A Unified Framework for Planning in Adversarial and Cooperative Environments

Anagha Kulkarni, Siddharth Srivastava, Subbarao Kambhampati
School of Computing, Informatics, and Decision Systems Engineering
Arizona State University, Tempe, AZ 85281 USA
{anaghak, siddharths, rao} @ asu.edu

Abstract

Users of AI systems may rely upon them to produce plans for achieving desired objectives. Such AI systems should be able to compute obfuscated plans whose execution in adversarial situations protects privacy, as well as legible plans which are easy for team members to understand in cooperative situations. We develop a unified framework that addresses these dual problems by computing plans with a desired level of comprehensibility from the point of view of a partially informed observer. For adversarial settings, our approach produces obfuscated plans with observations that are consistent with *at least k* goals from a set of decoy goals. By slightly varying our framework, we present an approach for producing legible plans in cooperative settings such that the observation sequence projected by the plan is consistent with *at most j* goals from a set of confounding goals. In addition, we show how the observability of the observer can be controlled to either obfuscate or convey the actions in a plan when the goal is known to the observer. We present theoretical results on the complexity analysis of our approach. We also present an empirical evaluation to show the feasibility and usefulness of our approaches using IPC domains.

1 Introduction

AI systems have become quite ubiquitous. As users, we heavily rely on these systems to plan our day-to-day activities. Since all these systems have logging and tracking abilities, an observer can get access to our data and our actions. Such observers can be of two types: adversarial or cooperative. In adversarial settings, like mission planning, military intelligence, reconnaissance, etc., protection of sensitive data can be of utmost importance to the agent. In such situations, it is necessary for an AI system to produce plans that reveal neither the intentions nor the activities of the agent. On the other hand, in case of a cooperative observer, the AI system should be able to produce plans that help convey its intent to the observer. Therefore, it is desirable for an AI system to be capable of computing both obfuscated plans in adversarial settings and legible plans in cooperative settings.

In this work, we propose a new unifying formalization, and algorithms for computing obfuscated plans as well as legible plans. In our framework, we consider two agents:

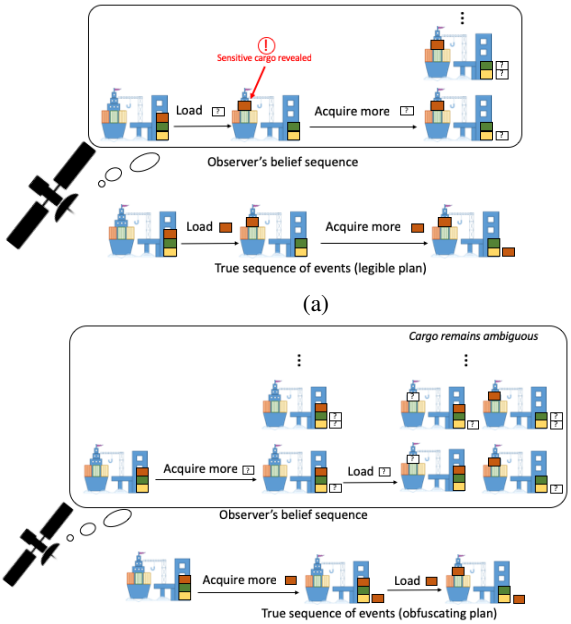

Figure 1: The differences in belief sequences induced by different plans for an observer with noisy sensors.

an acting agent and an observer. The acting agent has full observability of its activities. The observer is aware of the agent's planning model but has partial observability of the agent's activities. The observations are emitted as a side effect of the agent's activities and are received by the observer. An adversarial observer may be able to use the information gleaned from observations to interfere with or hamper the agent's activities. On the other hand, in cooperative scenarios, an agent is required to communicate its intentions to the observer as quickly and clearly as possible.

Example Consider a situation where the agent is a port management system and the observer has sensors or informants at the port who provide partial information about the nature of activity being carried out at the port (refer Figure 1). For instance, when a specific crate is loaded onto the ship, the observer finds out that something was loaded,

but not the identity of the loaded crate. The observer knows the initial inventory at the port, but when new cargo is acquired by the port, the observer's sensors reveal only that more cargo was received; they do not specify the numbers or identities of the received crates. A legible plan for loading sensitive cargo (the red crate) and acquiring more cargo may first load the crate and then acquire more crates. This plan reveals the identity of the crate that was loaded based on the observers' information about the remaining cargo in the port: the final belief state has a unique crate loaded on the ship even though it retains uncertainty about the new cargo in the port. However, if the plan were to first acquire more cargo, the observer's sensors are insufficient to determine which crate was loaded: the plan maintains ambiguity in the observer's belief. This is reflected in the observer's belief state sequence, where the last belief state includes states with all different types of crates in the ship. Although both plans have the same cost and effects for the dock, one obfuscates while the other conveys the activity being carried out. Our framework allows the agent to select plans that may be obfuscating or legible in this manner.

In this work, we introduce the problems of *goal obfuscation* and *goal legibility*. Our solutions to these problems use a common underlying algorithm with minimal variations. We also introduce two other problem variants where the observer knows the true goal of the agent and the agent wants to either obfuscate or reveal the actions in its plan. We call these problems *plan obfuscation* and *plan legibility* respectively. In the following sections, we present a common framework that encapsulates the planning problems discussed above. And thereafter, we discuss each of the problems in detail. We also provide a theoretical and empirical analysis of the value and scope of our approaches.

2 Controlled Observability Planning Problem

We assume an offline setting, such that the observer only receives the observations after the agent has finished executing a plan. In our setting, the agent has a set of candidate goals, inclusive of its true goal. The observer has access to the candidate goal set but is unaware of the agent's true goal. In the case of goal obfuscation, the objective is to generate a plan solution without revealing the true goal. Our solution ensures that *at least* k goals are possible at the end of the observation sequence. On the other hand, in the goal legibility problem the objective is to convey *at most* j goals to the observer. Our solution ensures that at most j goals are possible at the end of the observation sequence. In the case of the other two variants: plan obfuscation and plan legibility, the set of candidate goals consists of only one goal, which is the agent's true goal. For plan obfuscation, the objective is to generate a plan solution with an observation sequence that is consistent with at least ℓ diverse plans, such that, the diverse plans are at least d distance away. On the other hand, for plan legibility, the objective is to generate a plan solution that is consistent with at least m similar plans, such that, the similar plans are at most d distance away.

2.1 Classical Planning

A classical planning problem can be defined as a tuple $\mathcal{P} = \langle \mathcal{F}, \mathcal{A}, \mathcal{I}, G \rangle$, where $\mathcal{F}$, is a set of fluents, $\mathcal{A}$, is a set of actions. A state s of the world is an instantiation of all fluents in $\mathcal{F}$. Let $\mathcal{S}$ be the set of states. $\mathcal{I} \subset \mathcal{S}$ is the initial state. G is the goal where a subset of fluents in $\mathcal{F}$ are instantiated. Each action $a \in \mathcal{A}$ is a tuple of the form $\langle pre(a), add(a), delete(a), c(a) \rangle$ where $c(a)$ denotes the cost of an action, $pre(a) \subseteq \mathcal{F}$ is a set of preconditions for the action a, $add(a) \subseteq \mathcal{F}$ is a set of positive effects and $delete(a) \subseteq \mathcal{F}$ is a set of negative effects, i.e., $\Gamma(s, a) \models \perp$ if $s \not\models pre(a)$; else $\Gamma(s, a) \models s \cup add(a) \setminus delete(a)$ where $\Gamma(\cdot)$ is the transition function. The solution to $\mathcal{P}$ is a *plan* or a sequence of actions $\pi = \langle a_1, a_2, \ldots, a_n \rangle$, such that, $\Gamma(\mathcal{I}, \pi) \models G$, i.e., starting from the initial state sequentially executing the actions lands the agent in a goal state. The cost of the plan, $c(\pi)$, is summation of the cost of all the actions in the plan π, $c(\pi) = \sum_{a_i \in \pi} c(a_i)$.

2.2 Problem Setting

We now introduce a general planning problem framework that will be used to define the adversarial and the cooperative cases. The controlled observability problem involves an acting agent and an observer.

Definition 1. *A **controlled observability planning problem** is a tuple, $\mathcal{P}_{\mathcal{CO}} = \langle \mathcal{D}, \mathcal{G}, \Omega, \mathcal{O} \rangle$, where,*

- *$\mathcal{D} = \langle \mathcal{F}, \mathcal{A}, \mathcal{I} \rangle$ is the planning domain of the agent.*
- *$\mathcal{G} = \{G_1 \cup G_2 \ldots \cup G_{n-1} \cup G_A\}$ is a set of candidate goal conditions, each defined by subsets of fluent instantiations, where G_A is the true goal of the agent.*
- *$\Omega = \{o_i | i = 1, \ldots, m\}$ is a set of m observations that can be emitted as a result of the action taken and the state transition.*
- *$\mathcal{O} : (\mathcal{A} \times \mathcal{S}) \rightarrow \Omega$ is a many-to-one observation function which maps the action taken and the next state reached to an observation in Ω. That is to say, the observations are deterministic, each $\langle a, s' \rangle$ pair is associated with a single observation but multiple pairs can be mapped to the same observation.*

The observer has access to $\mathcal{P}_{\mathcal{CO}}$, but is unaware of the true goal of the agent. Also, the observer does not have access to the actions performed by the agent, instead receives the observations corresponding to the plan executed by the agent. The observation function can be seen as a sensor model, as modeled in several prior works (Geffner and Bonet 2013; Bonet and Geffner 2014; Keren, Gal, and Karpas 2016b). For every action taken by the agent and an associated state transition, the observer receives an observation. This observation might be consistent with multiple action-state pairs because of the many-to-one formulation of $\mathcal{O}$. Therefore, the observer operates in the belief space. The agent takes the belief space of the observer into account in its planning process, so as to control the observability of the observer.

2.3 Observer's Belief Space

The observer may use its observations of the agent's activity to maintain a *belief state*, or the set of possible states con-

sistent with the observations. We use $\hat{s}$ as a notational aid to denote a state that is a member of the belief state.

Definition 2. *The **initial belief**, b_0, induced by observation, o_0 is defined as, $b_0 = \{\hat{s}_0 \mid \mathcal{O}(\emptyset, s_0) = o_0 \wedge \mathcal{O}(\emptyset, \hat{s}_0) = o_0\}$.*

Whenever a new action is taken by the agent, and the state transition occurs, the observer's belief updates as follows:

Definition 3. *A **belief update**, b_{i+1} for belief b_i is defined as, $b_{i+1} = update(b_i, o_{i+1}) = \{\hat{s}_{i+1} \mid \exists \hat{a},\ \Gamma(\hat{s}_i, \hat{a}) \models \hat{s}_{i+1} \wedge \hat{s}_i \in b_i \wedge \mathcal{O}(\hat{a}, \hat{s}_{i+1}) = o_{i+1}\}$.*

A sequence of belief updates gives us the observer's belief sequence that is consistent with the observation sequence emitted by the agent.

Definition 4. *A **belief sequence** induced by a plan p starting at state s_0, BS(p, s_0), is defined as a sequence of beliefs $\langle b_o, b_1, \ldots, b_n\rangle$ such that there exist $o_0, o_1, o_2, \ldots, o_n \in \Omega$ where,*

- $o_i = \mathcal{O}(a_i, s_i)$
- $b_{i+1} = update(b_i, o_{i+1})$

The set of plans that are consistent with the belief sequence of a given plan are called as belief plan set.

Definition 5. *A **belief plan set**, BPS(p, s_0) = $\{p_1, \ldots, p_n\}$, induced by a plan p starting at s_0, is a set of plans that are formed by causally consistent chaining of state sequences in $BS(p, s_0)$, i.e., BPS(p, s_0) = $\{\langle \hat{s}_0, \hat{a}_1, \hat{s}_1, \ldots, \hat{s}_n\rangle \mid \forall\ \hat{a}_j,\ \hat{s}_{j-1} \models pre(\hat{a}_j) \wedge \hat{s}_{j-1} \in b_{j-1} \wedge \hat{s}_j \models \hat{s}_{j-1} \cup add(\hat{a}_j) \setminus delete(\hat{a}_j) \wedge \hat{s}_j \in b_j\}$.*

The agent's objective is to generate a belief sequence in observer's belief space, such that, the last belief in the sequence satisfies certain desired conditions.

2.4 Variants of $\mathcal{P}_{\mathcal{CO}}$

We now discuss the two major variants of $\mathcal{P}_{\mathcal{CO}}$ namely, goal obfuscation and goal legibility planning problems.

Goal Obfuscation In this setting, the adversary is unaware of agent's true goal. The aim of goal obfuscation is to hide the true goal from the observer. This is done by taking actions towards agent's true goal, such that, the corresponding observation sequence exploits the observer's belief space in order to be consistent with multiple goals.

Definition 6. *A **goal obfuscation planning problem**, is a $\mathcal{P}_{\mathcal{CO}}$, where, $\mathcal{G} = \{G_A \cup G_1 \cup \ldots \cup G_{n-1}\}$, is the set of n goals where G_A is the true goal of the agent, and $G_1, \ldots, G_{n-1}$ are decoy goals.*

A solution to a goal obfuscation planning problem is a *k-ambiguous* plan. The objective of a *k-ambiguous* plan is to make the observation sequence consistent with at least k goals, out of which $k-1$ are decoy goals, such that, $k \leq n$. These $k-1$ goals can be chosen to maximize the obfuscation. A *k-ambiguous* plan achieves at least k goals in the last belief of the observation sequence.

Definition 7. *A plan, π_k, is a **k-ambiguous plan**, if $\Gamma(\mathcal{I}, \pi_k) \models G_A$ and the last belief, $b_n \in BS(\pi_k, \mathcal{I})$, satisfies the following, $|G \in \mathcal{G} : \exists s \in b_n, s \models G| \geqslant k$, where $1 \leqslant k \leqslant n$.*

Definition 8. *An observation sequence $O_k = \langle o_1, \ldots, o_n\rangle$ is **k-ambiguous observation sequence** if it is an observation sequence emitted by a k-ambiguous plan.*

Goal Legibility The aim of goal legibility is to take actions exclusive to the goal so as to help the observer in goal deduction. This can be useful in cooperative scenarios where the agent wants to notify the observer about its goal without explicit communication. Here we generalize the notion of goal legibility with respect to j number of goals.

Definition 9. *A **goal legibility planning problem** is a $\mathcal{P}_{CO}$, where, $\mathcal{G} = \{G_A \cup G_1 \cup \ldots \cup G_{n-1}\}$ is the set of n goals where G_A is the true goal of the agent, and $G_1, \ldots, G_{n-1}$ are confounding goals.*

The objective here is to generate a legible plan so as to convey at most j goals. To that end, we ensure that the observation sequence of a legible plan is consistent with *at most* j goals so as to limit the number of goals in the observer's belief space.

Definition 10. *A plan, π_j, is a **j-legible plan**, if $\Gamma(\mathcal{I}, \pi_j) \models G_A$ and the last belief, $b_n \in BS(\pi_j, \mathcal{I})$, satisfies the following, $|G \in \mathcal{G} : \exists s \in b_n, s \models G| \leqslant j$, where $1 \leqslant j \leqslant n$.*

The definition of *j-legible* observation sequence follows that of *k-ambiguous* case.

Goal Diversity Note that, the solutions to the goal obfuscation and goal legibility planning problems can be expressed in terms of a goal diversity measure. A goal diversity measure can be used to establish the extent of diversity among the set of goals present in the observer's belief. For instance, a simple goal diversity measure could be the cardinality of the goals satisfied in the last belief state, which is the measure admitted by Definitions 7 and 10. However, other goal diversity measures can also be used to solve the problems of goal obfuscation and goal legibility, and then the *k-ambiguous* and *j-legible* solutions can be written as *at least k goal-diverse* and *at most j goal-diverse*.

2.5 Complexity Analysis

In this section, we discuss the complexity results for $\mathcal{P}_{\mathcal{CO}}$. Given the Definitions 7 and 10 of goal obfuscation and goal legibility plan solutions, we prove that the plan existence problem for $\mathcal{P}_{\mathcal{CO}}$ is EXPSPACE-complete.

Theorem 1. *The plan existence problem for a controlled observability planning problem is EXPSPACE-hard.*

Proof. To show that the plan existence problem for $\mathcal{P}_{\mathcal{CO}}$ is EXPSPACE-hard, we will show that the NOD (No-Observability Deterministic) planning problem is reducible to $\mathcal{P}_{\mathcal{CO}}$. The plan existence problem for NOD has been shown to be EXPSPACE-complete (Haslum and Jonsson 1999; Rintanen 2004).

Let $\mathcal{P}_N = \langle \mathcal{F}_N, \mathcal{A}_N, \mathcal{I}_N, G_N, \mathcal{V}\rangle$ be a NOD planning problem, where, $\mathcal{F}_N$ is the set of fluents (or Boolean state variables), such that, state s is an instantiation of $\mathcal{F}_N$. $\mathcal{A}_N$ is a set of actions, such that, when an action $a \in \mathcal{A}_N$ is applied to a state, s_i, a deterministic transition to the next state occurs, $\Gamma(s_i, a) \models s_{i+1}$. $\mathcal{I}_N$ and $G_N = \{\phi_{G_N}\}$ are Boolean

formulae that represent set of initial and goal states. $\mathcal{V} = \emptyset$ is the set of observable fluents. We are considering a NOD problem, therefore there are no observations. Since the underlying system state is unknown, the deterministic transition function does not reveal the hidden state. $\mathcal{P}_N$ can be expressed as a $\mathcal{P}_{\mathcal{CO}}$ problem, $\mathcal{P}_C = \langle \mathcal{D}_N, G_C, \Omega, \mathcal{O} \rangle$, where, $\mathcal{D}_N = \{\mathcal{F}_N, \mathcal{A}_N, \mathcal{I}_N\}$, such that $\mathcal{I}_N$ is a set of possible initial states, $G_C = \{\phi_{G_N}, \neg\phi_{G_N}\}$ is the set of goal states, $\Omega = \emptyset$ and $\mathcal{O} = \emptyset$. The goal set G_C consists of all states: the first goal formula is common with the NOD problem and the second one is the negation of it, which, in essence, encapsulates the rest of the states.

Suppose $\pi_{\mathcal{P}_C} = \langle a_1, \ldots, a_r \rangle$ is a 1-*ambiguous*/1-*legible* plan solution to $\mathcal{P}_C$, such that, $\Gamma(\mathcal{I}_C, \pi_{\mathcal{P}_C}) \models \phi_{G_N}$ and the last belief $b_r \in BS(\pi_{\mathcal{P}_C}, \mathcal{I}_C)$ satisfies $|G \in G_C : \exists \hat{s} \in b_r, \hat{s} \models G| = 1$. Then according to the definition of $\mathcal{P}_N$, the plan $\pi_{\mathcal{P}_C}$ satisfies the following, $\exists \hat{s}_r \in b_r, \hat{s}_r \models \phi_{G_N}$ and therefore solves $\mathcal{P}_N$.

Conversely, suppose $\pi_{\mathcal{P}_N} = \langle a_1, \ldots, a_q \rangle$ is a plan solution to $\mathcal{P}_N$, such that, $\Gamma(\mathcal{I}_N, \pi_{\mathcal{P}_N}) \models G_N$. Let b_q be the belief associated with the last action in $\pi_{\mathcal{P}_N}$. Since it achieves the goal, we can say that $\exists \hat{s}_q \in b_q, \hat{s}_q \models \phi_{G_N}$. According to Definitions 7, 10, for $k = j = 1$, b_q satisfies the condition. Therefore $\pi_{\mathcal{P}_N}$ is a solution to $\mathcal{P}_C$. □

Theorem 2. *The plan existence problem for a controlled observability planning problem is EXPSPACE-complete.*

Proof. In $\mathcal{P}_{\mathcal{CO}}$, the planner operates in belief space and the search space is bounded by $2^{2^{|\mathcal{F}|}}$, where $|\mathcal{F}|$ is the cardinality of the fluents (or Boolean state variables). If there exists a plan solution for $\mathcal{P}_{\mathcal{CO}}$, it must be bounded by $2^{2^{|\mathcal{F}|}}$ in length. Any solution longer in length must have loops, which can be removed. Therefore, by selecting actions nondeterministically, the solution can be found in at most $2^{2^{|\mathcal{F}|}}$ steps. Hence, the plan existence problem for $\mathcal{P}_{\mathcal{CO}}$ is in NEXPSPACE. By Savitch's theorem (Savitch 1970), NEXPSPACE = EXPSPACE. Therefore, the plan existence problem for $\mathcal{P}_{\mathcal{CO}}$ is EXPSPACE-complete. □

3 Computing Obfuscating and Legible Plans

Now we present a common algorithm template that will be used to compute plans for the problem variants.

3.1 Algorithm for Plan Computation

In our algorithm, each search node is represented by an approximate belief estimate, which is an arbitrary Δ-sized subset of the belief state. A search node, b_Δ, consists of state s and a partial belief update, $\tilde{b}$ (cardinality of $\tilde{b}$ is given by $\Delta - 1$, where Δ is a counter). We borrow the concept of "width" from Lipovetzky and Geffner (2012), but we consider the width in terms of cardinality of b_Δ. For example, when $\Delta = 1$, b_Δ only consists of the state s, when $\Delta = 2$, b_Δ consists of s, followed by a state from its belief such that the augmented b_Δ has cardinality of 2. The successor node uses the s in b_Δ to generate the successor state, and b_Δ to update its partial belief. There are two loops in the algorithm: the outer and the inner loop. The outer loop maintains the cardinality of b_Δ by incrementing the value of Δ in each iteration, such that value of Δ ranges from $1, 2, \ldots, |\mathcal{S}|$. In the inner loop, a heuristic-guided forward search (for instance, GBFS) can be used to search over space of belief states of cardinality Δ. These loops ensure the complete exploration of the belief space.

Proposition 1. *The algorithm necessarily terminates in finite number of $|\mathcal{S}|$ iterations, such that, the following conditions hold:*
*(**Completeness**) The algorithm explores the complete solution space of $\mathcal{P}_{\mathcal{CO}}$, that is, if there exists a $\pi_{\mathcal{P}_{\mathcal{CO}}}$ that correctly solves $\mathcal{P}_{\mathcal{CO}}$, it will be found.*
*(**Soundness**) The plan, $\pi_{\mathcal{P}_{\mathcal{CO}}}$, found by the algorithm correctly solves $\mathcal{P}_{\mathcal{CO}}$ as ensured by the corresponding goal-test.*

The algorithm terminates either when a plan is found or after running the outer loop for $|\mathcal{S}|$ iterations. The outer loop ensures that all the paths in the search space are explored. The goal tests of both of the problem variants ensure that the solutions are correct as per Definitions 7 and 10.

Optimization In order to speed up the search process, we perform an optimization on the aforementioned algorithm. For each search node, b_Δ, apart from the approximate belief estimate, we maintain the full belief update b consistent with a path to s. The approximate belief update b_Δ can be generated by choosing Δ-sized combinations of states from the complete belief. For example, when $\Delta = 1$, b_Δ only consists of the state s but still maintains full belief update b, when $\Delta = 2$, b_Δ consists of a new combination of approximate belief of size 2 derived from the maintained full belief. When $\Delta = 1$, because of the check for duplicate states in the closed list, only one path to the search node is explored. Therefore, the use of Δ allows the search process to explore multiple paths leading to a particular search node. The complete b helps in finding *k-ambiguous* and *j-legible* solutions faster at lower Δ values. We present the details of the optimization in Algorithm 1. In the following subsections, we show how we customize the goal-test (line 14) and the heuristic function (line 21) to suit the needs of each of our problem variants.

3.2 Computing Goal Obfuscated Plans

We randomly choose $\mathcal{G}_k$, a set of k goals (inclusive of true goal) from n goals in $\mathcal{G}$. In order to ensure goal obfuscation, we randomly pick a goal from this set and run Algorithm 1 with this goal as the true goal. Let's call it the decoy true goal. Apart from maintaining the belief, we also maintain the BPS (refer Definition 5) associated with the observation sequence. We then obtain the *k-ambiguous* observation sequence as output. Note that the plan output is discarded (because that plan only reaches the decoy true goal). Then we perform a post processing step, where we filter the BPS to retrieve a plan that reaches the agent's true goal. This process ensures that even if the adversary runs the algorithm again with either the same or different input (different goal as true goal, or different combination of k goals), the identity of the true goal remains *k-ambiguous*.

Algorithm 1: Plan Computation

Input: $\mathcal{P}_{\mathcal{CO}} = \langle \mathcal{D}, \mathcal{G}, \Omega, \mathcal{O} \rangle$
Output: plan solution $\pi_{\mathcal{P}_{\mathcal{CO}}}$, observation sequence, $O_{\mathcal{P}_{\mathcal{CO}}}$

Initialize *open*, *closed* and *unopened* lists
$\Delta \leftarrow 1$ ▷ Counter
$b_\Delta \leftarrow \{\mathcal{I}\}$ ▷ Initial search node
$b_0 \leftarrow \{\mathcal{O}(\emptyset, \mathcal{I})\}$ ▷ Initial belief
$open.push(\langle b_\Delta, b_0 \rangle, priority = 0)$
while $\Delta \leqslant |\mathcal{S}|$ **do**
  while $open \neq \emptyset$ **do**
    $b_\Delta, b, h(b_\Delta) \leftarrow open.pop()$
    if $|b_\Delta| \neq \Delta$ **then**
      $unopened.push(\langle b_\Delta, b \rangle, h(b_\Delta))$
      $continue$
    end
    $closed \leftarrow closed \cup b_\Delta$
    if $\langle b_\Delta, b \rangle \models$ GOAL-TEST$(\mathcal{G})$ **then**
      return $\pi_{\mathcal{P}_{\mathcal{CO}}}, O_{\mathcal{P}_{\mathcal{CO}}}$
    end
    for $s' \in successors(s)$ **do**
      $o \leftarrow \mathcal{O}(a, s')$
      $b' \leftarrow$ Belief-Generation(b, o)
      $b'_\Delta = \langle s', \tilde{b}' \rangle$ ▷ $\tilde{b}'$ of size Δ-1
      $h(b'_\Delta) \leftarrow$ HEURISTIC-FUNCTION(b'_Δ, b')
      add b'_Δ to *open* if not in *closed*
    end
  end
  $\Delta \leftarrow \Delta + 1$
  copy items of $unopened$ to *open*, empty $unopened$
end
procedure Belief-Generation(b, o)
$b' \leftarrow \{\}$
for $\hat{s} \in b$ **do**
  for $\hat{a} \in \mathcal{A}$ **do**
    if $\mathcal{O}(\hat{a}, \Gamma(\hat{s}, \hat{a})) = o$ **then**
      $b' \leftarrow b' \cup \Gamma(\hat{s}, \hat{a})$
    end
  end
end
return b'

Goal test We ensure that the solution plan does not end unless all k goals occur in the belief state. The goal condition checks whether the decoy true goal has been achieved in the agent's true state and also whether the $k-1$ goals have been achieved in the belief state. If there doesn't exist a plan that can achieve the chosen decoy true goal with the given k goals, we restart the search with next combination of k goals (inclusive of true goal) among the total n candidate goals. In the worst case, the value of k can be reduced until a solution is found.

Heuristic function We now propose a heuristic strategy for generating a plan where the last belief state satisfies k goals, making it a *k-ambiguous* plan. We use two heuristic functions to estimate a node's cost: $h(s) = h_{G_A^{decoy}}(s) + h_{\mathcal{G}_{k-1}}(b)$, where the first part computes the heuristic cost to the decoy true goal from the agent's actual state, and the second part computes the heuristic cost to $k-1$ goals from the belief induced by the emitted observation. This is computed by finding the *max* of heuristic cost to each of the $k-1$ goals. The heuristic value of a state is computed as the total of these two values. Thus the heuristic helps to guide the search towards at least k goals in the observer's belief.

In this setting, a perfectly rational adversary would be one that has unbounded computational resources, and can use them to make perfect inferences. In other words, that is the worst case scenario for our algorithm. The following proposition shows that our algorithm provides *k-ambiguous* guarantee even in the worst case, such that a perfectly rational adversary remains uncertain about the true goal.

Proposition 2. *Let $\mathcal{H}$ be a perfectly rational adversary. Given a $\mathcal{P}_{\mathcal{CO}}$ whose $\mathcal{O}$ function models the observation model of agent $\mathcal{R}$. Plans computed using Algorithm 1 with k-ambiguous goal test and heuristic function are such that, $\mathcal{H}$ will be able to infer the true goal with probability $\leqslant 1/k$.*

Let's say goal G_i from the set $\mathcal{G}_k$ is chosen randomly to be the decoy true goal. The observation sequence O_i is obtained by running the algorithm with $G_A = G_i$. The adversary can run the algorithm with each goal from $\mathcal{G}_k$ to search for G_i. But G_i can be the agent's true goal with probability $1/k$. Therefore the adversary can detect the true goal with probability $\leqslant 1/k$.

Note that the goals can be specified using sub-formulae. This makes the input specification much smaller depending on how many states are in the domain that are consistent with the sub-formulae. In the least constrained domain, that may be exponential in the number of propositions not used.

3.3 Computing Goal Legible Plans

In the case of goal legibility, we run the algorithm with the agent's true goal.

Goal test In order to ensure that the computed plan is consistent with *at most* j true goals, we change our goal condition to additionally check whether at most $j-1$ confounding goals have been achieved in the last belief in BS, or it can be interpreted as at least $n-j$ goals are absent in the belief.

Heuristic function In this case, our objective is to avoid at least $n-j$ goals, that is be consistent with at most j goals. We achieve this by minimizing the heuristic cost to the true goal from the agent's actual state and to the $j-1$ confounding goals from the agent's belief state. However, we maximize the heuristic cost to other $n-j$ goals in order to achieve at most j goals in the last belief state. This is written as, $h(s) = h_{G_A}(s) + h_{\mathcal{G}_{j-1}}(b) - h_{\mathcal{G}_{n-j}}(b)$.

4 Plan Obfuscation and Plan Legibility

We now discuss two other problem variants called plan obfuscation and plan legibility. These problems come into picture when the observer is aware of the agent's goal, and the

objective is to either obfuscate or convey the steps in the plan.

4.1 Obfuscation

We achieve plan obfuscation by computing a plan whose observation sequence conforms to a set of diverse plans, making it hard to predict the actions in the plan even when the goal is known to the observer.

Definition 11. *A **plan obfuscation planning problem** is a tuple, $\mathcal{P}_{PO} = \langle \mathcal{D}, \mathcal{G}_{PO}, \Omega, \mathcal{O} \rangle$, where, $\mathcal{G}_{PO} = \{G_A\}$, and G_A is the true goal of the agent.*

The solution to a plan obfuscation planning problem is an ℓ-*diverse* plan. An ℓ-*diverse* plan has an observation sequence that is consistent with at least ℓ plans that are at least d distance away. In order to compute an ℓ-*diverse* plan, we need to keep track of the plans that are consistent with the belief sequence, we use the belief plan set to obtain these plans. The diversity between plans can be enforced by using plan distance measures (Srivastava et al. 2007; Nguyen et al. 2012). A plan distance measure can be used to compute the diversity between all the pairs of plans in a belief plan set. Our approach can use any valid plan distance. We now define an ℓ-*diverse* plan.

Definition 12. *Two plans, p_1, p_2, are a **d-distant pair** with respect to distance function δ if, $\delta(p_1, p_2) = d$, where δ is a diversity measure.*

Definition 13. *A BPS induced by plan p starting at s_0 is **minimally d-distant**, $d_{min}(BPS(p, s_0))$, if $d = \min_{p1,p2 \in BPS(p,s_0)} \delta(p1, p2)$.*

Definition 14. *A plan, π_l, is an **ℓ-diverse plan**, if for a given value of d and distance function δ, $d_{min}(BPS(\pi_l, \mathcal{I})) \geq d$, $|BPS(\pi_l, \mathcal{I})| \geq \ell$, where $\ell \geq 2$ and every plan in $BPS(\pi_l, \mathcal{I})$ achieves the goal in $\mathcal{G}_{PO}$.*

The definition of an ℓ-*diverse observation sequence* follows that of *k-ambiguous* observation sequence.

Computing Obfuscated Plans Here we return a plan that is at least ℓ-*diverse* and that maximizes the plan distance between *BPS* induced by a plan.

Goal test To ensure the plans in BPS, induced by an ℓ-*diverse* plan, can achieve the goal in $\mathcal{G}_{\mathcal{PO}}$, we additionally check whether at least ℓ plans are reaching the goal or not and whether the minimum distance between plans in BPS is at least d. Also in order to ensure termination of the algorithm, there is a cost-bound given as input to the algorithm.

Heuristic function We now present our heuristic strategy to compute ℓ-*diverse* observation sequence: $h(s) = h_{G_A}(s) - d_{min}(BPS(p, s_0))$. Apart from minimizing the heuristic cost to the goal, we want to maximize the d of $d_{min}(BPS(p, s_0))$ induced by plan p starting at s_0. We do this in order to increase the minimum distance between the plan pairs. This distance is computed using a plan distance measure.

4.2 Plan Legibility

The definition of plan legibility planning problem, $\mathcal{P}_{PL}$, is similar to that of $\mathcal{P}_{PO}$. Here the objective is to convey the actions in a plan. We achieve this by computing a plan whose observation sequence is consistent with plans that are similar to each other. Again, a plan distance measure can be used to define the similarity of the plans. We call such a plan an m-*similar* plan. Essentially, the BPS induced by the m-*similar* plan consists of at least m plans to the goal, that are at most d distance apart. For similar plans in BPS, we define the maximum distance between any two pairs of plans.

Definition 15. *A BPS induced by plan p starting at s_0 is **maximally d-distant**, $d_{max}(BPS(p, s_0))$, if $d = \max_{p1,p2 \in BPS(p,s_0)} \delta(p1, p2)$.*

Definition 16. *A plan, π_m, is an **m-similar plan**, if for a given value of d and distance function δ, $d_{max}(BPS(\pi_m, \mathcal{I})) \leq d$, $|BPS(\pi_m, \mathcal{I})| \geq m$, where $m \geq 2$ and every plan in $BPS(\pi_m, \mathcal{I})$ achieves $\mathcal{G}_{PL}$.*

The definitions of *m-similar* observation sequence follows that of goal obfuscation case.

Computing Legible Plans Here we return a plan that is at least m-*similar* by minimizing the maximum plan distance between plan pairs in the *BPS* induced by a plan. The goal test for this case is similar to that of plan obfuscation case.

Heuristic function The heuristic function to compute m-*similar* observation sequence is similar to that of ℓ-*diverse*: $h(s) = h_{G_A}(s) + d_{max}(BPS(p, s_0))$. The only difference being, we want to minimize the d value of $d_{max}(BPS(p, s_0))$ induced by plan p starting at s_0.

5 Empirical Evaluation

We now present an empirical analysis of all four approaches. Through evaluation, we intend to investigate the following objectives **(1)** comparison between run time and plan costs across all four problems versus the optimal solution to the true goal, **(2)** cost overhead incurred by the decoy true goal randomization step in goal obfuscation problem, **(3)** impact of Δ in Algorithm 1 for goal obfuscation and legibility.

5.1 Domains and Experimental Setup

We use three IPC domains, namely `Blocksworld`, `Logistics` and `Driverlog` to evaluate our approach. For each of the domains, we randomly generated 50 problem instances. For the `Blocksworld` domain, we generated problems with 4 to 8 blocks and towers of maximum height 5 for both initial and goal states. After grounding, the smallest problem had 29 variables and 40 operators, and the largest problem had 89 variables and 144 operators. For the `Logistics` domain, we generated problems with goals consisting of 2 to 6 facts. After grounding, the smallest problem had 63 variables and 78 operators, and the largest problem had 63 variables and 198 operators. For the `Driverlog` domain, we generated problems with goals consisting of 2 to 6 facts. After grounding, the smallest problem had 33 variables and 100 operators, and the largest prob-

Domain	Metrics	*k-amb*	*j-leg*	*ℓ-div*	*m-sim*	opt
Blocksworld	time	196.9	213.68	140.88	100.96	0.99
	length	12.33	12.4	10.84	11.28	9.6
Logistics	time	206.13	247.9	100.04	45.05	8.31
	length	28.63	27.25	25.92	27.94	23.74
Driverlog	time	145.03	186.05	111.99	58.52	1.54
	length	14.33	13.5	13.59	12.95	11.16

Table 1: Empirical evaluation for all four variants using the optimization presented in Algorithm 1 versus the optimal plan solution (opt column) to the true goal. We report the average time (in seconds) and the average plan length.

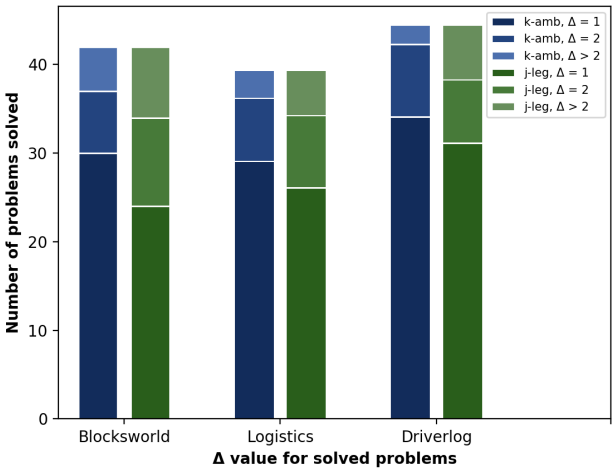

Figure 2: Empirical evaluation of Δ in Algorithm 1 for goal obfuscation and goal legibility variants. We report the number of problem instances solved for different values of Δ.

lem had 61 variables and 180 operators. We generated 5 random candidate goals (n=5) for each problem.

The partially observable models have many-to-one mapping of action-state pairs to observation symbols. For the sake of simplicity, we used lifted action names as observation symbols. The grounded actions taken and associated states are mapped to the corresponding lifted action names. For the `Blocksworld` domain, the observation symbols were *pickup, putdown, stack, unstack*. For the `Logistics` domain, the observation symbols were *load-truck, unload-truck, load-airplane, unload-airplane, drive-truck, fly-airplane*. Finally, for the `Driverlog` domain, the symbols were *load-truck, unload-truck, board-truck, disembark-truck, drive-truck, walk*.

5.2 Results

The evalution results are presented in Tables 1 and 2 and Figure 2. We modified the STRIPS planner Pyperplan (Alkhazraji et al. 2016) to implement our algorithms. We used the `hsa` (Keyder and Geffner 2008) heuristic of Pyperplan because it gave the best results in terms of computation time. We ran the experiments with $k = 3$, $j = 2$, $\ell = 3$, $m = 3$, $d_{min} = 0.25$ and $d_{max} = 0.50$ for all the domains. We used action distance measure to compute the distance between plans. We ran our experiments on 12 core Intel Xeon

Domain	*k-amb* w/o	*k-amb* w/	*k-amb* robust
Blocksworld	122.7	140.62	196.9
Logistics	159.28	181.74	206.13
Driverlog	112.93	123.02	145.03

Table 2: Empirical evaluation to report the average time (in seconds) for different versions of *k-ambiguous* algorithms. *k-amb w/* and *w/o* do not use BPS and report time with and without the additional post-processing step. *k-amb robust* uses BPS to provide robust solutions to replay attack.

CPU with an E5-2643 v3@3.40GHz processor with a 64G RAM with 20 minutes time-out. The following number of problems reached time-out before a solution could be found for goal legibility and obfuscation variants together: 8/50 in the `Blocksworld`, 11/50 in `Logistics` and 6/50 in `Driverlog`. These problems were excluded from the results to ensure consistency across problems.

The Table 1 presents a comparison between different variants of our framework. For *k-amb*, we randomize the decoy true goal and use BPS to retrieve the plan to the true goal. The process of maintaining BPS for every node is expensive and leads to higher run time, as can be seen in the Table. Another reason for higher run time is that the planner is working with multiple goals, as $k = 3$ in our experiments. Our approach is robust against replay attacks (attacks where the adversary tries running algorithm with different inputs). The *j-leg* solution also takes slightly longer time. This is because, we have to ensure that at most j goals are present, which can be translated as at least $n-j$ goals are absent. For *j-leg*, we used 3 goals, where at most 2 goals were allowed, and at least 1 was disallowed to be present in the solution. Thus both *k-amb* and *j-leg* use 3 goals to solve the problem. For *ℓ-div* and *m-sim*, the run time cost comes from maintaining BPS for each node. We also compare the time taken to compute an optimal plan to the true goal represented by the *opt* column.

The Table 2 shows performance comparison across different techniques of achieving *k-ambiguous* solutions. For *k-amb w/*, we do not maintain BPS for every node, instead we retrieve the plan to the true goal by filtering the belief space using the observation sequence. This can sometimes lead to incorrect solutions (because path consistency is not checked) but it can be computed faster than the solution that maintains BPS (*k-amb robust*). For *k-amb w/o*, we show the run times without the extra filtering step to retrieve the plan to the true goal, i.e., the run time for generation of k-ambiguous observation sequence. From Figure 2, we see that for both goal obfuscation and goal legibility, most of the solutions are obtained for $\Delta = 1$ for all the 3 domains.

6 Related Work

There are prior works which discuss the problem of privacy preservation in distributed multi-agent systems (Brafman 2015; Borrajo 2013; Bonisoli et al. 2014). A recent work on privacy for multi-agents by Maliah, Shani, and Stern (2016) is complementary to our approach, as they consider prob-

lems where the model needs to be protected from the team members but goals and behavior are coordinated. In contrast, we consider problems where the models are public but goals and behavior need to be protected.

The problem of goal obfuscation is also related to plan recognition literature (Ramırez and Geffner 2009; 2010; E-Martin, R-Moreno, and Smith 2015; Sohrabi, Riabov, and Udrea 2016; Keren, Gal, and Karpas 2016a). Traditional plan recognition systems have focused on scenarios where actions being executed can be observed directly. In our case, observational equivalence due to the many-to-one formulation of $\mathcal{O}$ introduces, in effect, noisy action-state observations. This, in turn, complicates plan recognition. More crucially, the agent uses the observational equivalence to actively help or hinder the ease of plan recognition.

In the recent years, the problem of goal recognition design (GRD) (Keren, Gal, and Karpas 2014; 2015; Wayllace et al. 2016; Wayllace, Hou, and Yeoh 2017) has received increasing attention. The GRD problem involves using environment changing actions to simplify the task of legibility. Such environment changing actions are a special class of actions that can change observability. If the set of actions available to the agent includes environment changing actions, our approach would automatically solve the problem of goal recognition design by selecting actions that maximize legibility. Additionally, our approach can also address the problem of goal obfuscation design.

A few recent works have explored the idea of obfuscation in adversarial settings from the goal recognition aspect (Keren, Gal, and Karpas 2016b; Masters and Sardina 2017). Keren, Gal, and Karpas (2016b) propose a solution that obfuscates a goal by choosing one of the candidate goals that have the maximum non-distinct observation sequence in common with the true goal. In contrast, our approach chooses k-sized combination of random goals (inclusive of the true goal), and if goal obfuscation is not possible for this combination, our approach exhaustively tries other combinations until a solution is found. In the worst case, the value of k is reduced until a solution can be found. Therefore our approach generalizes the former approach, and in addition, also provides guarantees against replay attacks. In addition, our formulation also supports the case of a cooperative observer by making the agent's intentions legible to the observer with respect to at most j goals. Finally, we support the problem of plan obfuscation and legibility which aid in obfuscating and conveying the actions in a plan.

In motion planning and robotics community, legibility (Dragan and Srinivasa 2013; Knepper et al. 2017) has been a well-studied topic. However, this has been mostly looked at from the motion planning perspective, and the focus has been on optimizing the motion trajectories to reveal the goal. We borrow this notion and generalize it in a unified framework to provide legible as well as obfuscated plans from a task planning perspective.

This work also has connections with human aware planning. Especially, the work on explicable planning and explanations (Zhang et al. 2017; Chakraborti et al. 2017; Kulkarni et al. 2018; Chakraborti, Sreedharan, and Kambhampati 2018) proposes modeling the human's understanding of a planning agent and introduces the notion of human-aware multi-model planning. Their framework consists of two planning models representing the planner's domain model and the observing or interacting human's understanding of the planning model. Essentially, their framework captures the expectations of the observer in the form of a possibly noisy or incomplete planning model of the agent. In contrast, our setting captures the observer's uncertainty over actor's activities using the sensor model. In future, we intend to investigate the connections between these two frameworks.

7 Conclusion

We introduced a unified framework that gives a planner the capability of addressing both adversarial and cooperative situations. Our setting assumes that the observer has partial visibility of the agent's activities, but is aware of agent's planning capabilities. We define four problems: goal obfuscation and legibility when the agent's true goal is unknown and, plan obfuscation and plan legibility when the agent's true goal is known. We propose the following solutions to these problems: *k-ambiguous* plan which obfuscates the true goal with respect to at least k goals, *j-legible* plan which enables an observer to quickly understand the j true goals of the agent, ℓ*-diverse* plan which obfuscates the actions in a plan and, *m-similar* plan which conveys the actions in the plan. We present different search techniques to achieve these solutions and evaluate the performance of our approaches.

This work opens the door to multiple interesting research questions. In our current framework, we look at settings that are either adversarial or cooperative. However, the real-world scenarios often consist of mixed agents, where some are of adversarial nature while some others are of cooperative nature. In such a multi-agent setting, our framework can be extended to increase the obfuscation for the adversarial agents while reducing the obfuscation for the cooperative ones. Another interesting avenue for future work involves modeling the observer as an active agent, such that, the observer is capable of performing actions in the environment to either thwart the agent from achieving its objectives in an adversarial setting, or to collaborate with the agent on dyadic tasks in a cooperative setting. This would involve extending the framework to support the observer's planning model.

Acknowledgments

We thank David Smith for helpful comments on the paper. This research is supported in part by the ONR grants N00014-16-1-2892, N00014-18-1-2442, N00014-18-1-2840, the AFOSR grant FA9550-18-1-0067, and the NASA grant NNX17AD06G.

References

Alkhazraji, Y.; Frorath, M.; Grützner, M.; Liebetraut, T.; Ortlieb, M.; Seipp, J.; Springenberg, T.; Stahl, P.; and Wülfing, J. 2016. Pyperplan. https://bitbucket.org/malte/pyperplan.

Bonet, B., and Geffner, H. 2014. Belief tracking for planning with sensing: Width, complexity and approximations. *Journal of Artificial Intelligence Research* 50:923–970.

Bonisoli, A.; Gerevini, A. E.; Saetti, A.; and Serina, I. 2014. A privacy-preserving model for the multi-agent propositional planning problem. In *Proceedings of the Twenty-first European Conference on Artificial Intelligence*, 973–974.

Borrajo, D. 2013. Multi-agent planning by plan reuse. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '13, 1141–1142. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.

Brafman, R. I. 2015. A privacy preserving algorithm for multi-agent planning and search. In *IJCAI*, 1530–1536.

Chakraborti, T.; Sreedharan, S.; Zhang, Y.; and Kambhampati, S. 2017. Plan explanations as model reconciliation: Moving beyond explanation as soliloquy. In *IJCAI*.

Chakraborti, T.; Sreedharan, S.; and Kambhampati, S. 2018. Explicability versus explanations in human-aware planning. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, 2180–2182. International Foundation for Autonomous Agents and Multiagent Systems.

Dragan, A., and Srinivasa, S. 2013. Generating legible motion. In *Proceedings of Robotics: Science and Systems*.

E-Martin, Y.; R-Moreno, M. D.; and Smith, D. E. 2015. A fast goal recognition technique based on interaction estimates. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Geffner, H., and Bonet, B. 2013. A concise introduction to models and methods for automated planning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 8(1):1–141.

Haslum, P., and Jonsson, P. 1999. Some results on the complexity of planning with incomplete information. In *European Conference on Planning*, 308–318. Springer.

Keren, S.; Gal, A.; and Karpas, E. 2014. Goal recognition design. In *ICAPS*.

Keren, S.; Gal, A.; and Karpas, E. 2015. Goal recognition design for non-optimal agents. In *AAAI*, 3298–3304.

Keren, S.; Gal, A.; and Karpas, E. 2016a. Goal recognition design with non-observable actions. In *AAAI*, 3152–3158.

Keren, S.; Gal, A.; and Karpas, E. 2016b. Privacy preserving plans in partially observable environments. In *IJCAI*, 3170–3176.

Keyder, E., and Geffner, H. 2008. Heuristics for planning with action costs revisited. In *ECAI*, 588–592.

Knepper, R. A.; Mavrogiannis, C. I.; Proft, J.; and Liang, C. 2017. Implicit communication in a joint action. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, 283–292. ACM.

Kulkarni, A.; Zha, Y.; Chakraborti, T.; Vadlamudi, S. G.; Zhang, Y.; and Kambhampati, S. 2018. Explicable robot planning as minimizing distance from expected behavior. In *ICAPS XAIP*.

Lipovetzky, N., and Geffner, H. 2012. Width and serialization of classical planning problems. In *Proceedings of the 20th European Conference on Artificial Intelligence*, ECAI'12, 540–545. Amsterdam, The Netherlands, The Netherlands: IOS Press.

Maliah, S.; Shani, G.; and Stern, R. 2016. Stronger privacy preserving projections for multi-agent planning. In *ICAPS*, 221–229.

Masters, P., and Sardina, S. 2017. Deceptive path-planning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 4368–4375.

Nguyen, T. A.; Do, M.; Gerevini, A. E.; Serina, I.; Srivastava, B.; and Kambhampati, S. 2012. Generating diverse plans to handle unknown and partially known user preferences. *Artificial Intelligence* 190(0):1 – 31.

Ramırez, M., and Geffner, H. 2009. Plan recognition as planning. In *Proceedings of the 21st international joint conference on Artifical intelligence. Morgan Kaufmann Publishers Inc*, 1778–1783.

Ramırez, M., and Geffner, H. 2010. Probabilistic plan recognition using off-the-shelf classical planners. In *Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI 2010)*.

Rintanen, J. 2004. Complexity of planning with partial observability. In *ICAPS*, 345–354.

Savitch, W. J. 1970. Relationships between nondeterministic and deterministic tape complexities. *Journal of computer and system sciences* 4(2):177–192.

Sohrabi, S.; Riabov, A. V.; and Udrea, O. 2016. Plan recognition as planning revisited. In *IJCAI*, 3258–3264.

Srivastava, B.; Nguyen, T. A.; Gerevini, A.; Kambhampati, S.; Do, M. B.; and Serina, I. 2007. Domain independent approaches for finding diverse plans. In *IJCAI*, 2016–2022.

Wayllace, C.; Hou, P.; Yeoh, W.; and Son, T. C. 2016. Goal recognition design with stochastic agent action outcomes. In *IJCAI*.

Wayllace, C.; Hou, P.; and Yeoh, W. 2017. New metrics and algorithms for stochastic goal recognition design problems. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 4455–4462. AAAI Press.

Zhang, Y.; Sreedharan, S.; Kulkarni, A.; Chakraborti, T.; Zhuo, H. H.; and Kambhampati, S. 2017. Plan explicability and predictability for robot task planning. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 1313–1320. IEEE.