(s, a, s') + V_\pi(s')\big]$ is the expected cost of following policy π. The expected cost of an optimal policy π^* for the starting state $s_0 \in \mathbf{S}$ is the expected cost $V(s_0)$, and the expected cost $V(s)$ for all states $s \in \mathbf{S}$ is calculated using the Bellman equation [2], choosing for each state s the action that minimizes $V(s)$:

$$V(s) = \min_{a \in \mathbf{A}} \sum_{s' \in \mathbf{S}} T(s, a, s') \big[C(s, a, s') + V(s')\big] \qquad (1)$$

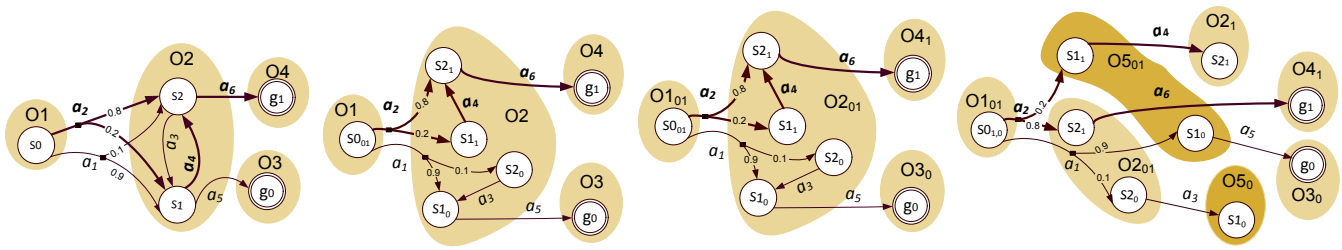
Value Iteration (VI) [2] is a fundamental algorithm for solving an MDP using an expected cost value function V. Methods such as *Topological VI* (TVI) [4] guarantee efficient execution of the VI algorithm.

3 Partially-Observable Stochastic GRD

We next define the *Partially-Observable Stochastic GRD* (POS-GRD) problem, where (1) actions are non-observable; and (2) states are partially observable, so that several states may be indistinguishable from one another. The degree of observation uncertainty is related to the resolution of sensors in the problem.

Due to low sensor resolution, more than one state can be mapped to the same observation. We demonstrated it in Example 1, where the hallway and Professor McGonagall's office are no longer distinguishable (Figure 1(top)). Figure 2(a) illustrates a more elaborated example with states (annotated nodes), actions (annotated edges), and observations (annotated shaded areas). Goals are marked with double-lined circles. Each edge is labeled with an action name, and an action with a stochastic outcome is represented by a multi-head arrow, with probabilities associated with each arrow head. We also mark the intended goal of an action using arrow width, where thin arrows represent intended goal g_0 and thick arrows represent intended goal g_1. The observation model we propose is different from others such as HMM or POMDP as we assume actions are not observable *at all*. The observer only observes a transition between two states that are associated with different observations. Any transition between states that are mapped to the same observation is undetectable. This is suitable for sensors that produce a continuous reading of the observed state and settings in which the agent can spend an arbitrary amount of time in each state.

We model a POS-GRD problem using two components, namely a *goal recognition model* and a *design model*. For the latter, modifications create a new goal recognition model from an existing one. We formulate each component separately before introducing the POS-GRD problem.



(a) Original MDP (b) Cycle-Free MDP (c) Augmented MDP (expected *wcd* = 2.2) (d) After Sensor Refinement (expected *wcd* = 2.0)

Figure 2. A Running Use-Case

3.1 Partially-Observable Stochastic Goal Recognition

Definition 1. *A* partially-observable goal recognition model with stochastic action outcomes *is a tuple* $PO = \langle M, \mathbf{G}, N \rangle$ *where*

- M *is an MDP with* positive *costs* $\mathbf{C}$ *and no goal;*
- $\mathbf{G}$ *is a set of possible goals, that is for each* $g \in \mathbf{G}$, $g \subseteq \mathbf{S}$ *is a possible goal of* M*; and*
- N *is a sensor model. Each state* s *is associated with an observation* $N(s)$*, which we refer to as the* projected observation *of* s*. The set* $\mathbf{S}$ *is partitioned into observation sets* $\mathbf{O}_1, ..., \mathbf{O}_n$ *such that* $\forall s, s' : N(s) = N(s')$ *iff* $\exists i : s, s' \in \mathbf{O}_i$.

Definition 1 generalizes the stochastic goal recognition model [19] by including a sensor model N that defines the degree of partial observability of states. To illustrate the model, consider Figure 2(a), where $\mathbf{G} = \{g_0, g_1\}$, $O2 = \{S1, S2\}$, and $N(S1) = N(S2)$.

Given a partially-observable goal recognition model with stochastic action outcomes (Definition 1), we next define expected *wcd*, starting with partial policy containment.

Definition 2. *A partial policy* $\hat{\pi}$ *is* contained *in a partial policy* $\hat{\pi}'$ *(marked* $\hat{\pi} \subseteq \hat{\pi}'$*) if* $S_{\hat{\pi}} \subseteq S_{\hat{\pi}'}$ *and* $\forall s \in S_{\hat{\pi}}$, $\hat{\pi}(s) = \hat{\pi}'(s)$.

To define the relationship between a partial policy and a possible goal, we denote the set of all *legal* policies for a goal g by $\Pi_{leg}(g)$. For optimal agents, $\Pi_{leg}(g) = \Pi_{opt}(g)$ is the set of all optimal policies with respect to goal g.

Definition 3. *A partial policy* $\hat{\pi}$ satisfies *a goal* g *if* $\exists \pi \in \Pi_{leg}(g)$ *s.t.* $\hat{\pi} \subseteq \pi$*. The set of goals satisfied by a partial policy* $\hat{\pi}$ *is marked by* $G(\hat{\pi})$.

A *trajectory* $\vec{\tau} = \langle s_0, a_1, s_1, \ldots, a_n, s_n \rangle$ is a realization of an agent's policy, denoted by alternating actions an agent performs and states reached by an agent. We use trajectory indices to relate an action with its resulting state. A trajectory $\vec{\tau}$ is *feasible* if $\forall i : T(s_i, a_{i+1}, s_{i+1}) > 0$. We next relate trajectories and goals.

Definition 4. *A feasible trajectory* $\vec{\tau} = \langle s_0, a_1, s_1, \ldots, a_n, s_n \rangle$ satisfies *a possible goal* g *if* $\exists \pi_g \in \Pi_{leg}(g)$ *s.t.* $\forall i \in \{0 \ldots n-1\}$, $s_i \in S_{\pi_g}$ *and* $a_{i+1} = \pi_g(s_i)$.

Partial observability is materialized by limiting the observer to only see changes in emitted observations. To model this, we define an observable projection of a trajectory, where $\cdot$ denotes the concatenation of two sequences.

Definition 5. *The* observable projection *of a trajectory* $\vec{\tau} = \langle s_0, a_1, s_1, \ldots, a_n, s_n \rangle$ *is:* $obs(\vec{\tau}) = obs(\langle s_0 \ldots s_n \rangle) =$

$$\begin{cases} \langle N(s_0) \rangle & n = 0 \\ obs(\langle s_0 \ldots s_{n-1} \rangle) & n > 0 \wedge N(s_{n-1}) = N(s_n) \\ obs(\langle s_0 \ldots s_{n-1} \rangle) \cdot \langle N(s_n) \rangle & n > 0 \wedge N(s_{n-1}) \neq N(s_n) \end{cases}$$

For example, consider Figure 2(a). The *observable projection* of trajectory $\vec{\tau} = \langle S0, a_1, S2, a_3, S1, a_5, g_0 \rangle$ is $obs(\vec{\tau}) = \langle N(S0), N(S1), N(g_0) \rangle$.

Definition 6. *An observable projection* o satisfies *a possible goal* g *if there exists a feasible trajectory* $\vec{\tau}$ *that satisfies* g *and* $o = obs(\vec{\tau})$.

We denote by $G(\hat{\pi})$, $G(\vec{\tau})$, and $G(o)$ the set of goals satisfied by a partial policy $\hat{\pi}$, trajectory $\vec{\tau}$, and observation sequence o, respectively. Finally, we are ready to define non-distinctiveness.

Definition 7. *A partial policy* $\hat{\pi}$ *(respectively trajectory* $\vec{\tau}$ *or observation sequence* o*) is* non-distinctive *if it satisfies more than one goal* ($|G(\hat{\pi})| > 1$ *(respectively* $|G(\vec{\tau})| > 1$ *or* $|G(\vec{\tau})| > 1$*)).*
If a partial policy (respectively, trajectory or observation sequence) is not non-distinctive we say it is distinctive*.*

We now (re)define the expected worst case distinctiveness (*wcd*) for partially-observable models with stochastic actions. Distinctiveness cost is the total cost of the maximal prefix of a trajectory whose observable projection is non-distinctive. A partial policy $\hat{\pi}$ induces a *distribution* on trajectories, in which the probability of trajectory $\vec{\tau} = \langle s_0, a_1, s_1 \ldots, a_n, s_n \rangle$ is $P_{\hat{\pi}}(\vec{\tau}) = \Pi_{i=1}^{n} \mathcal{I}_{\hat{\pi}(s_{i-1})=a_i} P(s_i | s_{i-1}, a_i)$, where $\mathcal{I}$ is the indicator function that takes value 1 when $\hat{\pi}(s_{i-1}) = a_i$ and 0 otherwise. Therefore, the expected *wcd* of a goal recognition model is the (legal) partial policy with the maximal expected distinctiveness.

Definition 8. *The* distinctiveness cost $DC(\vec{\tau})$ *of a trajectory* $\vec{\tau} = \langle s_0, a_1, s_1 \ldots, a_n, s_n \rangle$ *is*

$$\max_{i \in \{0 \ldots n\} \text{ s.t. } |G(obs(\langle s_0, \ldots, a_i, s_i \rangle))| > 1} \sum_{j=1}^{i} \mathbf{C}(s_{j-1}, a_j, s_j) \quad (2)$$

The expected distinctiveness $ED(\hat{\pi})$ *of a partial policy* $\hat{\pi}$ *is the expected distinctiveness cost of its trajectories,* $\sum_{\vec{\tau}} P_{\hat{\pi}}(\vec{\tau}) DC(\vec{\tau})$.
The expected worst case distinctiveness *of a partially-observable goal recognition model with stochastic action outcomes PO is* wcd $= \max_{\hat{\pi} \in \hat{\Pi}_{leg}(G)} ED(\hat{\pi})$*, where* $\hat{\Pi}_{leg}(G) = \bigcup_{g \in G} \hat{\Pi}_{leg}(g)$.

Two notes are in order here. First, for an empty trajectory ($i = 0$), distinctiveness cost is defined to be 0. Second, $DC(\vec{\tau})$ and $ED(\hat{\pi})$ are well-defined for proper policies (as clarified in Section 2).

3.2 POS-GRD Problem Definition

Having defined our optimization measure (expected *wcd*) we can now discuss the design of a goal recognition model, using modifications, to minimize expected *wcd*.

Definition 9. *A partially-observable goal recognition design (POS-GRD) problem is given by the pair $T = \langle PO_0, \mathcal{D} \rangle$, where*

- *PO_0 is an initial partially-observable goal recognition model with stochastic action outcomes, and*
- *$\mathcal{D} = \langle \mathcal{M}, \delta, \phi \rangle$ is a* design model *where:*
  - *$\mathcal{M}$ is a set of applicable modifications,*
  - *δ is a modification function, specifying the effect of modifications on the partially-observable goal recognition model with stochastic action outcomes, and*
  - *ϕ is a constraint function that specifies the allowable modification sequences.*

We focus on two types of modifications. The first, *action removal* is defined as in [7], and removes an action from the set of applicable actions. The second, *state sensor refinement* (or *sensor refinement* for short), is defined next, allowing to distinguish between states previously mapped to the same observation. State sensor refinement is different from the sensor refinement reported in [10], which was defined over actions.

Definition 10. *A sensor model N' is a* refinement *of sensor model N if $\forall s_i, s_j : N'(s_i) = N'(s_j) \implies N(s_i) = N(s_j)$ (but not necessarily vice versa).*

Let PO^m represent the model that results from applying $m \in \mathcal{M}$ to PO and let N^m and N denote the sensor models of PO^m and PO, respectively.

Definition 11. *A modification m is a* state sensor refinement modification *if for any partially-observable goal recognition model with stochastic action outcomes PO, PO^m is identical to PO except that N^m is a refinement of N.*

Problem 1 (The POS-GRD problem)**.** *Let PO_0 be an initial partially-observable goal recognition model with stochastic action outcomes. Find a sequence of modifications $\vec{m} = \langle m_1 \dots m_n \rangle$, such that $\vec{m}$ is feasible (that is, $\phi(\vec{m}) = \top$), and which minimizes the expected* wcd *of the resulting model $PO_0^{\vec{m}} := (PO_0^{m_1})^{\cdots m_n}$.*

4 Solving POS-GRD Problems

$G(\hat{\pi})$, $G(\vec{\tau})$, and $G(o)$, the sets of goals satisfied by a partial policy $\hat{\pi}$, trajectory $\vec{\tau}$, and observation sequence o, respectively, demonstrate a non-Markovian behavior, which depends not just on the current state but also on what we have observed in the past. Intuitively, this is because once goal g has been eliminated as a possible goal (by observing the agent performing an action that is not part of any optimal policy with respect to g), then g never becomes a possible goal again, even if the agent executed an action that is optimal with respect to g. We therefore propose the use of augmented MDPs (first introduced in [18]) to capture the non-Markovian behavior in a partial observability setting.

4.1 Augmented MDP for POS-GRD

Let $PO = \langle M, \mathbf{G}, N \rangle$ be a partially-observable goal recognition model with stochastic action outcomes (Definition 1), with $M = \langle \mathbf{S}, s_0, \mathbf{A}, \mathbf{T}, \mathbf{C}, \emptyset \rangle$ being an MDP with positive costs $\mathbf{C}$ and no goal. An augmented MDP adds a Boolean variable pos_g for each possible goal g, to keep track of whether g has been eliminated as a possible goal or not. The terminal states of this MDP are those where there is exactly one possible goal, with transitions and costs defined according to the original MDP.

We account for partial observability by overlaying a sensor model on the augmented MDP. We first define a notion of connectivity in which an agent can transition from state s to state s', while following a policy that is optimal with respect to some goal $g \in G$, without being observed.

Definition 12. *State s is* unobservably connected *to state s' with respect to a set of possible goals G if there exists a policy $\pi \in \cup_{g \in G} \Pi_{leg}(g)$, and a trajectory $\vec{\tau} = \langle s_0, \pi(s_0), s_1, \pi(s_1), \dots, s_n \rangle$ with $s = s_0$ and $s' = s_n$, such that $N(s_0) = N(s_1) = \dots = N(s_n)$, and with $\mathbf{T}(s_i, \pi(s_i), s_{i+1}) > 0$ for $0 \leq i \leq n$.*

We denote by $uc_G(s)$ the set of states s' such that s is unobservably connected to s' with respect to $\mathbf{G}$, *e.g.*, in Figure 2(c), $uc_{G=\{g_0,g_1\}}(S1_1) = \{S1_1, S2_1\}$.

In a fully-observable setting, a transition from s to s' using an action a that is not part of an optimal policy to goal g, results in the removal of g from the set of possible goals. However, if $N(s) = N(s')$, the transition cannot be observed and g cannot be eliminated. Moreover, even when $N(s) \neq N(s')$, there may be another transition from $\hat{s} \in uc_{\{g\}}(s)$ to $\hat{s}'$ using action $\hat{a}$, such that $\mathbf{T}(\hat{s}, \hat{a}, \hat{s}') > 0$, $N(s) = N(\hat{s})$, $N(s') = N(\hat{s}')$, and $\hat{a}$ is an optimal action at $\hat{s}$ with respect to g. In this case, an observer cannot distinguish between the two transitions and as a result g still cannot be eliminated from the set of possible goals. As an example, consider Figure 2(a) and assume a fully-observable scenario (ignoring the shaded areas). Actions a_3 and a_4 (compensating actions for the stochasticity of actions a_1 and a_2, respectively) reveal the agent's goal since each one is optimal for only one (different) goal. In the partially-observable scenario of Figure 2(a), actions a_3 and a_4 are not observed as $N(S2) = N(S1)$ and even though $N(S0) \neq N(S1)$, if a_1 is executed, goal g_1 cannot be eliminated because a_2, optimal for g_1, also transitions from $N(S0)$ to $N(S1)$ and they cannot be distinguished.

Taking into account the observation above, the augmented MDP for POS-GRD $\Pi_{aug} = \langle \mathbf{S}', s_0', \mathbf{A}', \mathbf{T}', \mathbf{C}', g \rangle$ is defined as follows:

- $\mathbf{S}' = \mathbf{S} \times \{F, T\}^{|\mathbf{G}|}$: for each $s \in \mathbf{S}$ we create $2^{|\mathbf{G}|}$ possible states, corresponding to all subsets of possible goals. We use $w(s') = s$ to denote that s is the state of the world at $s' \in \mathbf{S}'$.
- $s_0' = s_0 \cdot \langle T \dots T \rangle$: initially all goals are possible.
- $\mathbf{A}' = \mathbf{A}$ (action labels remain unchanged).
- $\mathbf{T}'(s \cdot \langle pos_1 \dots pos_n \rangle, a, s' \cdot \langle pos_1' \dots pos_n' \rangle) =$

$$
\begin{cases}
T(s,a,s') & (s \notin g) \wedge \quad (3)\\
 & \forall i \in \{1 \dots, n\} (pos_i' = (pos_i \wedge \quad (4)\\
 & \big(N(s) = N(s') \quad (5)\\
 & \vee (\exists \pi \in \Pi_{leg}(g_i) \mid \pi(s) = a) \quad (6)\\
 & \vee (\exists \pi \in \Pi_{leg}(g_i) \wedge \exists \hat{s} \in uc_{\{g_i\}}(s) \wedge \quad (7)\\
 & \exists \hat{s}' : T(\hat{s}, \pi(\hat{s}), \hat{s}') > 0 \wedge \quad (8)\\
 & N(s) = N(\hat{s}) \wedge N(s') = N(\hat{s}'))))) \quad (9)\\
0 & \text{otherwise} \quad (10)
\end{cases}
$$

To compute whether the probability that executing action a when in state s with $\langle pos_1 \dots pos_n \rangle$ (where pos_i indicates whether goal

g_i is possible) leads to state s' with $\langle pos'_1 \dots pos'_n \rangle$ is equal to $T(s, a, s')$, we test the observer belief regarding each g_i according to the following cases. A goal cannot become possible (Line 4). A goal remains possible if s' is unobservably connected (Definition 12) to s (Line 5) or a is optimal with respect to the goal (Line 6). Finally, lines 7-9 cover the case of undistinguishable actions discussed above.

- $\mathbf{C}'(s \cdot \langle pos_1 \dots pos_n \rangle, a, s' \cdot \langle pos'_1 \dots pos'_n \rangle) = -\mathbf{C}(s, a, s')$: action costs flip sign to find policies with maximal cost in the original MDP, and
- $g = \{s \cdot \langle pos_1 \dots pos_n \rangle \mid \exists i : pos_i = T \wedge \forall j \neq i : pos_j = F\}$: terminal states are those where exactly one goal remains possible.

Lemma 1. *Let π' be any policy for the augmented MDP Π_{aug}, and define the non-distinctive partial policy $\hat{\pi}$ for PO by:*

$$\hat{\pi}(s) = \begin{cases} a & \pi'(s') = a \; \forall s' \in \mathbf{S}' \text{ s.t. } w(s') = s \\ \perp & \textit{otherwise} \end{cases} \tag{11}$$

Then $V_{\pi'}(s_0) = ED(\hat{\pi})$, that is, the value of policy π' in Π_{aug} at s_0 is equal to the expected distinctiveness of $\hat{\pi}$ in PO.

Lemma 1, proof of which we refrain from giving here for space consideration, connects the expected distinctiveness cost of a partial policy value to the expected reward from a policy in the augmented MDP. The following corollary establishes the connection to the expected *wcd*, leading to the algorithm to be detailed next for efficiently computing the expected *wcd* for optimal agents.

Corollary 1. *Let π' be an optimal policy for the augmented MDP Π_{aug}, and let $\hat{\pi}$ be as defined in Eq. 11, then $V_{\pi'}(s_0)$ is equal to the expected* wcd*.*

4.2 Constructing an Augmented MDP

Recall that expected *wcd* is defined as the maximal cost over all legal (optimal when using MDP's optimal policies) partial plans that aimed at more than a single goal (Definition 8). To avoid policy enumeration, we propose to consider all optimal policies simultaneously by creating a single augmented MDP and maximize the expected cost instead of minimizing it (as in MDPs). The number of augmented states is $O(|S| \times 2^{|G|})$, which is exponential in the number of model goals. However, not all augmented states are reachable, which provides us with an opportunity not to generate them all when computing expected *wcd* and redesigning the model. We offer next a method to generate exactly the augmented states needed for expected *wcd* computation, solving a single augmented MDP.

The proposed method has the following four steps: (1) Find all optimal policies; (2) Join them and remove infinite cycles to avoid computing the expected distinctiveness cost for each policy; (3) Construct the augmented MDP for reachable states taking partial observability into account; and (4) Solve this augmented MDP to compute the expected *wcd*. We next provide details of each of these steps.

Finding Optimal Policies: To identify $\Pi_{opt}(G)$, we separately solve an MDP for each goal. Using $V^*(s_0)$, the optimal expected cost at the starting state, we identify all optimal policies per goal. Figure 2(a) shows two optimal policies, one per goal, using two different edge widths (optimal policy to g_1 is marked in bold).

Removing Cycles: Combining all optimal policies into a single augmented MDP possibly creates cycles that are the result of joining two or more policies. Solving such an augmented MDP leads to optimal policies that choose to remain within the cycle to achieve an infinite maximum expected cost. For example, Figure 2(a) contains a cycle of actions a_3 and a_4, which belong to different optimal policies and therefore will never be executed together by an optimal agent. Therefore, we eliminate such cycles.

As a first step, we model the agent's true behavior using augmented MDPs assuming a fully-observable case. This step guarantees that there are no infinite loops [18]. In this step, states with only one possible goal that were not goal states are not yet defined as absorbing states. For each augmented state, the set of goals becomes part of the state ID. In what follows, we refer to this MDP as *cycle-free MDP* and to the sets of possible goals as *FO possible goals*.

For example, consider Figures 2(a)-(b). State $S0$ is augmented with goals g_0 and g_1 (denoted as subindex 01), then each action is analyzed to generate successors. When a_1 (optimal for goal g_0) is examined, states $S1$ and $S2$ are augmented with goal g_0, and when action a_2 (optimal for goal g_1) is analyzed, $S1$ and $S2$ are augmented with goal g_1. Later, when action a_3 is analyzed, no new state needs to be generated as $S1$ augmented with goal g_0 was already created; a similar situation occurs with a_4. It is worth noting that all the augmented states generated from the same state (*e.g.*, $S1$ or $S2$) project to the same observation. Also, it is worth noting that after removing the cycles in the use-case, the resulting MDP has separate distinguished paths for each goal.

Constructing augmented MDP for POS-GRD: To generate the set of possible goals and their corresponding transition function for reachable states of the cycle-free MDP, we use an iterative 7-step procedure, reaching each non-distinctive augmented state: (1) Augment the initial state with all possible goals; (2) Find all unobservably connected states and augment them with the same set of goals; (3) Find all immediately connected states projecting successors not belonging to the unobservably connected states; (4) Group them according to their observations; (5) Augment states in each group; (6) Keep non-duplicated augmented states; and (7) Update the transition function. The resultant transition function is equivalent to the augmented transition function (Section 4.1) for all augmented reachable states.

To illustrate the procedure, consider the resulting augmented MDP in Figure 2(c). Possible goals for a state are shown as indices to the ID of their observations. The start sate $S0_{01}$ is augmented with goals g_0 and g_1 (Step 1). States $S1_0, S2_0, S1_1$, and $S2_1$ are generated and since (1) actions a_0 and a_1 are each optimal for different goals; (2) both of them transition from $O1$ to $O2$; and (3) they are non-observable, then all these states are augmented with goals g_0 and g_1 (Steps 3 to 5). The transition function in this case does not change (Step 7). When $S1$ and $S2$ are examined, all their unobservably connected states should be first generated and augmented with the same set of goals. However, in this case, no new state needs to be created. Later, other connected states not projecting both g_0 and g_1 are generated and augmented following the same procedure.

Algorithm 1 presents a pseudocode for constructing an augmented MDP for POS-GRD,[7] receiving as input a goal recognition model and a set of optimal policies. The algorithm initially builds a cycle-free MDP (lines 1-3) and initializes the output variables and a stack (line 4). Then, the 7-step procedure starts. Step 1 is executed in line 5 and the stack is used to find successors in a DFS-fashion (lines 6-26). Each augmented state in the stack is explored to generate its immediate successors (lines 8-10). Following steps 2 and 3, the algorithm finds and augments successors (lines 11-12). The set of possible goals to temporarily augment a successor found in step 3 corresponds to the intersection of its predecessors' goals with the set of

[7] Source code is available at https://github.com/cwayllace/POS-GRD

goals for which the action executed to arrive at it is optimal (line 13). Next, successors are grouped as specified in step 4 (lines 14-15) and the final set of goals per group is generated (lines 18-19). The algorithm uses this set to augment states in that group (step 5, line 21). If the newly augmented state was not explored before, it is added to the stack for future exploration (step 6, lines 23-24), and to the set of augmented goals if it is distinctive (line 25). Finally, the augmented transition function is updated in line 26 (step 7) and once all reachable augmented states have been explored, the algorithm returns all parameters of an augmented MDP.

Algorithm 1: AugMDP-PO($M, G, N, \Pi_{opt}(G)$)

1 $\langle \hat{s}_0, \hat{S}, \hat{T}, \hat{G} \rangle \leftarrow AugMDP\text{-}FO'(M, G, \Pi_{opt}(G))$
2 **foreach** $\hat{s} \in \hat{S}$ **do**
3 **if** $\hat{s} = s \cdot \langle pos_1 \dots pos_{|G|} \rangle = s \cdot \langle G' \rangle$ **then** $\hat{s} \leftarrow sG'; N(\hat{s}) \leftarrow N(s)$;
4 $G', S', A', Stack \leftarrow \emptyset; T' \leftarrow null$
5 $s'_0 \leftarrow \hat{s}_0 \cdot \langle \hat{G} \rangle$
6 $Stack.push(s'_0)$
7 **while** $stack \neq \emptyset$ **do**
8 $s' \leftarrow \hat{s} \cdot \langle \hat{G'} \rangle \leftarrow Stack.pop()$
9 $Keys \leftarrow \emptyset; Map \leftarrow null$
10 **foreach** $\hat{T}(\hat{s}, \pi(\hat{s}), \hat{s}') > 0 | \pi(\hat{s}) \in \Pi_{opt}(\hat{G''})$ **do**
11 **if** $N(\hat{s}) = N(\hat{s}')$ **then** $s'' \leftarrow \hat{s}' \cdot \langle \hat{G'} \rangle$;
12 **else**
13 $s'' \leftarrow \hat{s}' \cdot \langle \hat{G'} \cap \hat{G''} \rangle$
14 $Keys \leftarrow Keys \cup \{N(s'')\}$
15 $Map(N(s'')) \leftarrow Map(N(s'')) \cup \{\langle s', \pi(\hat{s}), s'' \rangle\}$
16 **foreach** $k \in Keys$ **do**
17 $Goals \leftarrow \emptyset$
18 **foreach** $\langle s', \pi(\hat{s}), \hat{s''} \cdot \langle \hat{G''} \rangle \rangle \in Map(k)$ **do**
19 $Goals \leftarrow Goals \cup \hat{G''}$
20 **foreach** $\langle s', \pi(\hat{s}), \hat{s''} \cdot \langle , \hat{G''} \rangle \rangle \in Map(k)$ **do**
21 $s'' \leftarrow \hat{s''} \cdot \langle Goals \rangle$
22 **if** $s'' \notin S'$ **then**
23 $S' \leftarrow S' \cup \{s''\}$
24 $Stack.push(s'')$
25 **if** $|Goals| \leq 1$ **then** $G' \leftarrow G' \cup \{s''\}$;
26 $T'(s', \pi(\hat{s}), s'') \leftarrow T(\hat{s}, \pi(\hat{s}), \hat{s}')$
27 return $(\langle s'_0, S', A', T', G' \rangle)$

Computing *wcd*: The maximum expected cost can be found using a TVI-like algorithm [4] to solve the augmented MDP where instead of using Eq. 1, the minimization condition is replaced with a maximization condition, due to the reversal of the cost function $\mathbf{C}'$ in the augmented MDP, see Section 4.1.

4.2.1 Discussion of Correctness and Complexity

Step 6 ensures that no duplicated augmented state is generated. Therefore, at most $O(|S| \times 2^{|G|})$ augmented states are generated; Here, S is the set of states of the cycle-free MDP and G is the set of goals. However, since no successors are generated for states with less that 2 observable goals, and only reachable states are considered, the actual number of generated augmentes states is lower.

An augmented state is expanded at Step 2 only once and only if it is non-distinctive. At each iteration, all successors are ready to be analyzed and their respective augmented transition function has been created/updated.

Steps 3-5 guarantee that the resultant augmented MDP structure is unique and does not depend on the type of traversal of the cycle-free MDP.

The complexity of constructing an augmented MDP is therefore $O(|\Xi| + \Lambda)$, where Ξ is the set of augmented reachable states and Λ is the average number of state-action successors.

4.3 Reducing Expected *wcd*

Having shown our proposed method for computing expected *wcd* in a partially observable stochastic setting, we now propose two modification types, namely action removal and sensor refinement, to reduce the expected *wcd* of a given model. Action removal has been shown multiple times in the Goal Recognition Design literature and here we show how it is implemented in the stochastic setting under partial observability. In addition, we present a new modification of sensor refinement over states. We note that modifications to the model are introduced to the MDP, which is augmented once more to compute the revised expected *wcd*.

Action Removal: In this modification, up to k actions are removed with the objective of minimizing expected *wcd*. The naïve approach would be to remove all combinations of up to k actions and compute the expected *wcd* for each combination. To prune the search space, we perform an iterative search, working our way up from a single action to multiple actions. We first remove every legal action in turn. Any action whose removal leads to unreachable goals is pruned, as any combination with such an action will not change the expected *wcd* value. Next, we iteratively increase the action set size and repeat the same analysis, seeking a set of actions whose removal leads to unreachable goals and prune any combination containing it.

Sensor Refinement: Partial observability reduces an observer's ability to distinguish agent states. The design objective is thus to identify sensors whose refinement yields expected *wcd* improvement under budget constraints. At the extreme, refining all sensors leads to a fully-observable model, yet we note that in our model, even with all sensors refined actions are still unobservable.

We consider a sensor refinement modification that refines a **single** state to make it fully observable (Definition 11). Hence, if an agent is in any of the *refined states*, its state is known with full certainty. Figure 2(d) shows the augmented MDP after $S1$ is refined (leading to the refinement of $S2$ as well) and mapped to a new observation $O5$, allowing the observer to distinguish it from $S2$. After refinement, expected *wcd* is reduced to 2.0. Note that all distinctive states ($S2_1, S1_0$, and the two goal states) are terminal. Also, states can be augmented with different sets of possible goals. As an example, state $S1_0$ has two augmented versions: one with both goals $O5_{01}$, and the other with goal g_1 ($O541$). The observer's knowledge is represented through the set of possible goals, which changes according to the observed sequence. Therefore, sequence $\langle O1, O54, O2 \rangle$ reveals the agent's true goal whereas sequence $\langle O1, O2 \rangle$ does not. Finally, the augmented model represents every possible trajectory an agent can traverse.

A naïve approach for using this modification involves finding all combinations of up to k states to refine, compute a new expected *wcd* for each combination, and choose the combination that minimizes its value. However, note that once refining a state, if the set of possible goals of any other state does not change, then refining

that state would not affect *wcd* as the structure of the augmented MDP (*i.e.*, set of augmented goals and transition function) remains the same. Given this observation, we propose the use of two pruning methods to improve scalability.

First, we identify and prune from any combination states that, if refined, are guaranteed not to change the augmented MDP structure. We prune states whose all augmented versions have the same set of goals under full and partial observability, and whose predecessors and successors share the same set of possible goals, then refining that state would not affect wcd". Finding those states is linear in the size of the augmented states and it requires to store the set of goals found assuming full observability while the augmented MDP is being built (FO possible goals).

Second, we leverage the fact that the best solution is a fully-refined model. Therefore, we refine all states within a single observation and if the expected *wcd* is not reduced, we prune all combinations of states that are part of the observation. This step requires solving the augmented MDP o times where o is the number of observations. However, the number of pruned combinations is usually much larger than o.

5 Empirical Evaluation

Our experiments aim at evaluating: (1) algorithms scalability and (2) effectiveness of pruning. In what follows we introduce the evaluation datasets (Section 5.1), the experiment settings (Section 5.2), and an analysis of the results (Section 5.3).

5.1 Data

We evaluate our algorithms on five domains:

1 GRID-NAVIGATION, a grid world where the agent has a 90% chance of success when moving to an adjacent cell.
2 ROOM, a grid world where actions and transition probabilities are defined individually for each state.
3 BLOCKSWORLD, with a 25% probability of slippage each time a block is picked up or put down. Each block has a color and the goal is specified in terms of colors.
4 BOXWORLD, a modified LOGISTICS domain where the only action that introduces uncertainty is "drive-truck" and there is a 20% probability that the truck ends up in one of two wrong cities.
5 ATTACK-PLANNING, a cybersecurity domain where each host on a network has a set of stochastically assigned vulnerabilities and a random subset of hosts has files that an attacker may want to access [17]. The initial state contains random user credentials that the attacker obtained (*e.g.*, through phishing attacks). An attacker can perform one of three types of malicious actions:

- **Exploit** existing vulnerabilities in a host to gain read access to the target file or, in case of failure, compromise the host. The success probability is derived from the industry standard *Common Vulnerability Scoring System* [12].
- **Update** gains network access to a host connected to a compromised host for which the attacker has network access. Update has an 80% success probability.
- **Access** a target file if it has read access at the host where the file is located. Access is deterministically successful.

To induce partial observability in the grid domains, we map up to four contiguous states to the same observation. For BLOCKSWORLD, observability was affected in two ways: either a random number of blocks were non-observable, or the effects of actions involving some randomly selected blocks were hidden. In the BOXWORLD domain, trucks or airplanes may be unobservable in some random cities. For ATTACK-PLANNING, initial sensor configurations per instance define whether the effect of *update* is observable for a given host.

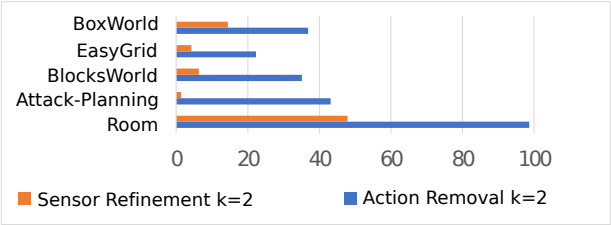


Figure 3. Average Percentage of Reduced Combinations

5.2 Settings

Experiments ran with a budget (of allowed modifications) of $k \in \{1, 2, 3\}$ in a total of 40 instances. We consider three combination of modifications: (1) sensor refinement only; (2) action removal only; and (3) both. *wcd* computation uses the cycle-free MDP to generate the augmented MDP. Experiments were conducted on a 2.10 GHz machine with 16 GB of RAM and a timeout of 2 days.

5.3 Preliminary Results

We first analyze the fraction of instances where *wcd* is reduced for each combination of modifications and budget. Overall, 83.3% of the instances had a reduced *wcd* with a budget of $k = 1$, 97.2% with $k = 2$, and for all the 64% of instances that were able to finish using $k = 3$, *wcd* was reduced. Action removal (56.8%) does not perform as well as sensor refinement (78.9%). The maximum reduction obtained was from 79 to 48 for action removal and from 79 to 38 for sensor refinement. This is likely due to the partially-observable setting, where refining sensors makes the model more similar to a fully-observable case. The gap is almost uniform for all k values. It is worth noting that the extent of reduction depends on the transition function, the relative position of possible goals and initial state, and the initial sensor configuration as they determine the distinctiveness of the policies. We assigned goals and initial state randomly to get a general understanding of the competitive advantage of sensor refinement and action removal.

Using both modifications reduced *wcd* more than using each modification separately in 15.8% of the cases. Specifically, for 19.4% of the instances, *wcd* was reduced more when both types of modifications were used ($k = 2$) and 34.8% of the finished instances ($k = 3$).

Figure 3 illustrates the impact of pruning on *wcd* reduction. We compare the number of total combinations of modifications needed before and after pruning, taking the percentage of reduction (larger is better). Generally, action removal benefited more than sensor refinement from pruning. The ROOM domain benefited the most, mainly because there were few optimal policies and removing actions made a large part of the state space, or even a goal, unreachable.

Figure 4 shows the average running time in seconds per domain for all three values of k when the pruning methods were used (AR: action removal, SR: sensor refinement, ARSR: both). Instances that timed-out are not considered in the computation and reduced average time with higher values of k is a result of this omission. Specifically,

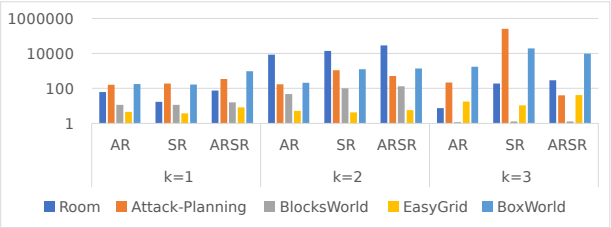


Figure 4. Average Running Time in sec. (Logarithmic Scale)

with a budget of $k = 2$, one instance of BLOCKSWORLD timed-out when SR and ARSR were used. With a budget of $k = 3$, a total of 12/17/19 instances timed-out when AR/SR/ARSR were used (5/5/5 from ROOM, 3/3/4 from ATTACK-PLANNING, 4/4/4 from BLOCKSWORLD, and 0/5/6 from BOXWORLD). Clearly, runtime increases with k. The largest number of actions to remove was 1979 and the largest augmented MDP generated had 16376 augmented states.

6 Related Work

Goal recognition design, first introduced by Keren *et al.* [7] and followed by several extensions [8, 15], offers tools to analyze and solve a GRD model in fully-observable settings. Other works account for non-observable actions [9] and partially-observable actions [10], supporting non-deterministic sensor models that can reflect an arbitrary assignment of action tokens emitted by actions. All these models assume deterministic action outcomes. In contrast, Wayllace *et al.* [19, 18] tackled GRD models with stochastic action outcomes in fully-observable environments. Our proposed model generalizes these two lines of research into a GRD model with stochastic action outcomes in partially-observable environments.

Partial observability in goal recognition has been modeled in various ways [14, 6, 1], usually assuming the agent deals with partial observability. In particular, observability is modeled using a sensor model that includes an observation token for each action [5]. Our sensor model considers observer's partial observability and uses state observations, rather than action observation, providing a basis for many practical applications.

7 Conclusions and Future Work

We present a new GRD variation that accounts for partially-observable states and stochastic action outcomes, which is relevant to many applications such as agent navigation. The agent state is observable, subject to sensor resolution, which means that some states can be perceived as identical, also, the intention of movement is unknown. In response to these considerations, we propose the *Partially-Observable Stochastic GRD* (POS-GRD) problem, where (1) actions are not observable and (2) states are partially observable. Our formal framework takes partial observability into account to compute *wcd* for POS-GRD problems. We also provide a skeleton description of a new algorithm to compute expected *wcd* and propose a new model modification of state sensor refinement to reduce expected *wcd*.

POS-GRD is a new general GRD model, the first to handle partial observability in stochastic GRDs. In addition, sensor modification over states is a new model modification. These contributions combined make a substantial (and challenging!) step towards increasing the generality of the GRD framework, allowing it to model more interesting and realistic applications (*e.g.*, cybersecurity).

Preliminary experiments show that combining modification types reduces expected *wcd* for more instances with less budget.

An immediate non-trivial avenue for future research is to extend our techniques for computing and reducing expected *wcd*. While the proposed model is sufficiently general to model sub-optimal agents (using Π_{leg}), our use of MDP optimal policies limit our solution to optimal agents (Π_{opt}). Moving beyond optimality in the stochastic case is more involved than in the deterministic counterpart.

Acknowledgments

This research is partially supported by NSF grants 1810970 and 1838364.

REFERENCES

[1] Dorit Avrahami-Zilberbrand, Gal Kaminka, and Hila Zarosim, 'Fast and complete symbolic plan recognition: Allowing for duration, interleaved execution, and lossy observations', in *Proceedings of the AAAI Workshop on Modeling Others from Observations, MOO*, (2005).
[2] Richard Bellman, *Dynamic Programming*, Princeton University Press, 1957.
[3] Sandra Carberry, 'Techniques for plan recognition', *User Modeling and User-Adapted Interaction*, **11**, 31–48, (2001).
[4] Peng Dai, Mausam, Daniel S Weld, and Judy Goldsmith, 'Topological value iteration algorithms', *Journal of Artificial Intelligence Research*, **42**, 181–209, (2011).
[5] Hector Geffner and Blai Bonet, *A Concise Introduction to Models and Methods for Automated Planning*, Morgan & Claypool Publishers, 2013.
[6] Christopher W Geib and Robert P Goldman, 'Partial observability and probabilistic plan/goal recognition', in *Proceedings of the International Workshop on Modeling Others from Observations, MOO*, (2005).
[7] Sarah Keren, Avigdor Gal, and Erez Karpas, 'Goal recognition design', in *Proceedings of ICAPS*, pp. 154–162, (2014).
[8] Sarah Keren, Avigdor Gal, and Erez Karpas, 'Goal recognition design for non-optimal agents', in *Proceedings of AAAI*, pp. 3298–3304, (2015).
[9] Sarah Keren, Avigdor Gal, and Erez Karpas, 'Goal recognition design with non-observable actions', in *Proceedings of AAAI*, pp. 3152–3158, (2016).
[10] Sarah Keren, Avigdor Gal, and Erez Karpas, 'Privacy preserving plans in partially observable environments', in *Proceedings of IJCAI*, pp. 3170–3176, (2016).
[11] Mausam and Andrey Kolobov, *Planning with Markov Decision Processes: An AI Perspective*, Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers, 2012.
[12] Peter Mell, Karen Scarfone, and Sasha Romanosky, *A Complete Guide to the Common Vulnerability Scoring System Version 2.0*, NIST and Carnegie Mellon University, 1 edn., June 2007.
[13] Miquel Ramírez and Hector Geffner, 'Probabilistic plan recognition using off-the-shelf classical planners', in *Proceedings of AAAI*, (2010).
[14] Miquel Ramírez and Hector Geffner, 'Goal recognition over POMDPs: Inferring the intention of a POMDP agent', in *Proceedings of IJCAI*, pp. 2009–2014, (2011).
[15] Tran Cao Son, Orkunt Sabuncu, Christian Schulz-Hanke, Torsten Schaub, and William Yeoh, 'Solving goal recognition design using ASP', in *Proceedings of AAAI*, pp. 3181–3187, (2016).
[16] Gita Sukthankar, Christopher Geib, Hung Hai Bui, David Pynadath, and Robert P Goldman, *Plan, activity, and intent recognition: Theory and practice*, Newnes, 2014.
[17] Yevgeniy Vorobeychik and Michael Pritchard, 'Plan interdiction games', *CoRR*, (2018).
[18] Christabel Wayllace, Ping Hou, and William Yeoh, 'New metrics and algorithms for stochastic goal recognition design problems', in *Proceedings of IJCAI*, pp. 4455–4462, (2017).
[19] Christabel Wayllace, Ping Hou, William Yeoh, and Tran Cao Son, 'Goal recognition design with stochastic agent action outcomes', in *Proceedings of IJCAI*, pp. 3279–3285, (2016).