PRELIMINARY PREPRINT VERSION: DO NOT CITE
The AAAI Digital Library will contain the published
version some time after the conference.

Stochastic Goal Recognition Design Problems with Suboptimal Agents

Christabel Wayllace,[1] **William Yeoh**[2]

[1] University of Alberta
[2] Washington University of St Louis
wayllace@ualberta.ca, wyeoh@wustl.edu

Abstract

Goal Recognition Design (GRD) problems identify the minimum number of environmental modifications aiming to force an interacting agent to reveal its goal as early as possible. Researchers proposed several extensions to the original model, some of them handling stochastic agent action outcomes. While this generalization is useful, it assumes optimal acting agents, which limits its applicability. This paper presents the Suboptimal Stochastic GRD model, where we consider boundedly rational agents that, due to limited resources, might follow a suboptimal policy. Inspired by theories on human behavior asserting that humans are (close to) optimal when making perceptual decisions, we assume the chosen policy has at most u suboptimal actions. Our contribution includes (i) Extending the stochastic goal recognition design framework by supporting suboptimal agents in cases where an observer has either full or partial observability; (ii) Presenting methods to evaluate the ambiguity of the model under these assumptions; and (iii) Evaluating our approach on a range of benchmark applications.

Our ability to recognize other people's plans and goals relies on the assumption that most human behavior is goal-oriented. Nowadays, our interactions are not limited to humans but also artificial agents; as human-machine interaction and automated systems' intelligence grows, so does the need to understand each other's objectives. Therefore, researchers aim to provide AI agents with *goal recognizing* capabilities (Sukthankar et al. 2014). Research in goal recognition (GR) studies the problem of determining an agent's goal by observing its behavior (Sukthankar et al. 2014; Ramírez and Geffner 2010). Therefore, one of GR's main problems is to deal with ambiguous behavior. While most researchers focused on inferring the true goal promptly and correctly, Keren, Gal, and Karpas (2014) proposed to modify the environment aiming to reduce ambiguity for the observer. By doing so, the re-designed environment induces an acting agent to reveal its goal earlier. Keren, Gal, and Karpas (2014) call this problem *Goal Recognition Design* (GRD).

Typically, a GRD problem: (1) Evaluates the GR setting using a measure and (2) Finds minimal changes to the underlying environment aiming to optimize this measure. In their work, Keren, Gal, and Karpas (2014) proposed the worst-case distinctiveness (*wcd*) measure, which finds the maximum number of actions an agent can execute without revealing its goal. They modified the environment using action removal. Current work on GRD considers deterministic and stochastic settings (Keren, Gal, and Karpas 2015, 2016a,b; Keren et al. 2020; Son et al. 2016; Wayllace et al. 2016; Wayllace, Hou, and Yeoh 2017; Wayllace et al. 2020; Keren, Gal, and Karpas 2021) with different assumptions regarding observability, optimality, and type of goal recognition (Cohen, Perrault, and Allen 1981). Work in the stochastic GRD (S-GRD) framework assumes optimal acting agents in scenarios where an observer has full or partial observability (Wayllace et al. 2016, 2020). In this paper we relax the optimality assumption for both cases and consider boundedly rational agents that might follow suboptimal policies.

Many intelligent agents use human prediction models, studied in psychology and economics (Rosenfeld and Kraus 2018). Like humans, AI agents cannot account for every aspect of the world's dynamics; so they will probably need to deviate from optimal behavior. Therefore, we decided to consider close to rational policies with a limited number of suboptimal actions. Accounting for slightly suboptimal policies arguably improves the goal recognition model since there is a higher probability that one of them explains the agent's criteria. This decision also aligns with work on goal recognizers using top-k planners to improve goal recognition in domains with unreliable observations (Sohrabi, Riabov, and Udrea 2016; Riabov, Araghi, and Udrea 2020).

The contributions of this paper are: (i) Generalization of the S-GRD framework to support suboptimal agents where observers have full or partial observability; (ii) Delivery of a method to compute the maximal non-distinctive cost under these assumptions; (iii) Empirical evaluation of our approach on a range of benchmark applications.

Background

Markov Decision Process (MDP): A *Stochastic Shortest Path Markov Decision Process* (SSP-MDP) (Mausam and Kolobov 2012) is represented as a tuple $\langle \mathbf{S}, s_0, \mathbf{A}, \mathcal{T}, \mathcal{C}, \mathbf{G} \rangle$. 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 $\mathcal{T} : \mathbf{S} \times \mathbf{A} \times \mathbf{S} \rightarrow [0, 1]$ that gives the probability $\mathcal{T}(s, a, s')$ of transitioning from state s to s' when action a is executed; a cost function $\mathcal{C} : \mathbf{S} \times \mathbf{A} \times \mathbf{S} \rightarrow \mathbb{R}$ that gives the cost $\mathcal{C}(s, a, s')$ of ex-

Copyright © 2022, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

ecuting action a in state s and arriving in state s'; and a set of goal states $\mathbf{G} \subseteq \mathbf{S}$. The goal states are terminal, i.e., $\mathcal{T}(s, a, s') = 1$ and $\mathcal{C}(g, a, g) = 0$ for all goal states $g \in \mathbf{G}$ and actions $a \in \mathbf{A}$. An SSP-MDP (MDP hereinafter) must also satisfy the following two conditions: (1) There must exist a *proper policy*, which is a mapping from states to actions with which an agent can reach a goal state from any state with probability 1. (2) Every *improper policy* must incur an accumulated cost of ∞ from all states from which it cannot reach the goal with probability 1. Solving an MDP is to find an optimal policy π^* with the smallest expected cost.

Value Iteration (VI) (Bellman 1957) is one of the fundamental algorithms to find an optimal policy. The expected cost $V(s_0)$ of an optimal policy π^* for the starting state $s_0 \in \mathbf{S}$ and the expected cost $V(s)$ for all states $s \in \mathbf{S}$ are calculated using the Bellman equation (Bellman 1957):

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

VI suffers from a limitation that it updates each state in every iteration even if its expected cost has converged. *Topological VI* (TVI) (Dai et al. 2011) addresses this limitation by detecting the MDP structure and updating states grouped in topological sequences. It first divides the MDP into strongly connected components (SCCs) and repeatedly updates the states in only one SCC until their values converge before updating the states in another SCC. Since the SCCs form a directed acyclic graph, states in an SCC only affect the states in upstream SCCs. Thus, by choosing the SCCs in reverse topological sort order, it no longer needs to consider SCCs whose states have converged in a previous iteration. TVI is guaranteed to terminate and to converge to an optimal value function.

GRD and (PO)S-GRD: A *Goal Recognition Design* (GRD) problem (Keren, Gal, and Karpas 2014) is represented as a tuple $T = \langle P, \mathcal{D} \rangle$, where P is an initial GR model and $\mathcal{D}$ is a design model. P, in turn, is represented by the tuple $\langle D, \mathbf{G} \rangle$, where D captures the domain information and $\mathbf{G}$ is a set of possible goal states of the agent. *Stochastic GRD* (S-GRD) (Wayllace et al. 2016) extends the GRD framework by assuming that actions executed by the agent have stochastic outcomes. The elements of $D = \langle \mathbf{S}, s_0, \mathbf{A}, \mathbf{T}, \mathbf{C} \rangle$ are as described in MDPs, except that the cost function $\mathbf{C}$ is restricted to positive costs (assumed to be 1 for simplicity) and in case of GRDs, the transition function $\mathbf{T}$ is deterministic.

In GRD problems, the *worst-case distinctiveness* (*wcd*) of problem P is the length of the longest sequence of actions that is the prefix in *cost-minimal* plans to distinct goals. For S-GRDs, the *wcd* is the highest *expected* cost of policy prefixes common to multiple goals. Wayllace et al. (2020) define the *worst case distinctiveness* of a stochastic GR problem P as:

$$wcd = \max_{\hat{\pi} \in \hat{\Pi}_{\mathbf{G}}} \sum_{\vec{\tau}} P_{\hat{\pi}}(\vec{\tau}) DC(\vec{\tau}) \quad (2)$$

where: (i) $\hat{\Pi}_{\mathbf{G}} = \bigcup_{g \in \mathbf{G}} \hat{\Pi} g$ is the set of *all legal policies* of P for *all possible goals*. (ii) The probability of trajectory $\vec{\tau} = \langle s_0, a_1, s_1 \ldots, a_n, s_n \rangle$ is $P_{\hat{\pi}}(\vec{\tau}) = \Pi_{i=1}^{n} \mathcal{I}_{\hat{\pi}(s_{i-1})=a_i} P(s_i | s_{i-1}, a_i)$; $\mathcal{I}$ is the indicator function that takes value 1 when $\hat{\pi}(s_{i-1}) = a_i$ and 0 otherwise. (iii) The *distinctiveness cost* $DC(\vec{\tau})$ of a trajectory $\vec{\tau}$ is

$$\max_{i \in \{0 \ldots n\} \text{ s.t. } |\mathbf{G}(ob(\langle s_0, \ldots, a_i, s_i \rangle))| > 1} \sum_{j=1}^{i} \mathcal{C}(s_{j-1}, a_j, s_j) \quad (3)$$

$\mathbf{G}(ob(\vec{\tau}))$ represents the set of goals satisfied by the observed sequence of a trajectory. In S-GRD $ob(\vec{\tau}) = \vec{\tau}$ whereas in POS-GRD, $ob(\vec{\tau})$ contains only a subset of the states, not the actions.

A design model $\mathcal{D}$ (Keren, Gal, and Karpas 2018) includes the set of applied modifications, a modification function defining their effects, and a constraint function to specify the modification sequences used. The seminal work suggests action removal (Keren, Gal, and Karpas 2014) and Wayllace et al. (2020) propose state sensor refinement as possible modifications.

The objective in (POS-)GRD is to find a feasible modification sequence that, when applied to the initial goal recognition model P, will minimize the measure of the problem.

Augmented MDP for (PO)S-GRD: In (PO)S-GRD problems, the set of possible goals for a particular state is not Markovian as it depends on the path used to reach that state. For this reason, (PO)S-GRD algorithms use augmented MDPs to compute the measure where states incorporate the set of possible goals given the trajectory used. Any augmented state whose successors have less than two possible goals becomes an absorbing state. The value of *wcd* is the *largest* expected cost at the initial augmented state. TVI-like algorithms are used to compute *wcd* (Wayllace, Hou, and Yeoh 2017; Wayllace et al. 2020).

Suboptimal Stochastic Goal Recognition Design (SS-GRD)

The difficulty of GR arises when the observed partial trajectory can explain multiple goals. GRD tries to reduce the complexity of GR by decreasing the size of those ambiguous policies. However, the number and length of non-distinctive trajectories explode when assuming suboptimality and partial observability. Moreover, simultaneously considering all ambiguous policies under the new assumptions may generate infinite loops. This work presents methods to tackle the complexity added by the new assumption and shows that none of the optimizations in previous work are applicable. Since policy enumeration is required to exactly compute *wcd*, the optimization minimizes policy re-evaluation and focuses on the design part.

Due to the offline nature of GRD problems, modifications assuming incorrect agent behavior might not help a goal recognizer by reducing the complexity of the original problem. We present a simple example where a system assuming optimal agents does not find any helpful modification, whereas some changes are possible when considering suboptimal agents. The modified environment will help the observer infer the goal sooner, reduce the decision-making load to the agent, and be more robust to different agents.

The problem in hand considers two interested parties: boundedly rational agents aiming to achieve a goal and observers trying to detect the agent's real goal as early as possible. Agents can execute up to u suboptimal actions (which could have stochastic outcomes). We require that an agent selects only *proper* and *stationary* policies and assume they are unaware of the recognition process (i.e., *keyhole* goal recognition). Low sensor resolution may limit observers, in which case they cannot perceive the agent's executed actions, and some states can be indistinguishable from others. We aim to find a limited set of environmental modifications to help the observer's task without hindering the agent's objective. Due to the offline property of GRD problems, we need to account for *all possible* sequences of observations generated by an agent.

Model

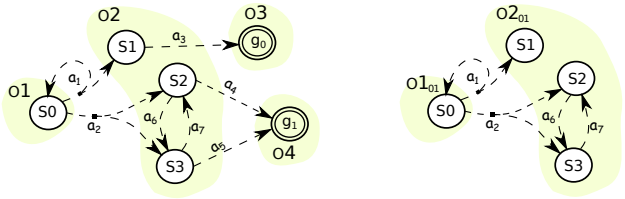
The model for SS-GRD problems accounts for a stochastic *environment*, the *observer*'s capability of perceiving the actor's behavior, and the degree of *agent*'s suboptimality. Our model has two components: the *initial GR* model and the *design* model. The design model describes how to modify the initial GR setting to reduce ambiguity. We formulate each component separately before defining the SS-GRD problem formally. This section also considers *wcd* as the measure to evaluate a GR model and redefines it for SS-GRD.

Stochastic Goal Recognition (Stochastic GR)

Definition 1 (Stochastic Goal Recognition Model)**.** *A stochastic goal recognition problem P is a tuple $P = \langle M, \mathbf{G}, u, \mathcal{N}, \mathcal{C}_o \rangle$, where: (1) $M = \langle \mathbf{S}, s_0, \mathbf{A}, \mathcal{T}, \mathcal{C} \rangle$ is an SSP-MDP without a goal. The four first elements model the world mechanics, and the cost function $\mathcal{C} : \mathbf{S} \times \mathbf{A} \times \mathbf{S} \to \mathbb{R}^+$ specifies the agent's cost $\mathcal{C}(s, a, s')$ of taking action a at state s and arriving to state s'. (2) $\mathbf{G}$ is a set of candidate goals, i.e., $\forall g \in \mathbf{G}$, $\mathbf{G} \subseteq \mathbf{S}$; g is a possible goal of the agent. (3) $u \geq 0$ is the degree of suboptimality allowed, where $u = 0$ represents optimal agents. (4) $\mathcal{N}$ is a sensor function that defines the observer's degree of observability. In partially-observable (PO) models, each state s is associated with an observation $\mathcal{N}(s)$, which we refer to the* projected observation *of s. The set $\mathbf{S}$ is partitioned into observation sets $\mathbf{O}_1, ..., \mathbf{O}_n$ such that $\forall s, s' : \mathcal{N}(s) = \mathcal{N}(s') \iff \exists i : s, s' \in \mathbf{O}_i$. In fully-observable (FO) models, $\mathcal{N}$ is an identity function. (4) $\mathcal{C}_o$ is the observer's cost function $\mathcal{C}_o : \mathbf{S} \times \mathbf{A} \times \mathbf{S} \to \mathbb{R}^+$ that assigns a potentially different cost to each agent's action.*

The model considers distinct cost functions for agents and observers, allowing each to specify their preferences without affecting the other. For example, two ambiguous policies with the same cost for the agent but of different lengths would be indifferent to the agent, but the observer should prefer the shorter one. In this paper, we assigned a cost of 1 to all actions in both cases.

To illustrate the PO setting, consider Fig. 1(a), where circles represent states, dashed arrows denote non-observable actions, and green bubbles stand for observations ($O1 - O4$) that group undistinguishable states. All actions have a cost



(a) (b)

Figure 1: PO SS-GRD. (a) Original MDP. (b) Non-distinctive Observations.

of execution equal to 1; agents executing action a_1 have a 90% probability of succeeding, and if choosing action a_2 have a 50% probability of transitioning in states $S2$ or $S3$. Observers only perceive sequences of observations, and an acting agent could follow one of the multiple legal trajectories. In our example, the sequence $\langle O1, O2 \rangle$ will not provide new information to an observer; she will not know whether the agent executed action a_1 or a_2, nor could she discern whether the agent transitions between states $S2$ and $S3$.

Design Model

The design model describes the characteristics of applicable modifications.

Definition 2 (Design Model)**.** *A design model in S-GRD is a tuple $\mathcal{D} = \langle \mathcal{M}, \delta, \phi, \mathcal{C}_m, k \rangle$ where: (1) $\mathcal{M}$ is a finite set of applicable modifications. A modification set is a combination of modifications $\vec{m} = \{m_1, m_2, \dots m_n\}$ with $m_i \in \mathcal{M}$. We refer to $\vec{\mathcal{M}}$ as the set of all those combinations. (2) $\delta : \mathcal{M} \times P \to P$ is a modification function, specifying the effect of modifications on the stochastic GR model. (3) $\phi : \vec{\mathcal{M}} \times P \to \{\bot, \top\}$ is a constraint function that specifies the allowable modification sets. (4) $\mathcal{C}_m : \mathcal{M} \to \mathbb{R}^+$ defines the cost $\mathcal{C}_m(m)$ to apply modification $m \in \mathcal{M}$ to a stochastic GR model. (5) k is a user-defined parameter that limits the size of a modification set.*

In this paper, $\mathcal{C}_m(m) = 1$ for all proposed modifications. Similar to previous work, the constraint function prevents changes that increase the original costs to reach each goal.

The resultant model after the correct application of a set of modifications is defined as:

Definition 3 (Application of a Set of Modifications)**.** *Given a stochastic goal recognition model P and a modification set $\vec{m} \in \vec{\mathcal{M}}$ such that $\vec{m} = \{m_1, m_2, \dots m_n\}$; $m_i \in \mathcal{M}$; and $\phi(\vec{m}) = \top$; the set $\vec{m}$ applied to P gives a new stochastic GR model $P^{\vec{m}} = \delta(m_n, \dots, \delta(m_1, P))$.*

Evaluating the Stochastic Goal Recognition Problem

GRD problems use design optimization to minimize ambiguous paths and consequently to facilitate GR. Therefore, they require a measure or criterion to assess the difficulty of performing GR in a given model. Traditionally, the measure used is called the worst-case distinctiveness (*wcd*). In this subsection, we redefine *wcd* for the SS-GRD problem.

Worst-Case Distinctiveness (*wcd*) Contrary to Keren, Gal, and Karpas (2019), we propose that the cost used should be the cost for the observer ($\mathcal{C}_o$ in Def. 1) as the optimization is for the observer's benefit. i.e., we reduce the ambiguity of the problem so that the observer infers the real goal sooner. At a high level, we propose the wcd for stochastic settingsAt a high level, we propose the *wcd* for stochastic settings corresponds to the highest expected cost or penalty an *observer* could experience while the *acting agent* does not reveal its goal. The argument is that observer and agent might have unrelated priorities. An agent could use a lot of energy executing one action or could execute several actions using the same power in total. However, as long as each action takes the same amount of time (non-durative), an observer would prefer shorter trajectories (i.e., fewer actions) regardless of the cost they have for the agent. This decision will affect the solution when both costs are different, i.e. when $\mathcal{C} \neq \mathcal{C}_o$.

Note that the formal definition of *wcd* for SS-GRD is the same as for POS-GRD (Wayllace et al. 2020). However, the components are different. Definitions 4 to 6 introduce the new elements.

Definition 4 (Agent's Strategies)**.** *Given the initial stochastic GR model $P_0 = \langle M, \mathbf{G}, u, \mathcal{N}, \mathcal{C}_o \rangle$ the agent's strategies are the set of all policies Π_g^u of MDP M for goal $g \in \mathbf{G}$ within the limits imposed by u.*

Definition 5 (Legal Policies)**.** *Given the agent's strategies Π_g^u for goal $g \in \mathbf{G}$, the set $\Pi_{\mathbf{G}} = \bigcup_{g \in \mathbf{G}} \Pi_g^u$ is the set of all legal policies of P for all possible goals.*

In other words, the set of *legal policies* contains every policy that an agent can generate for every candidate goal according to the limitations imposed by the model through the parameter u. In this paper, u controls the suboptimal policies allowed. Note that this definition subsumes the optimal case where the set of legal policies contains only *optimal* policies ($u = 0$).

The worst-case distinctiveness of a stochastic GR problem P is as defined by Eq. 2, where $\Pi_{\mathbf{G}}$ follows Definition 5 and the distinctiveness cost $DC(\vec{\tau})$ of a trajectory $\vec{\tau}$ uses the cost function for the observer $\mathcal{C}_o$, i.e., Eq. 3 becomes:

$$\max_{i \in \{0 \ldots n\} \text{ s.t. } |\mathbf{G}(ob(\langle s_0, \ldots, a_i, s_i \rangle))| > 1} \sum_{j=1}^{i} \mathcal{C}_o(s_{j-1}, a_j, s_j) \quad (4)$$

$DC(\vec{\tau})$ is well-defined for proper policies, and DC is 0 for an empty trajectory ($i = 0$).

Definition 6 (Expected Distinctiveness)**.** *The* expected distinctiveness $ED(\hat{\pi})$ *of a policy $\hat{\pi}$ is the expected distinctiveness cost of its trajectories, $\sum_{\vec{\tau}} P_{\hat{\pi}}(\vec{\tau}) DC(\vec{\tau})$. Where $DC(\vec{\tau})$ is given by Eq. 4*

Objective of SS-GRD

An SS-GRD problem is a tuple $T = \langle P_0, \mathcal{D} \rangle$ where P_0 is the *initial stochastic GR* model and $\mathcal{D}$ is the *design model*, which specifies the rules to generate alternative stochastic GR models P by applying modification sets to P_0. The objective of an SS-GRD problem is to find a set of modifications $\vec{m} = \{m_1 \ldots m_n\}$, such that $\vec{m}$ is feasible (i.e., $\phi(\vec{m}) = \top \wedge |\vec{m}| \leq k$), and which minimizes the *wcd* of the resulting model $P_0^{\Delta} := (P_0^{m_1})^{\cdots m_n}$. That is:

$$\vec{m}^o = \underset{\vec{m} \in \vec{\mathcal{M}}: \phi(\vec{m}) = \top \wedge |\vec{m}| \leq k}{\operatorname{argmin}} wcd(P_0)$$

Solving SS-GRD

Bounded Suboptimality

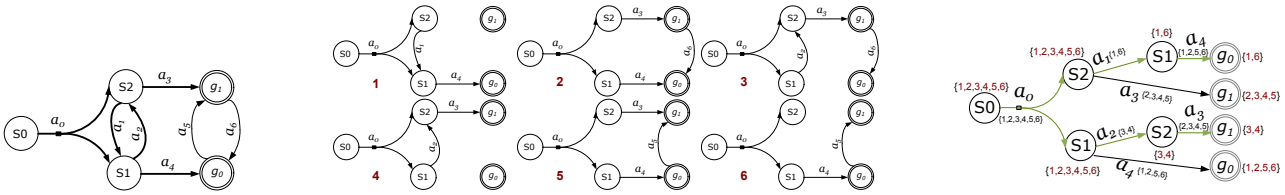
In multiple real-world scenarios, the agent may not act optimally, even if it is rational. Moreover, the effectiveness of GRD depends on accurately modeling the agent's interaction with the environment. Assuming an utterly irrational agent is usually not practical since it is unlikely that an agent with a goal or purpose presents such behavior. Additionally, erratic behaviors might cause infinite *wcd*. Therefore, we decided to consider a boundedly rational agent. Initially, we tried to find a set of top-k suboptimal policies that acting agents could take (Dai and Goldsmith 2009) where the next best policy differs from the previous one in only one state. However, as Dai and Goldsmith (2010) mention, there are usually multiple "trivially extended policies" that differ from another only in a non-reachable state. Additionally, the tie-breaking rule uses a lexicographic order, which sometimes excludes policies with the same expected cost. Therefore, even if it is computationally more demanding, we decided to find all policies with up to u suboptimal actions.[1] A newly generated policy differs from the previous one in *one* of its *reachable states*. Since generating legal (suboptimal) policies could use different methods, and due to space constraints, we present the pseudocode of the proposed algorithm in the appendix. After running this algorithm, all actions to reach goal g carry information of the legal policies that use them.

Computing *wcd*

In GRD problems for stochastic environments, the set of possible goals for a particular state depends on the observed path of the agent to that state (Wayllace, Hou, and Yeoh 2017; Wayllace et al. 2020). Therefore, the original MDP is augmented with goal information to represent all possible observations that all allowed trajectories may generate. The *wcd* is equal to the largest expected cost at the initial state of such augmented MDP.

When considering suboptimal policies, we cannot use the same method. To illustrate the problem, consider Fig. 2(a), where the start state is s_0 and there are two possible goals: g_0 and g_1. All actions are deterministic except for action a_0, which has 50% probability to transition from s_0 to states s_1 or s_2. The cost of executing an action is 1. Bold arrows represent all non-distinctive actions, which, together with all states, match the structure of the augmented MDP as defined for S-GRD (Wayllace, Hou, and Yeoh 2017). Note that actions a_1 and a_2 form an infinite loop. Therefore, the largest expected cost of this augmented MDP at s_0 is infinite and will not provide the *wcd* value. Further, we require the agent

[1] The implementation also limits policies with expected costs larger than five times the optimal cost.



(a) (b) (c)

Figure 2: FO SS-GRD. (a) Original MDP. (b) All Legal Policies. (c) Policy-Aware Augmented MDP

to select proper and stationary policies, that is, a policy containing actions a_1 and a_2 is not legal. Therefore, the *wcd* computation should not consider that loop.

Intuitively, keeping track of the policies breaks infinite loops. However, augmenting states with policy IDs may create maximal augmented policies that do not correspond to any legal policy. For instance, Fig. 2(c) shows the resultant MDP of this example after augmenting states and actions with policy IDs (shown in red numbers in Fig. 2(b)). The policy marked in green highlights the largest augmented policy; note that each trajectory maps to different pairs of policy IDs ($\{1, 6\}$ and $\{3, 4\}$). Due to this problem, we should evaluate the non-distinctive part of a policy only in the augmented space *reachable* by that policy. By doing so, we are effectively computing its *expected distinctiveness* (Def. 6). To evaluate policy 1, for instance, we should not consider augmented states $\{\langle S2, \{3, 4\}\rangle, \langle g_1, \{3, 4\}\rangle, \langle g_1, \{2, 3, 4, 5\}\rangle\}$.

In case of partial observability (PO), the non-distinctive prefix of a policy depends not only on policies that share the same actions, but also on policies sharing the same *observable trajectories*. For instance, consider Fig. 1(a). When evaluating the ambiguity of policy formed by actions a_1 and a_3, there is no legal policy for goal g_1 sharing those actions. However, all other policies share part of its *observable trajectory* ($\vec{\tau} = \langle O1, O2\rangle$).

We now define the policy-aware augmented MDP for SS-GRD problems; later, we describe how to use it to find the *wcd* of the problem.

Policy-Aware Augmented MDP: Let $P = \langle M, \mathbf{G}, u, \mathcal{N}, \mathcal{C}_o\rangle$ with $u \geq 0$ be a GR model with stochastic action outcomes, $M = \langle \mathbf{S}, s_0, \mathbf{A}, \mathcal{T}, \mathcal{C}\rangle$ is an MDP with positive costs $\mathcal{C}$ for the agent and no goal. The states of an augmented MDP add a boolean variable pos^g_ρ per possible policy to keep track of its validity given an observed trajectory. Note that each policy with ID ρ also serves to keep track of possible goals since they are legal for a specific goal g. Terminal states of this MDP are those that have less than two possible goals. To comply with Eq. 4, the cost of an action transitioning to a terminal state is 0.

The policy-aware augmented MDP $\Pi_{aug} = \langle \mathbf{S}', s'_0, \mathbf{A}', \mathcal{T}', \mathcal{C}', \mathbf{G}'\rangle$ for SS-GRD is defined as follows:

- $\mathbf{S}' = \mathbf{S} \times \{\top, \bot\}^{|\Pi_\mathbf{G}|}$: for each $s \in \mathbf{S}$ we create $2^{|\Pi_\mathbf{G}|}$ augmented states, corresponding to all subsets of possible legal policies.
- $s'_0 = s_0 \cdot \langle \top \dots \top\rangle$: initially all policies are possible.
- $\mathbf{A}' = \mathbf{A}$ (action labels remain unchanged).
- $\mathcal{T}'(s \cdot \langle pos^1_1 \dots pos^g_\rho\rangle, a, s' \cdot \langle pos'^1_1 \dots pos'^g_\rho\rangle) =$

$$\begin{cases} \mathcal{T}(s,a,s') & (\exists i \neq j, b \neq d : pos^b_i = pos^d_j = \top) \wedge & (5)\\ & \forall i \in \{1 \dots, |\Pi_\mathbf{G}|\}(pos'^g_i = (pos^g_i \wedge & (6)\\ & \big(\pi_i \in \Pi^u_g \wedge \pi_i(s) = a \wedge id(\pi_i) = i\big) & (7)\\ & \vee(\mathcal{N}(s) = \mathcal{N}(s') & (8)\\ & \vee(\exists \pi \in \Pi_\mathbf{G} \wedge \exists \hat{s} : \mathcal{N}(s) = \mathcal{N}(\hat{s}) \wedge & (9)\\ & \exists \hat{s}' \mid \mathcal{T}(\hat{s}, \pi(\hat{s}), \hat{s}') > 0 \wedge & (10)\\ & \mathcal{N}(s') = \mathcal{N}(\hat{s}'))))) & (11)\\ 0 & \text{otherwise} & (12) \end{cases}$$

where $id : \Pi_\mathbf{G} \to \mathbb{Z}^+$ is a function mapping legal policies to policy IDs. The probability of transitioning from state s with $\langle pos^1_1 \dots pos^g_\rho\rangle$ to state s' with $\langle pos'^1_1 \dots pos'^g_\rho\rangle$ (where pos^b_i, pos'^b_i indicate whether policy with ID i, legal for goal b is possible) depends on multiple factors. The transition probability from a state where the true goal was revealed is 0 (Line 5). Once discarded, a policy cannot become possible (Line 6). A policy remains possible if action a is part of the legal policy with ID i (Line 7) or s and s' emit the same observation (Line 8). Finally, Lines 9-11 cover cases like action a_1 and a_2 in Fig. 1(a) discussed before.

- $\mathcal{C}'(s \cdot \langle pos^1_1 \dots pos^g_\rho\rangle, a, s' \cdot \langle pos'^1_1 \dots pos'^g_\rho\rangle) = \begin{cases} \mathcal{C}_o(s,a,s') & \forall(s' \cdot \langle pos'^1_1 \dots pos'^g_\rho\rangle) \notin \mathbf{G}' \\ 0 & \text{otherwise} \end{cases}$
  We want to find policies with maximal cost for the observer without including the cost of actions that transition to a terminal state.
- $\mathbf{G}' = \{s \cdot \langle pos^1_1 \dots pos^g_\rho\rangle \mid (\exists pos^b_i, \forall j \neq i, d \neq b : pos^d_j = \bot)\}$: terminal states are those with less than two possible goals.

Computing *wcd* - Practical Considerations: As stated before, the highest expected cost at the starting state of the policy-aware augmented MDP for SS-GRD is not always equivalent to the *wcd* of the problem. Therefore, we need to evaluate the non-distinctive prefix of every legal policy (using the augmented MDP). There is no need to generate all $|\mathbf{S}| \times 2^{|\Pi_\mathbf{G}|}$ augmented states, just the reachable states using the policy to be evaluated. In the PO case, we also need to generate states emitting the same observation sequences as the reachable states. For example, when evaluating the policy to reach goal g_0 in Fig. 1(a), there is no need to generate augmented states of g_1, but we should augment states $S2$ and

$S3$ since they share the observed sequence $\langle O1, O2\rangle$. While building this smaller augmented MDP, we can keep track of all policies that share all non-distinctive actions. If multiple policies share their maximum non-distinctive prefixes at the end, we group them, and there is no need to re-evaluate other policies in the group. The *wcd* is equivalent to the maximum expected cost among all groups. In the worst case, we will need to evaluate individually all $|\Pi_{\mathbf{G}}|$ policies.

We present now the method used for PO settings, which accounts for the observer's partial observability as previously defined[2]. Procedure augMDP-PO finds groups of policies sharing all their observable non-distinctive trajectories and builds a partial augmented MDP to evaluate them. In this procedure, $\mathbf{N}$ is the set of *observation sets* (Definition 1). Function $\mathcal{O} : \mathbf{N} \rightarrow 2^{\mathbf{S}'}$ is a mapping from $\mathbf{N}$ to the power set of augmented states that models augmented states grouped by their projected observations.

Procedure augMDP-PO($\rho, \mho, s_0, \mathbf{S}, \Pi^{\mho}_{\mathbf{G}}, \mathcal{T}, \mathcal{N}$)

1 $Stack \leftarrow \emptyset; \mathbf{S}' \leftarrow \emptyset; \mho_i \leftarrow \mho_o \leftarrow \mho, \mathcal{T}' \leftarrow null, \mathbf{G}' \leftarrow \emptyset; \mathcal{O} \leftarrow null$
2 $s_0^{\mho} \leftarrow \langle s_0, \mho\rangle$
3 $Stack.push(\{s_0^{\mho}\}); \mathbf{S}' \leftarrow \mathbf{S}' \cup \{s_0^{\mho}\}$
4 **while** $Stack \neq \emptyset$ **do**
5 $\quad \mathbf{S}_0 \leftarrow Stack.pop$
6 $\quad \langle \mho_i, \mho_o, \mathcal{T}', \mathbf{G}', \mathbf{S}', \mathcal{O}\rangle \leftarrow$ CREATENODE$(\mathbf{S}_0, \mathcal{T}, \mathcal{N}, \rho, \mho_i, \mho_o, \mathcal{T}', \mathbf{G}', \mathbf{S}')$
7 $\quad$ **foreach** $\mathbf{O} \in \mathbf{N} \mid \mathcal{O}(\mathbf{O}) \neq null$ **do**
8 $\quad\quad \mathbf{S}'' \leftarrow \mathcal{O}(\mathbf{O})$
9 $\quad\quad Stack \leftarrow \mathbf{S}''$
10 $\quad$ **end**
11 **end**
12 return $\langle \mho_i, \mathbf{S}', \mathcal{T}', \mathbf{G}'\rangle$

At a high level, the procedure represents observation sets as nodes, and traverses these nodes in a DFS fashion. Each node contains sets of *unobservably connected* states (Wayllace et al. 2020), where all states emit the same observation and share the set of possible goals. For example, in Fig. 1(b), each green blob denotes a node. The node projecting observation $O2_{01}$ has two sets of unobservably connected states, one formed by states $S2$ and $S3$, and the other by $S1$. Goals g_0 and g_1, marked as subindex of the observation, are still possible for states in this node.

The procedure receives as arguments the policy ID ρ of the policy to be evaluated, the set $\mho$ of all policy IDs, the initial state s_0, the set $\mathbf{S}$ of original states, the set $\Pi^{\mho}_{\mathbf{G}}$ of all legal policies marked with their respective IDs, the original transition function $\mathcal{T}$, and the sensor configuration modeled by the sensor function $\mathcal{N}$. First, all variables are initialized and the start state s_0 is augmented with the set of all policy IDs $\mho$ (Lines 1-2). Next, a set containing the augmented initial state is pushed to a stack and the set of augmented states is updated (Line 3). Each stack entry is a set of augmented states emitting the same observation and whose predecessors emit a *different* observation. While traversing unobservably connected sates from $\mathbf{S}_0$, Procedure CREATENODE updates and returns (1) augmented MDP components, (2) the function $\mathcal{O}$ mapping emitted observations to newly expanded and unexplored augmented states, (3) the set $\mho_o$ containing IDs of policies that share observable trajectories with the evaluated policy, and (4) the ID set $\mho_i$ of policies sharing actions with the policy of ID ρ (Lines 4-11). Finally, Procedure AUGMDP-PO returns the set $\mho_i$ of policy IDs and the components of an augmented MDP useful to evaluate the group of policies signaled by $\mho_i$ (Line 12).

[2]The appendix contains the procedure for FO settings

The solution of the augmented MDP generated using Procedure AUGMDP-PO gives the expected cost of the largest non-distinctive prefix of policies in $\Pi^{\mho}_i$, i.e., policies with IDs indicated by $\mho_i$. The *wcd* is then computed using:

$$wcd(P) = \max_{i=1\ldots n} V_{\Pi^{\mho}_i}(s'_0) \tag{13}$$

$$V_{\Pi^{\mho}_i}(s') = \sum_{s''\in\mathbf{S}'} \mathcal{T}'(s', \pi(s'), s'')[\mathcal{C}'(s', \pi(s'), s'') + V_{\Pi^{\mho}_i}(s'')] \tag{14}$$

where: $\pi \in \Pi^{\mho}_i$ and $\bigcup_{i=1}^{n} \mho_i = \mho \wedge \bigcap_{i=1}^{n} \mho_i = \emptyset$

We use a VI-based algorithm that runs iterations of Eq. 14 for groups of policies with the same non-distinctive trajectories, and find the maximum among all using Eq. 13. Since this evaluation is costly, we store all partial results in a max priority queue to minimize recomputation during design.

Design: Minimizing *wcd*

Keren, Gal, and Karpas (2018) cast the design model as a tree where the root represents the initial GR model P_0, and each children a modified model $P^m = \delta(m, P)$ with exactly one modification m applied to the parent P, i.e., the path from a node to the root gives set of applied modifications. The objective is to find the smallest modification set whose application yields *wcd* reduction. The naïve approach to solve the problem consists of traversing the tree using BFS and computing *wcd* for each node. In this subsection, we (1) present two types of modifications, (2) characterize their properties to prune the search space, and (3) describe algorithms for faster *wcd* recomputation.

Action Removal (AR): Action removal consists of removing state-action pairs from the original MDP. It was first proposed by (Keren, Gal, and Karpas 2014) as a method to modify GRD models. Similar to Wayllace et al. (2020), we (1) detect sets of actions that cause unreachable goals with legal policies and prune any superset of those actions, and (2) prune actions used only by policies with distinctive prefixes. Further, removing an action removes policies that contain it, and since AR should not increase original costs, the *wcd* never increases (formal proofs in the appendix).

The algorithm for faster *wcd* recomputation uses a max priority queue Q created when computing *wcd* for the initial GR problem. Q contains groups of policies evaluated together and their expected distinctiveness (ED) as the key value. The high-level idea is to reevaluate only dequeued groups of removed policies or those containing a single policy (since their ED will not change otherwise). In addition,

we stop when the current *wcd* is higher than the top value in Q (as *wcd* does not increase).

Sensor Refinement (SR): Sensor refinement refines a single state to make it fully observable. This type of modification was proposed by Wayllace et al. (2020) for POS-GRD problems. While we have the same objective, our method takes advantage of the following two properties (detailed in the appendix): (1) SR does not increase the ED of any legal policy; and (2) Refining a state s can only affect the ED of policies that reach states s' projecting the same observation ($s'|\mathcal{N}(s) = \mathcal{N}(s')$). Further, we leverage the fact that the best solution is a fully-refined model. The algorithm recomputes the *wcd* for a fully-refined problem and, similar to POS-GRD, it refines all states within a single observation set and if the *wcd* did not reduce, it prunes all subsets with more than one element of that observation set. Faster *wcd* recomputation uses the max priority queue Q created at the time of computing *wcd* for the initial GR problem. The main idea is to recompute the ED of policies at the top of Q until we find a group of policies whose new ED is larger than the next value in Q. There is no need of further evaluation since it is guaranteed that *wcd* will not increase after refinement.

Note that the design performed may not be helpful if agents do not execute policies that contribute to the *wcd* or when the agent's model is incorrect. Considering slightly suboptimal policies improves the robustness of the solution in certain cases where the latter is true. For instance, in Fig. 2(a), a system assuming optimal agents would remove no action. In contrast, considering suboptimal agents will guide the design to remove actions a_6 and a_7. There is no problem if the acting agent is effectively optimal in this case. However, action removal could help the observer infer the correct goal faster if the agent's model was inaccurate. In the original case, the $wcd = 1.5$ whereas after removing actions a_6 and a_7, we obtain $wcd = 1.1$.

Empirical Evaluation

The objective of this section is to evaluate the usefulness and scalability of our methods. We describe the settings used and present and analyze the experimental results.

Data: We evaluated our approach on five modified planning domains: (1) GRID-NAVIGATION, (2) ROOM, (3) BLOCKSWORLD, (4) BOXWORLD, and (5) ATTACK-PLANNING (further details in the appendix).

Settings: We ran experiments in 39 instances with 36 different configurations per instance, evaluating action removal in FO and PO settings (FO-AR and PO-AR), and sensor refinement in PO settings (PO-SR). We used a budget *k* of up to 3 modifications and allowed up to 2 suboptimal actions ($u = 1$ and $u = 2$). Experiments were conducted on a 2.10 GHz machine with 16 GB of RAM and a timeout of 52 hours. The number of reachable states varies from 16 to 16,384, with 12 to 527,866 legal policies. The source code is available at https://github.com/cwayllace/SS-GRD.

Results: One way to evaluate the success of (POS-)GRD is through *wcd* reduction of the modified problems. Intuitively, if *wcd* reduces, an observer has a higher probability of recognizing the true goal earlier. Fig. 3 visualizes the *wcd* reduction with different budgets. Markers map instances (horizontal axes) to their corresponding *wcd* value (vertical axes). Blue marks represent the *wcd* value for the initial GR problem, red, yellow, and green markers denote the final *wcd* values for budgets of $k = 1, k = 2$, and $k = 3$ respectively. In PO-SR settings, pink markers represent *wcd* values for a fully-refined model. Comparisons should happen among markers in the same vertical line; the lower the mark, the smaller the *wcd* value. The top row shows values for instances with up to one suboptimal action ($u = 1$) and the bottom row for instances with up to two suboptimal actions ($u = 2$). As expected, the original *wcd* values increase with the number of suboptimal policies in all but five cases. Also, in most cases, the larger the budget, the higher the reduction.

Due to the branching factor of the ROOM domain, (up to 6 successors per action), almost every policy reaches all states, which creates vast augmented state spaces. Thus, only the smallest instance finished the *wcd* computation on time and so, we do not show results for ROOM.

We now present detailed observations per setting.

FO-AR: When considering $u = 1$, *wcd* reduced in 20 instances with budgets of $k = 1$ and $k = 2$, and in 23 instances with $k = 3$. However, the reduction was higher with larger budgets: 18 instances present a higher reduction with $k = 2$ and 11 instances with $k = 3$. For problems with $u = 2$, *wcd* reduced in 16 instances with a budget of $k = 1$ and in 19 instances with $k = 2$ and $k = 3$. The reduction amount increased in 14 instances from $k = 1$ to $k = 2$, and in 7 instances from $k = 2$ to $k = 3$.

PO-AR: Instances with $u = 1$ present a *wcd* reduction in 22 cases when $k = 1$, in 26 when $k = 2$, and in 22 when $k = 3$. The reduction amount increased in 18 cases from $k = 1$ to $k = 2$, and in 16 cases from $k = 2$ to $k = 3$. When considering $u = 2$, we observe a *wcd* reduction in 15 instances with $k = 1$, in 20 when $k = 2$, and in 17 when $k = 3$. A lower number of instances with a budget of $k = 3$ reduced their *wcd* value because some of them did not finish on time. The reduction amount increased in 16 cases from $k = 1$ to $k = 2$, and in 5 cases from $k = 2$ to $k = 3$.

PO-SR: In this setting, the optimal *wcd* value is equal to the *wcd* of a fully-refined GR problem. Instances with $u = 1$ present a *wcd* reduction in 18 cases when $k = 1$ (12 of which reached optimal values), in 18 cases when $k = 2$ (13 with optimal values), and in 17 cases when $k = 3$ (14 of them with optimal values). The *wcd* reduction was higher in 6 cases when comparing $k = 1$ to $k = 2$, and in 2 cases when using $k = 3$ instead of $k = 2$. When working with $u = 2$, 11 instances reduced *wcd* when $k = 1$ (6 of which were optimal), 15 reduced when $k = 2$ (8 with optimal values), and 14 when $k = 3$ (10 of them with optimal values). When comparing the amount of reduction, 8 instances had higher *wcd* reduction when using $k = 2$ instead of $k = 1$ and 5 when using $k = 3$ rather than $k = 2$.

We compared the running times of our algorithms against a naïve approach (no pruning and no faster *wcd* recomputation) and the running times within settings. Fig. 4 summarizes the tendency of running time (vertical axis, logarithmic scale in seconds) of instances in the GRID-NAVIGATION do-

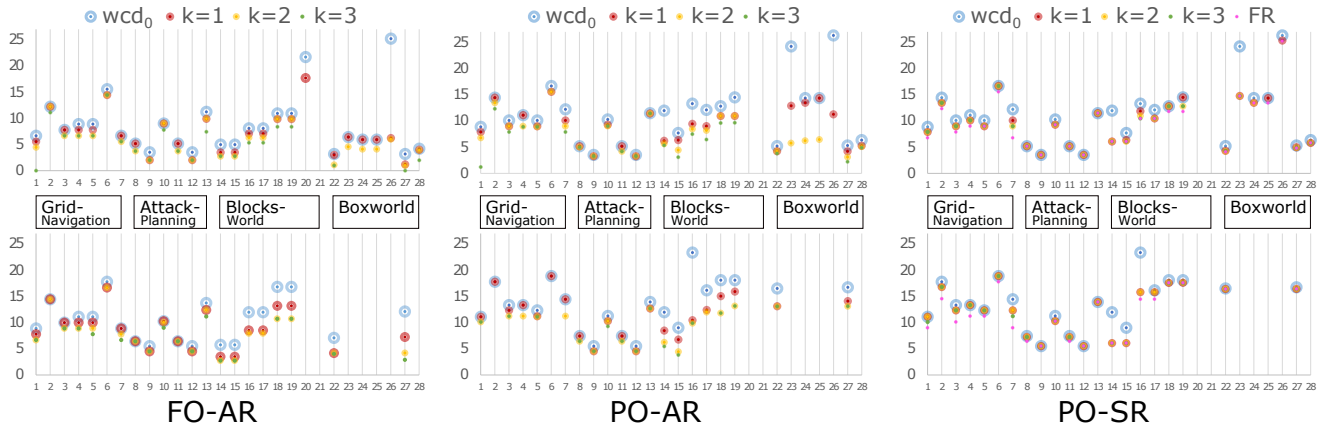


Figure 3: *wcd* Reduction in FO and PO Settings. $u = 1$ (Top Row), $u = 2$ (Bottom Row).

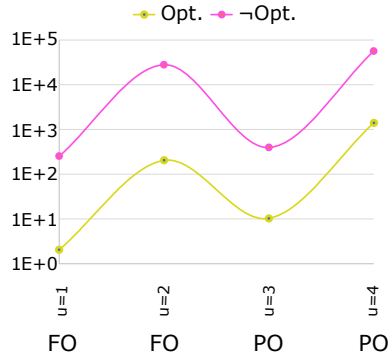


Figure 4: Average Running Time in sec. (Logarithmic Scale) for Instances of GRID-NAVIGATION.

main for FO and PO settings using AR and $k = 1$. Lines only allow us to connect the values for the same algorithm visually. Greenish markers plot the average values obtained for the optimized versions. In general, settings with a higher number of suboptimal actions ($u = 2$) require more time to find a solution since they handle the highest number of legal policies. However, this could change if policy evaluation is done in parallel. Finally, optimized versions outperform the naïve approach in 100% of the cases, with differences of up to six orders of magnitude (tabulated data in the appendix).

Conclusion and Future Work

We presented the Suboptimal Stochastic Goal Recognition Design (SS-GRD) problem, which assumes boundedly rational agents allowed to deviate from their optimal behavior by executing a limited number of suboptimal actions. Action outcomes are stochastic and observers could suffer from partial observability, in which case, actions and some states are not distinguished. The objective is to find the minimum number of modifications such that, once applied to the environment, the agent is forced to reveal its true goal earlier.

This paper formally defines the problem, adapts the *worst-case distinctiveness* (*wcd*) measure defined for POS-GRD, and provides and evaluates novel algorithms in several benchmark applications. Our analysis shows that to find the *wcd* in SS-GRD we need to keep track of the followed policy. Hence, an exact algorithm cannot avoid policy enumeration, which affects the scalability of the solution. Domains with a high branching factor and multiple loops such as ROOM are problematic for this strategy.

While our algorithms outperform the naïve approach, we believe other measures either with aggregated policy costs or independent of visited states would work better in stochastic settings. For instance, defining a measure that considers only the current state and its relation with the start state and possible goals could improve scalability, i.e., analyzing policy suffixes instead of prefixes (Masters 2019).

Considering other assumptions such as dynamic environments or incomplete agent (or environment) models also pose new challenges. Since the problem's complexity increases with the number of relaxed assumptions, it would be interesting to investigate approximate solutions in real-world scenarios.

References

Bellman, R. 1957. *Dynamic Programming*. Princeton University Press.

Cohen, P. R.; Perrault, C. R.; and Allen, J. F. 1981. Beyond Question Answering. In *Strategies for Natural Language Processing*, 245–274. Lawrence Erlbaum Associates.

Dai, P.; and Goldsmith, J. 2009. Finding Best k Policies. In *Proceedings of the International Conference on Algorithmic Decision Theory (ADT)*, 144–155.

Dai, P.; and Goldsmith, J. 2010. Ranking Policies in Discrete Markov Decision Processes. *Annals of Mathematics and Artificial Intelligence*, 59(1): 107–123.

Dai, P.; Weld, D. S.; Goldsmith, J.; et al. 2011. Topological Value Iteration Algorithms. *Journal of Artificial Intelligence Research (JAIR)*, 42: 181–209.

Keren, S.; Gal, A.; and Karpas, E. 2014. Goal Recognition Design. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, 154–162.

Keren, S.; Gal, A.; and Karpas, E. 2015. Goal Recognition Design for Non-Optimal Agents. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 3298–3304.

Keren, S.; Gal, A.; and Karpas, E. 2016a. Goal Recognition Design With Non-Observable Actions. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 3152–3158.

Keren, S.; Gal, A.; and Karpas, E. 2016b. Privacy Preserving Plans in Partially Observable Environments. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 3170–3176.

Keren, S.; Gal, A.; and Karpas, E. 2018. Strong Stubborn Sets for Efficient Goal Recognition Design. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 141–149.

Keren, S.; Gal, A.; and Karpas, E. 2019. Goal recognition Design in Deterministic Environments. *Journal of Artificial Intelligence Research (JAIR)*, 65: 209–269.

Keren, S.; Gal, A.; and Karpas, E. 2021. Goal recognition design-survey. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 4847–4853.

Keren, S.; Xu, H.; Kwapong, K.; and Parkes, D. 2020. Information Shaping for Enhanced Goal Recognition of Partially-Informed Agents. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 9908–9915.

Masters, P. 2019. *Goal Recognition and Deception in Path-Planning*. Ph.D. thesis, RMIT University.

Mausam; and Kolobov, A. 2012. *Planning with Markov Decision Processes: An AI Perspective*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Ramírez, M.; and Geffner, H. 2010. Probabilistic Plan Recognition Using Off-the-Shelf Classical Planners. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 1121–1126.

Riabov, A. V.; Araghi, S. S.; and Udrea, O. 2020. Plan recognition with unreliable observations. US Patent App. 16/670,098.

Rosenfeld, A.; and Kraus, S. 2018. Predicting Human Decision-Making: From Prediction to Action. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(1): 1–150.

Sohrabi, S.; Riabov, A. V.; and Udrea, O. 2016. Plan Recognition as Planning Revisited. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 3258–3264.

Son, T. C.; Sabuncu, O.; Schulz-Hanke, C.; Schaub, T.; and Yeoh, W. 2016. Solving Goal Recognition Design using ASP. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Sukthankar, G.; Geib, C.; Bui, H. H.; Pynadath, D.; and P Goldman, R. 2014. *Plan, activity, and intent recognition: Theory and practice*. Newnes.

Wayllace, C.; Hou, P.; and Yeoh, W. 2017. New Metrics and Algorithms for Stochastic Goal Recognition Design Problems. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 4455–4462.

Wayllace, C.; Hou, P.; Yeoh, W.; and Son, T. C. 2016. Goal Recognition Design with Stochastic Agent Action Outcomes. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 3279–3285.

Wayllace, C.; Keren, S.; Gal, A.; Karpas, E.; Yeoh, W.; and Zilberstein, S. 2020. Accounting for Observer's Partial Observability in Stochastic Goal Recognition Design: Messing with the Marauder's Map. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, 2394–2401.