Information Shaping for Enhanced Goal Recognition of Partially-Informed Agents

Abstract

We extend goal recognition design by considering a two-agent setting in which one agent, the *actor*, seeks to achieve a goal but has only partial information about its environment. The second agent, the *recognizer*, has perfect information and can selectively reveal information to the actor in order to recognize the actor's goal from its behavior as quickly as possible. The problem of selecting which information to reveal, which we call *information shaping*, is challenging because the space of information shaping options may be extremely large, and because more information revelation need not make an agent's goal easier to recognize. We formally define this problem, and suggest a pruning approach for efficiently searching the space of information shaping options. We demonstrate the ability to facilitate recognition via information shaping and the efficiency of the suggested method on a set of standard benchmarks.

Introduction

Goal recognition is the task of detecting the goal of agents by observing their behavior (Ramirez and Geffner 2009; 2010; Cohen *et al.* 1981; Kautz and Allen 1986; Carberry 2001; Sukthankar *et al.* 2014). We consider a two-agent goal recognition setting, where the first agent, the *actor*, has partial information about a deterministic environment and seeks to achieve a goal. The second agent, the *recognizer*, has perfect information, and tries to deduce the actor's goal as early as possible, by analyzing the actor's behavior.

As a one time offline intervention, and with the objective of facilitating the recognition task, the recognizer can apply a limited number of *information shaping* modifications, implemented as changes to the actor's sensor model. Such modifications can help to differentiate the actor's behavior for different goals, potentially making it easier to interpret.

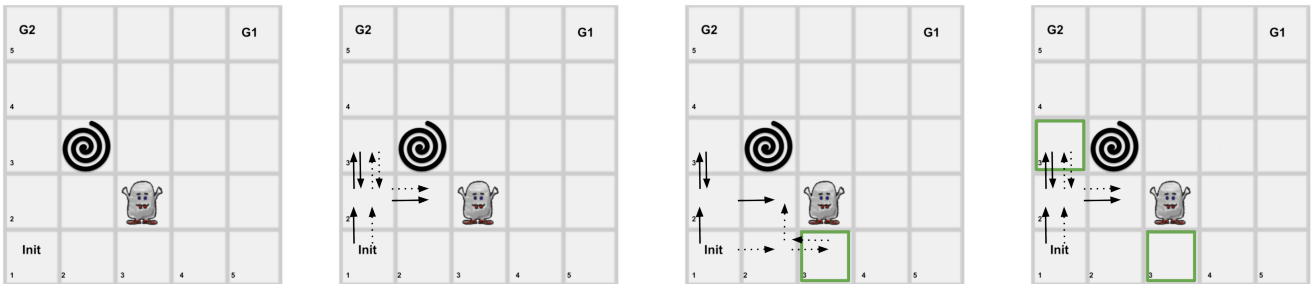
The ability to quickly understand what an agent is trying to achieve, without expecting it to explicitly communicate its objectives, is important in many applications. These include assistive cognition (Kautz *et al.* 2003), where it may be critical to know when a visually impaired user is approaching a hot oven, security applications, where a system tries to detect users aiming at a specific destination (Boddy *et al.* 2005), and human-robot collaborative settings (Levine and Williams 2014), where a robot aims to recognize what component a human user is trying to assemble, so it can gather the tools needed for the task.

Common to all these settings, is that agents have *incomplete information* about their environment. This affects their behavior and is key to the ability to interpret it. In addition, these settings can be controlled and modified in various ways. Specifically, it may be possible to modify an agent's behavior by manipulating its knowledge and its need to act in order to acquire new information. Such manipulations may induce behaviors that can be quickly associated to a specific goal. To demonstrate, in an assisted cognition setting, an auditory signal can inform users about a hot oven. Early notification potentially causes agents aiming at a different goal (e.g., open the cupboard) to move away from the oven, supporting early recognition of dangerous situations.

This work extends the *goal recognition design* (GRD) framework, which deals with redesigning agent settings in order to facilitate early goal detection (Keren *et al.* 2014; Wayllace *et al.* 2016). GRD work so far has assumed agents have perfect knowledge of their environment. This work is the first step in extending the framework to support agents with incomplete knowledge. Specifically, we focus on GRD in deterministic environments, and use contingent planning (Bonet and Geffner 2011; Brafman and Shani 2012b; Muise *et al.* 2014; Albore *et al.* 2009) to represent the actor. The design objective is to minimize the *worst case distinctiveness* (*wcd*) (Keren *et al.* 2014). *wcd* is the longest number of actions (or path cost) that is possible before the actor's goal is recognized. The task is to use information shaping to minimize *wcd*, where the information conveyed to the actor is required to be truthful and cannot mislead. Specifically, we use *sensor extensions*, that ease the access to the true value of some environment variable.

This is a challenging problem because the number of possible design options may be extremely large. Also, as we demonstrate below, more information need not make an agent's goal easier to recognize, and modifications need to be chosen carefully to promote early goal recognition.

Example 1 *As a simple example, consider Figure 1(a), depicting a variation of the well known Wumpus domain (Russell and Norvig 2016), where a partially informed actor is trying to achieve one of two goals (indicated by G_1 and G_2 in the image), without falling into pits or encountering a (deadly) wumpus. The actor knows its current position, but initially does not know the locations of the pits and wumpuses. When in a cell adjacent to a pit, it senses a 'breeze'.*



(a) The goal recognition setting

(b) Plans executed by an agent aiming at G_1 (dashed arrows) and G_2 (solid arrows)

(c) Plans executed when the recognizer reveals cell (3,1) is safe

(d) Plans executed when the recognizer reveals both cell (3,1) and cell (1,3) are safe.

Figure 1: An example of a GRD-APK problem

Equivalently, it can smell a Wumpus at an adjacent cell. The recognizer has perfect information: it knows the locations of the actor, the pit (the spiral at cell $(2,3)$*) and the wumpus (cell* $(3,2)$*).*

The actor starts at 'Init'. With no breeze or stench, it decuces the adjacent cells are safe*, with no wumpus or pit. We assume for the example that the actor is 'optimistic' (Bonet and Geffner 2011), i.e., it makes the most convenient assumptions about the values of unknown variables, plans accordingly, and revises the assumptions and replans, if its assumptions are refuted during execution. Specifically, we assume that among cost-minimal plans, it prefers one that requires making as few assumptions as possible. Consequently, an agent aiming at* G_2 *will start its progress by moving up. In contrast, an agent aiming at* G_1 *may go either up or right. This means that the choice to move up from the initial state, leaves the goal unrecognized. In fact, as depicted in Figure 1(b), the goal of the agent remains unrecognized even after the agent terminates execution (halting at cell (2,2) with no cells to explore).*

To promote early recognition, the recognizer can choose to share information, for example revealing safe cells. There is a budget, limiting the number of facts that can be revealed.

If the recognizer chooses to reveal cell $(3,1)$ *is safe (Figure 1(c)), an actor aiming at* G_1 *(originally indifferent to moving up or right) prefers moving right from the initial state. In contrast, an actor aiming at* G_2 *still prefers moving up, due to the added cost of passing through* $(3,1)$ *on the way to its goal. The goal of the actor becomes clear as soon as the first step is performed and* wcd *is minimized. Note that if, in addition, the recognizer reveals that cell* $(1,3)$ *is also safe (Figure 1(d)), the initial situation is recovered, since an agent to either goal may move up. This illustrates the need to carefully select the information to reveal to facilitate the recognition task.*

The contributions of this work are fourfold. First, we extend the GRD framework to support agents with partial information. We refer to our extended setting as *GRD for Agents with Partial Knowledge* (GRD-APK), and suggest information shaping modifications that can be applied to support goal recognition. Second, since our extended design setting induces a large space of possible information shaping modifications, we present a novel pruning method, and specify the conditions under which it is guaranteed not to prune optimal solutions. Third, we implement our suggested approach, using PDDL (McDermott *et al.* 1998) to represent our generic and adaptable redesign process. Finally, we evaluate the algorithm on a set of standard benchmarks, and demonstrate both the *wcd* reduction achievable through information shaping and the efficiency of our approach.

Related Work

GRD is a special case of the problem of *environment design* (Zhang *et al.* 2009), and was first introduced by Keren et al. (2014) and later extended to a variety of GRD settings (Keren *et al.* 2015; Son *et al.* 2016; Keren *et al.* 2016a; 2016b; Wayllace *et al.* 2016; Ang *et al.* 2017). Common to all previous GRD work is the assumption that actors have perfect observability of their environment. This includes the work of Keren *et al.* (2015), where the actor may be sub-optimal, but has perfect observability of its environment. Similarly, (Keren *et al.* 2016a; 2016b; 2018) accounts for setting where the *recognizer* has partial observability and sensor refinement modifications are applied to enhance it's sensor model, while the actor has perfect knowledge. Our work is the first to account for an actor with partial information, and suggests new information shaping modifications, implemented as sensor extensions, as a way to reduce the *wcd*.

Efficient communication via selective information revelation is fundamental to various multi agent settings, e.g., (Xuan *et al.* 2001; Wu *et al.* 2011; Unhelkar and Shah 2016; Dughmi and Xu 2016). This work is the first to suggest using information shaping as a one time and offline intervention stage in order to facilitate goal recognition.

Background

Goal Recognition Design

Goal Recognition Design (GRD) (Keren *et al.* 2014; 2015; 2016a; 2018; Wayllace *et al.* 2016; Son *et al.* 2016) is the task of analyzing and redesigning environments to support efficient online goal recognition. Following (Keren *et al.* 2018), the definition of a GRD setting includes two main components; the *goal recognition setting* and the *design model*.

The goal recognition setting can be defined in various ways (Sukthankar *et al.* 2014), but typically includes a description of the underlying environment, the way actors behave in it to achieve their goal, and the observations collected by the recognizer. The recognition settings can be classified as either 'keyhole', where the actor is agnostic to the recognition process, 'adversarial' where it tries to conceal its objectives, or 'intended,' where the actor helps the recognizer detect its objective (Carberry 2001; Cohen *et al.* 1981). In this work, we assume 'keyhole,' and support settings in which the actor's behavior is not affected by the recognizer's presence.

The design model specifies the ways by which a goal recognition setting can be modified (for example, by disallowing specific actions) and possible design constraints (for example, a design budget). The objective is to find a valid set of modifications to minimize the maximal path an actor may take before its goal is recognized, a measure referred to as *worst case distinctiveness* (*wcd*).

Contingent Planning

To support agents with partial knowledge, we follow Bonet and Geffner (2011) and consider partially observable contingent planning, formulated as follows.

Definition 1 *A* **partially observable planning with deterministic actions (POP-det) problem** *is a tuple* $P = \langle \mathcal{F}, \mathcal{A}, \mathcal{I}, \mathcal{G}, \mathcal{O} \rangle$ *where* $\mathcal{F}$ *is a set of fluent symbols,* $\mathcal{A}$ *is a set of actions,* $\mathcal{I}$ *is a set of* clauses *over fluent-literals defining the initial situation,* $\mathcal{G}$ *is a set of fluent-literals defining the goal condition, and* $\mathcal{O}$ *represents the agent sensor model.*

An action $a \in \mathcal{A}$ is associated with a fluent set of preconditions *prec*(a), and conditional effects *eff*(a). The sensor model $\mathcal{O}$ is a set of observations $o \in \mathcal{O}$ represented as pairs (C, L) where C is a set of fluents and L is a positive fluent, indicating that the value of L is observable when C is true. Each observation $o = \langle C, L \rangle$ can be conceived as a sensor on the value of L that can be activated when C is true.

A state s is a truth valuation over the fluents F ('true' or 'false'). For an agent, the value of a fluent may be known or unknown. A *belief state* b is a non-empty collection of states the agent deems possible at some point. A formula $\mathbb{F}$ *holds* in b if it holds for every state $s \in b$. The initial belief is the set of states that satisfy I, and the goal belief are those that satisfy G. A formula is *invariant* if it is true in each possible initial state, and remains true in any state that can be reached from the initial state. A fluent is *hidden* if its true value is unknown. An action a is *applicable* in b if the preconditions of a hold in b, and the *successor* belief state $b^{'}$ is the set of states that results from applying the actions a to each state s in b. When an observation $o = (C, L)$ is activated, the successor belief is the *maximal* set of states in b that agree on L. A *history* is the sequence of actions and beliefs $h = b_0, a_0, b_1, a_1, \ldots, b_n, a_n, b_{n+1}$. It is *complete* if the performing agent reaches a goal belief state.

A solution to a POP-det problem P is a *policy* Π, which is a partial function from beliefs to actions. A policy is *deterministic* if any belief b is mapped to at most one action. Otherwise it is *non-deterministic*. There are three policy types: *weak*, where at least one trajectory achieves the goal, and *strong* and *strong cyclic*, where every possible trajectory must achieve the goal. As opposed to strong cyclic policies, strong policies impose the extra constraint that trajectories cannot visit any state more than once.

The *k-planner* $K_{prudent}(P)$ Translation

A variety of solvers have been developed to solve a POP-det problem (e.g., (Bonet and Geffner 2011; Muise *et al.* 2014; Brafman and Shani 2012a)). Specifically, Bonet and Geffner (2011) suggest the *k-planner* that follows the *planning under optimism* approach; making the most convenient assumptions about the values of hidden variables, executing the plan that is obtained from the resulting classical planning problem, and revising the assumptions and re-planing, if during the execution, an observation refutes the assumptions made.

To transform the POP-det problem into a classical planning problem, the k-planner uses the $K(P)$ translation. At the core of $K(P)$ is the substitution of each fluent L in the original problem with a pair of fluents KL and $K\neg L$, representing whether L is known to be true or false, respectively (Albore *et al.* 2009). Each action $a \in \mathcal{A}$ is transformed into an equivalent action that replaces the use of every fluent L, with its corresponding fluent KL and $K\neg L$. Each sensor (C, L) is translated into two deterministic sensing actions, one for each possible value of L. For variables whose value is unknown, these sensing actions allow the solver to choose a preferred value, and correspond to the assumptions that the agent makes. Each invariant clause is translated into a set of actions, which we call *ramification actions*, that set the truth value of some variable, depending on the values of the other variables in the clause. The action set in the transformed problem is therefore $\mathcal{A}' = \mathcal{A}'_{ex} \cup \mathcal{A}'_{sen} \cup \mathcal{A}'_{ram}$, where $\mathcal{A}'_{ex}$, $\mathcal{A}'_{sen}$, and $\mathcal{A}'_{ram}$, represent the transformed, sensing and ramification actions, respectively. This representation captures the underlying planning problem at the knowledge level, accounting for the exploratory behavior of a partially informed agent.

Bonet and Geffner (2011) show that this linear translation of a POP-det problem into a classical planning problem is sound and complete for *simple* POP-det models with a connected state space. A POP-det model is simple if the non-unary clauses in I are all invariant, and no hidden fluent appears in the body of a conditional effect. In connected state spaces every state is reachable from any other. In simple problems there is no information loss and the model is *monotonic*, i.e., for every L, if L is known in a reachable belief state b over a simple problem P, and $b^{'}$ is a belief reachable form b, then L is known in $b^{'}$. As a consequence, for every policy π and history h of length n it follows that the number of states in beliefs b_i is a monotonically decreasing function, i.e. $|b_i| \geq |b_{i+1}|$ for every $0 \leq i < n$.

A key issue to note about the $K(P)$ compilation is that all its actions, including sensing and ramification actions, have equal cost. This means that a cost-minimizing solution to the resulting classical planning problem may be one that favors increasing the cost to goal, over the use of multiple ramification actions. As described in Example 1, we want a solver

that minimizes the number of actions needed to achieve its goal, breaking ties according to the number of assumptions and ramifications that need to be made. We therefore suggest the $K_{prudent}(P)$ translation, which extends the uniform cost $K(P)$ translation by associating a cost function to each action in $\mathcal{A}'$. Specifically, every transformed action $a \in \mathcal{A}'_{ex}$ is assigned a cost of 1, every sensing action (assumption) $a \in \mathcal{A}'_{sen}$ is assigned a small cost of ϵ, and every ramification action $a \in \mathcal{A}'_{ram}$ has 0 cost. When ϵ is small enough such that the accumulated cost of assumptions made for a plan at any replaning phase is smaller then minimal diversion from an optimal path, the optimal solution achieved using this formulation complies with our requirements.

Goal Recognition Design for Agents with Partial Knowledge (GRD-APK)

We focus on the problem of how to use information shaping in order to more effectively recognize the goals of partially informed agents. The *goal recognition design for agents with partial knowledge problem* (GRD-APK) consists of an initial goal recognition setting, a measure by which a setting is evaluated, and a design model, which specifies the information shaping modifications that can be applied. We define each of these components separately, before bringing them together (Definition 7).

Goal Recognition

The goal recognition model supports two agents; a partially informed contingent planning actor (Definition 1) with a goal, that executes history h until reaching a goal belief (or halting when no action is applicable). The second agent is a perfectly informed recognizer, that analyzes the actor's state transitions in order to recognize the actor's goal.

Definition 2 *The **goal recognition for agents with partial knowledge problem** (GR-APK) is a tuple $R = \langle \epsilon, \mathcal{G}, \theta, \mathcal{O}^{ac} \rangle$ where:*

- *$\epsilon = \langle \mathcal{F}, \mathcal{A}, \mathcal{I} \rangle$ is the environment, which consists of the fluents $\mathcal{F}$, actions $\mathcal{A}$ and initial state $\mathcal{I}$ as defined in Definition 1 (a cost $\mathcal{C}(a)$ for each action may also be specified).*
- *$\mathcal{G}$ is a set of possible goals G, s.t. $|\mathcal{G}| \geq 2$ and $G \subseteq F$.*
- *θ is the solver used by the actor to decide how to act in each belief state.*
- *$\mathcal{O}^{ac}$ is the actor's sensor model.*

Note that since we are analyzing the goal recognition setting, and need to account for all possible observations of agent behavior, we do not specify a particular history to be analyzed (a typical component in goal recognition models such as (Ramirez and Geffner 2010; Pereira *et al.* 2017)). Instead, it characterizes the different actor behaviors in the system, and the way they are perceived by the recognizer.

The cost of history h, denoted $\mathcal{C}_a(h) = \Sigma_i \mathcal{C}(a_i)$, is the accumulated cost of the performed actions (equal to path length when action cost is uniform). In executing h, the actor follows a (possibly non-deterministic) policy π from the set $\Pi(G)$ of possible policies to its goal. This set is induced by the environment ϵ, describing the possible states and the transitions between them, the sensor models $\mathcal{O}^{ac}$, describing the way the actor collects observations from its surrounding and updates its belief state, and the actor's solver θ. The solver we support here is any variant of the *k-planner* approach presented in the previous section.

In our setting, the actor and recognizer both know ϵ and the set $\mathcal{G}$ of possible goals. While the partially informed actor needs to collect information about the environment via its sensor model in order to achieve its premeditated goal, the recognizer knows the true state of the world and the actor's solver and sensor, but does not know the actor's goal. Also, it cannot see the actor's actions, but can observe its transitions between states. Since the recognizer has access to the actor's sensor model, it can deduce its transitions between belief states, and analyze them in order to recognize the actor's goal as early as possible.

Evaluating a GR-APK model

We want to minimize the *worst case distinctiveness* (*wcd*), which represents the maximum number of actions (in general, maximum total cost incurred by the actor) before its goal is revealed. To define *wcd*, we first define the relationship between the observations collected by the recognizer and a goal.

Under the assumptions we make, when an actor follows history h, the recognizer only observes the actor's transitions between belief states. We say that history h *satisfies* policy π, if it can be generated by an agent following π. It *satisfies* a goal G, if π is a possible policy to G.

Definition 3 *Given a GR-APK model R, history h* **satisfies** *policy π in R, if $\forall i\ 0 \leq i \leq n$, $a_i \in \pi(b_i)$. In addition, h* **satisfies** *goal $G \in \mathcal{G}$ if exists $\pi \in \Pi(G)$ s.t. h satisfies π.*

Let $\mathcal{G}^{rec}(h)$ represent the set of goals that history h satisfies, i.e., the set of goals the recognizer deems as possible actor goals. We define a history as *non-distinctive* if it satisfies more than one goal.

Definition 4 *Given a GR-APK model R, a history h is* **non distinctive** *in R, if exists G, G' s.t. $G \neq G'$, and h satisfies G and G'. Otherwise, it is distinctive.*

We mark the set of non-distinctive histories of a GR-APK model R by $H^{nd}(R)$.

Definition 5 *The* **worst case distinctiveness** *of a model R, denoted by* $\text{wcd}(R)$ *is:*

$$\text{wcd}(R) = \begin{cases} \max\limits_{h \in H^{nd}(R)} \mathcal{C}_a(h) & H^{nd}(R) \neq \emptyset \\ 0 & otherwise \end{cases}$$

Recall that a policy $\pi \in \Pi(\mathcal{G})$ may be strong cyclic, potentially containing infinite loops. A policy with such a cycle is considered to have a history with infinite cost. In particular, this means *wcd* in this setting can have an infinite cost.

Information Shaping

Our interest here is in modulating the behavior of the actor through information shaping; by changing the actor's knowledge, we can potentially change its behavior and the way

by which it acquires the information needed to achieve it's goal. We restrict the information shaping interventions to be *truthful* so that they cannot convey false information. In the context of contingent planning agents, this requirement is naturally implemented by requiring that we may only improve the actor's sensor model, i.e., improving its ability to access the true value of some environment feature. We define *sensor extension* modifications, which add a single observation to a senor model, using $\boldsymbol{\mathcal{O}}$ to denote the set of all sensor models.

Definition 6 *A modification* $\delta : \boldsymbol{\mathcal{O}} \rightarrow \boldsymbol{\mathcal{O}}$ *is a* **sensor extension** *if* $\delta(\mathcal{O}) = \mathcal{O} \cup \{o\}$*, for all* $\mathcal{O} \in \boldsymbol{\mathcal{O}}$*, and for some* $o = (C, L)$.

Sensor extensions correspond to adding new sensors to the environment, or, as a special case, communicating to the actor the true value of a feature (setting $C = \emptyset$).

To demonstrate, in Example 1 the recognizer can allow the actor to sense a stench in cell $(1, 2)$, two (rather then one) cells away from the wumpus in cell (3,2). This extension is implemented by adding the observation $o = (C = AgentAtCell(1, 2), L = BreezeInCell(2, 2))$ to the actor's sensor model. This could be realized through a visual indication or sign, similar to the auditory signal indicating the oven is hot in the assisted cognition example. The recognizer could also directly communicate with the actor and inform it about the location of a wumpus, or reveal a location without a wumpus. (e.g., $(C = True, L = WumpusAtCell(4, 4))$.

We are now ready to define a GRD-APK problem.

Definition 7 *A* **goal recognition design for agents with partial knowledge problem** *(GRD-APK) is defined as a tuple* $T = \langle R_0, \boldsymbol{\Delta}, \mathbf{b} \rangle$ *where:*

- R_0 *is the initial goal recognition model,*
- $\boldsymbol{\Delta}$ *are the possible sensor extensions, and*
- b *is a budget on the number of allowed extensions.*

We want to find a set $\Delta \subseteq \boldsymbol{\Delta}$ of up to b sensor extensions to apply to R_0 to minimize the *wcd*. This objective is formally defined below, where $wcd^{min}(T)$ is the minimum *wcd* achievable in a GRD model T, and R^{Δ} is the goal recognition model that results from applying set Δ to R.

$$\begin{aligned} & wcd^{min}(T) = \min_{\Delta \subseteq \boldsymbol{\Delta}} \quad wcd(R_0^{\Delta}) \\ & s.t. |\Delta| \leq b \end{aligned} \tag{1}$$

Any solution to Equation 1 is *optimal*, i.e. it achieves the minimal *wcd* possible. It is *strongly optimal* if it has minimum size among all optimal solutions, i.e., it includes the minimal number of extensions needed to minimize *wcd*.

Methods for Information Shaping

In our search for an optimal design, we consider a sensor extension as *useful* with regards to a goal recognition model, if it reduces the *wcd*. Given a goal recognition model R and a sensor extension δ, we let T^{δ} denote the model that results from applying δ to the actor's sensor model $\mathcal{O}$, and define useful sensor extensions as follows.

Definition 8 *A modification* δ *is* **useful** *with regards to goal recognition model* R *if* $\text{wcd}(R^{\delta}) < \text{wcd}(R)$.

The challenge in information shaping comes from two sources. First, the number of possible information shaping options may be large, and evaluating the effect of each change may be costly, making it important to develop efficient search techniques. Also, the problem is non-monotonic, in that sensor extensions are not always useful, and providing more information may actually make recognition more difficult (Example 1).

To address these challenges, we follow Keren *et al.* (2018) and formulate the design process as a search in the space of modification sets $\Delta \subseteq \boldsymbol{\Delta}$. The root is the initial goal recognition model R_0, and the operators are the sensor extensions $\delta \in \boldsymbol{\Delta}$ that transition between models. Each node is evaluated by its *wcd*.

Calculating *wcd*

To calculate the *wcd* of a model, we need to find the maximal prefix of a policy that satisfies more than one goal. Recall that we assume the actor's solver is known to the recognizer, who cannot observe which actions are performed by the actor, but does observe its transition between states. At the initial state, all goals are possible. As the actor progresses in the system, the set of goals satisfied by its execution decreases. Given a history h, we let h_i represent its prefix up to belief state i.

Lemma 1 (GR Monotonicity) *Given a GRD-APK model* R *and a history* h *of* n *actions, for all* $0 \leq i < j \leq n$*,* $\mathcal{G}^{rec}(h_j) \subseteq \mathcal{G}^{rec}(h_i)$.

Proof Sketch: The definition of sensor models guarantees the true state is always included in the actor's belief. Moreover, it guarantees the belief is *monotonic*, i.e., as the actor advances, the number of states in its belief cannot increase. Consequentially, as the actor advances, the set of plans, and therefore goals, satisfied by the history can only decrease. ■

Lemma 1 guarantees a non-distinctive execution cannot have a distinctive prefix. We can therefore find the *wcd* of a GR-APK model by first calculating the plan to each of the goals. Then, starting at the initial state, we iteratively explore the non-distinctive policy prefixes, until its most distant boundary is found, and return its length (cost).

Design with *CG-pruning*

The baseline approach for searching in the modification space is a *breadth first search* (BFS), using our *wcd* calculation method to evaluate each node. Under the budget constraints, BFS explores modification sets of increasing size, avoiding the computation of pre-computed sets. The search halts if a model with $wcd = 0$ is found or if there are no more nodes to explore, and returns the shortest path (modification set) to a node that achieves the minimal *wcd*. This iterative approach is guaranteed to find a strongly optimal solution, i.e., a minimal set of modifications that minimizes the *wcd*. However, it does not scale with larger problems.

To increase efficiency, pruning can be applied to reduce the size of the search space. However, conditions under which pruning is *safe* (Wehrle and Helmert 2014), such that at least one optimal solution remains unpruned, offered previously for the case of perfectly informed agents (Keren *et*

al. 2018), do not hold in the present setting. We therefore suggest a new pruning approach that eliminates useless modifications, and specify conditions under which it can be used to find optimal solutions.

At a high level, our pruning technique, *CG-pruning*, uses the translation from a partially observable to fully observable planning for each goal. Given the result of the translation, we use a *causal graph* (Williams and Nayak 1997; Helmert 2006), to extract an encoding of the usefulness of information with regards to each goal, and prune any sensor extension that reveals information irrelevant to all goals.

Example 2 *To exemplify, consider Figure 2(left), depicting a modified version of Example 1, where the actor needs to collect a key to be able to access its goal (e.g.,* $PickedKey1$ *is a needed to reach* G_1*). There are multiple keys distributed in the grid (e.g.,* $Key1At(4,2)$*), each needed for accessing a particular location. The actor initially only knows a set of possible key locations for each key. When in a cell with a key, it senses it and can pick is up and use it to achieve its goal. In this scenario, the recognizer, with perfect information, can notify the actor about safe locations, as before, but also about the absence or presence of a particular key in some location. Applying the* $K(P)$ *transformation here creates fluents* $K_KeyiAt(x,y)$ *and* $K\neg KeyiAt(x,y)$ *for each key and location, representing whether the actor knows key* i *is or isn't at location* (x,y)*, respectively. The former is a precondition to picking up the key.*

Given a goal recognition model R, for every goal in $\mathcal{G}$, we use the $K(P)$ (or its variant $K_{prudent}(P)$ introduced above), to transform the partially observable planning problem into a fully observable problem. We use the result of the transformation to create the causal graph for each goal. Specifically, we use the causal graph as suggested by Helmert (2006). In this context, for every goal $G \in \mathcal{G}$, the resulting causal graph $CG(G) = (V, E)$ is one where the nodes V represent the multi-valued variables extracted from the propositional problem. The directed edges E represent dependencies between variables, such that the graph contains an edge $(v, v')$for $v, v' \in V$, if changes in the value of v' can depend on the value of v. By encoding all possible hierarchical dependencies between the different state variables, the causal graph captures all variables relevant for achieving the goal.

Recall, that our actor uses a *k-planner* to iteratively compute a policy; at each iteration it transforms the partially observable problem into the corresponding fully observable transformation, and uses a classical planner to compute a plan to its goal, making the most convenient assumptions about the values of hidden variables. It executes this plan until the goal is achieved or until one of its assumptions is refuted, in which case it revises the assumptions and replans. For each execution iteration $0 \ldots n$ and model R, we let $CG_i^R(G)$ represent the causal graph that corresponds to the planning problem of R at iteration i. Our pruning technique, *cg-pruning* is based on the computation of the causal graphs of the initial problem ($CG_0^R(G)$ for every $G \in \mathcal{G}$). We prune sensor extensions of variables (and their corresponding fluents) that are not in any $CG_0^R(G)$.

Figure 2(right), show a part of the causal graph that is generated for G_1. As demonstrated, the causal graph to each goal excludes any variables concerning the location of keys to other goals. Using this graph, we prune sensor extensions regarding variables that do not appear in any of the causal graphs (for example, the sensor extension that reveals the location of k_3).

Next, we show that *CG-pruning* is safe whenever the actor uses the *k-planner* with an optimal solver to computes its plan to the $K(P)$ (or $K_{prudent}(P)$) transformed problem.

As a first step in our proof, recall that the transformed problem includes sensing actions $\mathcal{A}'_{sen}$, that correspond to assumptions that can be made by an agent w.r.t values of unknown variables. We say that a sensor extension is *related to* a causal graph, if it changes the conditions under which the true value of a variable in the causal graph (and its corresponding fluents) can be observed. By applying a sensor extensions, we are limiting the assumptions an actor can make at some state about the value of a variable, and consequently the set of applicable sensing actions. As a result, a sensor extension can only remove nodes (variables) or edges from the causal graph. In the following, R^δ is the goal recognition model that results from applying δ to R.

Lemma 2 *For any model* R *and sensor extension* δ*,* $CG_0^{R^\delta}(G)$ *is a subgraph of* $CG_0^R(G)$ *for any* $G \in \mathcal{G}$*.*

Proof Sketch: We assume to the contrary that there is some node or edge in $CG_0^{R^\delta}(G)$ that did not belong to $CG_0^R(G)$. We then show that since a sensor extension adds a sensor $o = (C, L)$ to the actor's sensor model, any sensing action $a \in \mathcal{A}'_{sen}$ that does not set the value of L to its true value, is non-applicable in R^δ. Consequently, sensor extensions cannot add edges or nodes to the causal graph, thus reaching a contradiction. ∎

As a direct consequence of Lemma 2, Corollary 1 states that for any goal, the causal graph of any goal recognition model R that is reachable from the initial model R_0 by a set of sensor extensions, is a subset of $CG_0^R(G)$.

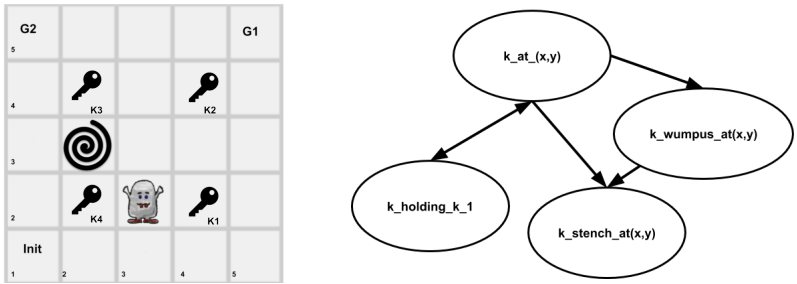
Corollary 1 *For any sensor extension set* $\Delta \subseteq \mathbf{\Delta}$ *and goal* $G \in \mathcal{G}$*,* $CG_0^{R^\Delta}(G)$ *is a subgraph of* $CG_0^R(G)$*.*

The implication of Corollary 1 is that any sensor extension that is not related to the initial causal graph of some model, is guaranteed not to be related to the causal graph of any model that can be reached via design.

Next, we observe that when an optimal solver is used, a sensor extension that does not reduce the initial causal graph to some goal, leaves the plans to all goals unchanged, and is therefore not useful.

Lemma 3 *Assuming the actor uses an optimal solver, for any model* R *and sensor extension* δ*, if for all* $G \in \mathcal{G}$*,* δ *is not related to* $CG_0^R(G)$*, then* δ *is not useful w.r.t* R*.*

Proof Sketch: Bonet and Geffner (2011) show that $K(P)$ transformation is sounds and complete for simple problem



(a) (b)

Figure 2: The Wumpus domain with keys

with a connected space, which are the only problems we are considering here. Helmert (2006) shows that any optimal plan can acquired by ignoring variables that are not in the causal graph. Therefore, by pruning sensor extensions that are related to variables not on the causal graph, we are removing from the actor's planning graph sensing actions that would anyway not appear in any optimal plan. This means that the behavior of any optimal agent to any goal is not affected by such sensor extensions.. Moreover, since sensor extensions only remove possible sensing actions, as the actor progresses and replans, no sensing actions can be added to the actor's model. Consequently, the behavior w.r.t to any goal will not change, the *wcd* will not change, and δ is not useful w.r.t R.

■

Lemma 3 considers a single sensor extension applied to some model R, and specifies conditions under which it is guaranteed not to be useful. In the context of our search graph, in which we apply a modification at each node, we want to prune sensor extensions that are not useful w.r.t a given model, while guaranteeing that the modification is not useful to any model that is reachable from R by applying a set of sensor extensions.

Corollary 2 *If the actor uses an optimal solver, any sensor extension δ that is not useful w.r.t R, is not useful for any model R^{Δ} for any set Δ.*

Proof Sketch: The proof is a direct consequence of Corolalry 1 and Lemma 3. ■

Finally, we are ready to show that *CG-pruning* is safe.

Theorem 1 *For any GRD-APK model $T = \langle R_0, \mathbf{\Delta}, \mathbf{b} \rangle$,* CG-pruning *is safe for an actor that uses the* k-planner *with an optimal planner.*

Proof Sketch: A pruning function is safe if it is guaranteed that after the pruning, some optimal solution is not pruned. Corollary 2 guarantees, that under the assumptions we make, any sensor extension δ that is not useful w.r.t some model R, is not useful for any model that is reachable from it via design (i.e., any model R^{Δ} for any set $\Delta \subseteq \mathbf{\Delta}$). This is true in particular for the initial goal recognition model R_0. Therefore, sensor extensions that are not related to $CG_0^R(G)$, are not useful to any reachable model and will not be part of a strongly optimal solution. Therefore *CG-pruning* is safe.

■

	Possible Goals	Sensor Extensions
WUMPUS	gold locations	safe cells
WUMPUS-KEY	gold locations	safe cells or locations with / without keys
C-BALLS	ball distribution	locations without a ball
TRAIL	final stone locations	locations with / without stones
LOGISTICS	package destination	package locations

Table 1: Possible goals and design options for each domain.

		No Pruning				*CG-pruning*			
	budget	sol	Δ*wcd*	time	nodes	sol	Δ*wcd*	time	nodes
WUMPUS	b=1	0.1	0.0 (1.8)	92.84	14.0	0.1	0.0 (1.8)	76.96	11.0
	b=2	0.1	0.0 (1.8)	663.71	106.0	0.1	0.0 (1.8)	421.27	67.0
WUMPUS-KEY	b=1	1.0	0.2(0.57)	16.05	4.1	1.0	0.2(0.57)	12.59	3.3
	b=2	0.71	0.2(0.57)	238.74	39.5	0.72	0.2(0.57)	200.34	28.8
LOGISTICS	b=1	0.14	8.0 (11.83)	1330.33	3.0	0.14	8.0(11.83)	1042.52	2.0
	b=2	NA	NA	NA	NA	NA	NA	NA	NA

Table 2: Results per domain for the FD solver (optimal)

Initial Empirical Evaluation

Our objective is to evaluate both the effect sensor extensions have on *wcd* as well as the efficiency of *CG-pruning*.We start by describing our dataset, and empirical setup, and then discuss our initial results.

Dataset. We used five domains adapted from Bonet and Geffner (2011) and Albore *et al.* (2009).

		No Pruning				*CG-pruning*			
	budget	sol	Δ*wcd*	time	nodes	sol	Δ*wcd*	time	nodes
WUMPUS	b=1	1.0	0.0 (3.0)	88.02	16.0	1.0	0.0 (6.0)	59.98	11.0
	b=2	0.25	0.0 (3.0)	697.31	137.0	1.0	0.0 (6.0)	351.37	67.0
WUMPUS-KEY	b=1	1.0	4.33 (4.33)	16.71	13.55	1.0	4.33 (4.33)	13.35	10.56
	b=2	1.0	3.95 (3.95)	85.73	54.56	0.8	3.95 (3.95)	75.55	42.55
C-BALLS	b=1	0.8	9.12 (9.2)	36.61	37.03	0.8	9.12 (9.2)	38.75	37.03
	b=2	0.8	11.5 (10.83)	30.19	22.01	0.8	11.5 (10.83)	30.19	22.01
TRAIL	b=1	1.0	0.0 (0.0)	14.71	28.0	1.0	0.0 (0.0)	12.97	26.5
	b=2	1.0	0.0 (0.0)	195.39	407.0	1.0	0.0 (0.0)	173.21	365.5
LOGISTICS	b=1	0.42	3.01 (4.14)	22.90	61.5	1.0	3.01 (4.14)	19.41	42.67
	b=2	0.28	9.05 (9.27)	133.68	234.4	0.86	9.05 (9.17)	112.89	175.1

Table 3: Results per domain for the FF solver (satisfying)

- WUMPUS: corresponding to the setting in Example 1.
- WUMPUS-KEY: corresponding to the setting in Example 2.
- C-BALLS (Colored-balls): the actor navigates a grid to pick up and deliver balls of different and initially unknown colors and destinations.
- TRAIL: An agent must follow a trail of bits to reach a destination, while sensing only the bits surrounding it.
- Logistics: Packages are transported to their destinations, relaying on sensing to reveal the packages in a location.

The adaptation from contingent planning to GRD-APK involves specifying for each instance the set of possible goals and sensor extensions (see Table 1 for details).

To support the design process, we use a PDDL file to specify the available modifications (and their preconditions). Sensor extension modifications are implemented as design actions, that add to the initial state fluents that represent the true value of a variable.

Setup. We use the *k-replanner* (Bonet and Geffner 2011) to represent the actor's solver, with two variations. For the first, the $K(P)$ compilation was used together with the satisfying FF classical planner (FF) (Hoffmann and Nebel 2001). The second used the $K_{prudent}(P)$ compilation together with the optimal Fast-Downward (Helmert 2006) classical planner (FD).

In our computation of *wcd*, we also consider the prefixes of failed executions, since they represent valid agent behavior. The design process is implemented as a breadth-first search (BFS) in the space of modification space, tested with and without *CG-pruning*.

We use 30 instances for each domain, and a design budget of $1-2$. Each execution had a time limit of 20 minutes and is capped at 1000 search steps (each corresponding to a design set), whichever was first.

To parse the design file, we adopt the parser of *pyperplan* (Alkhazraji *et al.* 2016). For each modification sequence, which represents a GRD-APK model and a node in our search, the parser provides the applicable successor modifications and the model that results from applying each of them.

Results. Tables 2 and 3 summarize the results per approach (No Pruning vs. *CGPruning*) for the FD and FF solver, respectively. For each domain and design budget ($b = 1$ and $b = 2$), the tables shows 'sol', as the fraction of instances completed within the time and resource bounds. For instances completed by both approaches, 'Δ-*wcd*' is the average *wcd* reduction. In parenthesis we show the *wcd* reduction achieved on all executions, including those that timed out. 'time' is the average calculation time (in seconds) for each approach, and 'nodes' is the average number of nodes evaluated on all instances. 'NA' represent settings for which no instance completed. In Table 2 we excluded C-BALLS and TRAIL, since no problem completed for both domains.

Our initial results show that design via information shaping reduces the *wcd* for all domains, with a reduction of 9.12 (about half) for C-BALLS. For all domains *CG-pruning* reduces the number of nodes explored and computation time for completed problems. For WUMPUS, WUMPUS-KEY and LOGISTICS using FF, *CG-pruning* also increases the ratio of solved problems.

The results show the potential of our pruning approach. However, many instances were not completed for FD, failing in some cases to complete the solution of the initial setting. To achieve more results for the optimal case, and hopefully a stronger indication of the benefit of our approach in such settings, we intend to add additional domains to our dataset and explore different heuristics used to guide the optimal search. We also intend to enhance pruning further. Specifically, using the plan the actor intends to execute with regards to each goal, we can prune sensor extensions that correspond to assumptions already made by the actor, and show that they will not reduce the *wcd*.

Conclusion

We have introduced GRD for the setting of a partially informed actor and an informed recognizer, who can share information about the domain with the actor. We formalized the information shaping design problem as one of minimizing the worst-case distinctiveness(*wcd*), and presented new sensor extension modifications, used to enhance recognition. To efficiently search the space of applicable sensor extensions, we present a novel pruning approach for finding the optimal set of sensor extensions. To the best of our knowledge, our approach is the first to suggest using techniques originally developed for classical planning, in the context of goal recognition of partially informed planning agents. Our initial results on a set of standard benchmarks show that *wcd* can be reduced via information shaping, as well as the efficiency of our approach.

There are many ways to extend this work. First, we use qualitative contingent planning models to represent the partially informed agents and their belief states. It would be interesting to also work with Partially Observable Markov Descision Process (POMDP) models (Kaelbling *et al.* 1998) to represent the actor. Another interesting direction is to consider settings where the actor is aware of the recognizer's presence. Specifically, our approach can be adopted to "transparent planning" (MacNally *et al.* 2018), where actors choose behaviors that facilitate recognition. These models rely on partially informed agents to be able to choose a behavior that maximizes the information conveyed about their intentions. In such settings, GRD can be viewed as a complementary approach, that can be applied to alleviate the need to completely rely on the actor, and reduce the number of non-distinctive behaviors. Finally, while we focus on pruning as a way to increase efficiency, other options are possible. In particular, heuristics can be used to estimate the value of a modification, and lead the search in promising directions.

References

Alexandre Albore, Héctor Palacios, and Héctor Geffner. A translation-based approach to contingent planning. In *IJCAI*, pages 1623–1628, 2009.

Yusra Alkhazraji, Matthias Frorath, Markus Grutzner, Thomas Liebetraut, Manuela Ortlieb, Jendrik Seipp, Tobias Springenberg, Philip Stahl, Jan Wulfing,

Malte Helmert, and Robert Mattmuller. Pyperplan: https://bitbucket.org/malte/pyperplan, 2016.

Samuel Ang, Hau Chan, Albert Xin Jiang, and William Yeoh. Game-theoretic goal recognition models with applications to security domains. In *International Conference on Decision and Game Theory for Security*, pages 256–272, 2017.

M. S. Boddy, J. Gohde, T. Haigh, and S. A. Harp. Course of action generation for cyber security using classical planning. In *CAPS 2005*, pages 12–21, 2005.

Blai Bonet and Hector Geffner. Planning under partial observability by classical replanning: Theory and experiments. In *IJCAI 2011*, pages 1936–1941, 2011.

Ronen Brafman and Guy Shani. Replanning in domains with partial information and sensing actions. *Journal of Artificial Intelligence Research 2012*, 45:565–600, 2012.

Ronen I Brafman and Guy Shani. A multi-path compilation approach to contingent planning. In *AAAI*, 2012.

S. Carberry. Technique for plan recognition. *User Modeling and User-Adapted Interaction*, 11(1-2):31–48, 2001.

P. R. Cohen, C. R. Perrault, and J. F. Allen. Beyond question-answering. Technical report, DTIC Document, 1981.

Shaddin Dughmi and Haifeng Xu. Algorithmic bayesian persuasion. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 412–425, 2016.

Malte Helmert. The fast downward planning system. *Journal of Artificial Intelligence Research(JAIR)*, 26:191–246, 2006.

Jörg Hoffmann and Bernhard Nebel. The ff planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:253–302, 2001.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

H. Kautz and J. F. Allen. Generalized plan recognition. In *AAAI*, volume 86, pages 32–37, 1986.

H. Kautz, O. Etzioni, D. Fox, D. Weld, and L. Shastri. Foundations of assisted cognition systems. Technical report, University of Washington, 2003.

Sarah Keren, Avigdor Gal, and Erez Karpas. Goal recognition design. In *Proceedings of the International Conference on Automated Planning and Scheduling*, 2014.

Sarah Keren, Avigdor Gal, and Erez Karpas. Goal recognition design for non optimal agents. In *(AAAI 2015)*, January 2015.

Sarah Keren, Avigdor Gal, and Erez Karpas. Goal recognition design with non-observable actions. In *AAAI*, 2016.

Sarah Keren, Avigdor Gal, and Erez Karpas. Privacy preserving plans in partially observable environments. In *IJCAI*, 2016.

Sarah Keren, Avigdor Gal, and Erez Karpas. Strong stubborn sets for efficient goal recognition design. In *Proceedings of the International Conference on Automated Planning and Scheduling*, 2018.

Steven James Levine and Brian Charles Williams. Concurrent plan recognition and execution for human-robot teams. In *ICAPS 2014)*, 2014.

Aleck MacNally, Nir Lipovetzky, Miquel Ramirez, and Adrian Pearce. Action selection for transparent planning. In *Proceedings of the Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2018.

Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. Pddl-the planning domain definition language. Technical report, Yale Center for Computational Vision and Control, 1998.

Christian J Muise, Vaishak Belle, and Sheila A McIlraith. Computing contingent plans via fully observable non-deterministic planning. In *AAAI 2014*, pages 2322–2329, 2014.

Ramon Fraga Pereira, Nir Oren, and Felipe Meneguzzi. Landmark-based heuristics for goal recognition. In *AAAI*, 2017.

M. Ramirez and H. Geffner. Plan recognition as planning. In *(IJCAI 2009)*, 2009.

M. Ramirez and H. Geffner. Probabilistic plan recognition using off-the-shelf classical planners. In *(AAAI 2010)*, 2010.

Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Pearson Education, 2016.

Tran Cao Son, Orkunt Sabuncu, Christian Schulz-Hanke, Torsten Schaub, and William Yeoh. Solving goal recognition design using asp. In *AAAI*, 2016.

Gita Sukthankar, Christopher Geib, Hung Hai Bui, David Pynadath, and Robert P Goldman. *Plan, activity, and Intent Recognition: Theory and practice*. Newnes, 2014.

Vaibhav V Unhelkar and Julie A Shah. Contact: Deciding to communicate during time-critical collaborative tasks in unknown, deterministic domains. In *AAAI*, pages 2544–2550, 2016.

Christabel Wayllace, Ping Hou, William Yeoh, and Tran Cao Son. Goal recognition design with stochastic agent action outcomes". In *IJCAI*, 2016.

Martin Wehrle and Malte Helmert. Efficient stubborn sets: Generalized algorithms and selection strategies. In *Proceedings of the Nineteenth International Conference on Automated Planningand Scheduling*, 2014.

Brian C Williams and P Pandurang Nayak. A reactive planner for a model-based executive. In *IJCAI*, volume 97, pages 1178–1185, 1997.

Feng Wu, Shlomo Zilberstein, and Xiaoping Chen. Online planning for multi-agent systems with bounded communication. *Artificial Intelligence*, 175(2):487–511, 2011.

Ping Xuan, Victor Lesser, and Shlomo Zilberstein. Communication decisions in multi-agent cooperation: Model and experiments. In *Proceedings of the fifth international conference on Autonomous agents*, pages 616–623, 2001.

Haoqi Zhang, Yiling Chen, and David C Parkes. A general approach to environment design with one agent. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 2002–2008, 2009.