Twenty-Eighth International Conference on Automated Planning and Scheduling (ICAPS 2018)

Strong Stubborn Sets for Efficient Goal Recognition Design

Sarah Keren, Avigdor Gal, Erez Karpas
{sarahn@campus,avigal@ie,karpase@}.technion.ac.il
Technion — Israel Institute of Technology

Abstract

Goal Recognition Design (*GRD*) is the task of redesigning environments (either physical or virtual) to allow efficient online goal recognition. In this work we formulate the redesign problem as an optimization problem, aiming at early goal recognition. To this end, we use a measure of *worst case distinctiveness* (*wcd*), which represents the maximal progress an agent may make before his goal is revealed. With the objective of minimizing *wcd*, we construct a search space in which each node in the space is a goal recognition model (one of which is the original model given as input) and one can move from one model to another by applying a model modification, chosen from a set of allowed modifications given as input. Our specific contribution in this work includes the specification of a class of modifications for which we can prune the search space using strong stubborn sets. Such positioning allows reducing the computational overhead of design while preserving completeness. We show that the proposed modification class generalizes previous works in goal recognition design and enriches the state-of-the-art with new modifications for which strong stubborn set pruning is safe. We support our approach by an empirical evaluation that reveals the performance gain brought by the proposed pruning strategy in different goal recognition design settings.

Introduction

Goal Recognition Design (*GRD*) (Keren, Gal, and Karpas 2014; 2015; 2016a; Wayllace et al. 2016; Son et al. 2016) is the task of redesigning environments (either physical or virtual) to allow efficient online goal recognition. *GRD* finds a valid sequence of modifications to some initial model which minimizes the *worst case distinctiveness* (*wcd*) of the model, representing the maximal progress an agent may make before his goal is revealed. *GRD* is relevant to any domain for which quickly performing goal recognition is essential and in which the model design can be controlled, *e.g.*, intrusion detection (Jarvis, Lunt, and Myers 2004; Kaluza, Kaminka, and Tambe 2011; Boddy et al. 2005), assisted cognition (Kautz et al. 2003), and human-robot collaboration (Levine and Williams 2014).

Analyzing *GRD* problems involves two main tasks; evaluating the *wcd* of a goal recognition model and optimizing a model by redesigning it. In this work we focus on the latter task, aiming at efficient ways to search for an optimal redesign strategy. With this objective, we construct a search space in which each node in the space is a goal recognition model, one of which is the original model. One can transition from one model to another by applying a model modification, chosen from a set of allowed modifications given as input.

Our main contribution in this work involves the characterization of a class of *GRD* models for which an effective pruning of the search space can be done using Strong Stubborn Sets (Valmari 1989; Wehrle and Helmert 2014). Originally suggested in the area of computer aided verifications, Strong Stubborn Sets are used to guarantee that the subset of modifications applied at each node in the search are chosen in a way that preserves completeness. Such positioning allows reducing the computational overhead of design while ensuring an optimal redesign strategy is found. We show that the proposed class generalizes previous works in *GRD* and enriches the state-of-the-art with new modifications that can be applied within *GRD* models and ensure valid pruning using Strong Stubborn Sets. Specifically, we present two new modification methods in this class, *action conditioning*, which enforces a partial order between actions and *single action sensor refinement*, which improves the observer's sensor model.

We support our approach by an empirical evaluation that reveals the performance gain brought by the proposed pruning strategy in different *GRD* settings. We show that by enriching the set of possible modifications we gain *wcd* reduction in a variety of settings.

Model

A *GRD* problem has two components: the analyzed goal recognition setting and a *design model*, specifying the possible ways to modify the goal recognition setting. Adopting notation from Keren, Gal, and Karpas (2016b; 2017) we formulate each component separately before integrating them into a *GRD* model (Definition 3).

Based on the STRIPS formalism (Fikes and Nilsson 1972), we support the analysis of goal recognition models based on planning domains as follows.

Definition 1 *A goal recognition model R is represented by the tuple $R = \langle F, I, A, C, G, O, S \rangle$ where:*

- $F \subseteq \mathcal{F}$ *is a set of fluents*
- $I \subseteq F$ *is the initial state,*
- $A \subseteq \mathcal{A}$ *is a set of actions.*
- $C : \mathcal{A} \rightarrow \mathbb{R}$ *specifies the cost of each action.*
- G *is a set of possible goals g s.t. $|G| \geq 2$ and $g \subseteq F$.*
- O *is a set of observation tokens, including the special observation token $o_\emptyset$, denoting that an action could be non-observable.*
- $S : \mathcal{A} \rightarrow 2^O \setminus \emptyset$ *is a sensor model, mapping each action $a \in \mathcal{A}$ into a set of observation tokens $S(a) \subseteq O$ that may be emitted when a is executed.*

We let $\mathcal{R}$ represent the set of goal recognition models that comply with Definition 1. We assume agents are agnostic to the goal recognition system. Each agent, aiming at one of the goals $g \in G$, enters the system at the initial state I and executes one of the legal plans $\pi \in \Pi^{leg}(g)$, which are the plans that achieve g and are allowed under the assumptions made on the behavior of the agent. The set of *legal paths* $\vec{\Pi}^{leg}(g)$ is the set of prefixes of the plans in $\Pi^{leg}(g)$.

The sensor model S refers to the way the goal recognition system observes agent actions, while agents are assumed to have full observability of the world state. This creates a distinction between the agent's activity and the way it is perceived by the system. Consequently, each executed path $\vec{\pi}$ may emit any one of its possible *observable projections* $op(\vec{\pi})$, the observation token sequences that may be emitted when $\vec{\pi}$ is performed.

Following (Keren, Gal, and Karpas 2016b), an observation sequence $\vec{o}$ *satisfies* a goal g if it is a possible observable projection of a path to g. A path is *non-distinctive* if at least one of its observable projections satisfies more than one goal. We let $\vec{\Pi}^{nd}(R)$ represent the set of non-distinctive paths in R. The *worst case distinctiveness*(*wcd*) of model R, denoted by $wcd(R)$, is the maximal cost of a path in $\vec{\Pi}^{nd}(R)$.

We note that the set $\Pi^{leg}(g)$ of legal plans to goal g can be described either explicitly or symbolically. (*e.g.*, the set of all optimal paths that do not make use of action a). In particular, when agents are optimal and the environment is fully observable the *wcd* can found using the compilation to planning based technique presented in (Keren, Gal, and Karpas 2014). Similarly, the compilations suggested in (Keren, Gal, and Karpas 2015; 2016a; 2016b) can be used to find the *wcd* for settings where agents are bounded-suboptimal and the environment is partially observable. In any case, the framework we present for *wcd* reduction is independent of the method used for its calculation.

The design model is defined next.

Definition 2 *A design model D is represented by the tuple $D = \langle \mathcal{M}, \delta, \phi \rangle$ where:*

- $\mathcal{M}$ *is a finite set of atomic modifications that can be applied. A modification sequence is an ordered set of modifications $\vec{m} = \langle m_1, \ldots, m_n \rangle$ s.t. $m_i \in \mathcal{M}$ and $\vec{\mathcal{M}}$ is the set of all such sequences.*
- $\delta : \mathcal{M} \times \mathcal{R} \rightarrow \mathcal{R} \cup \{R_\emptyset\}$ *is a deterministic modification transition function, specifying the goal recognition model that results from applying a modification to a goal recognition model. Non-applicable transitions result in the invalid model $R_\emptyset$.*
- $\phi : \vec{\mathcal{M}} \times \mathcal{R} \rightarrow \{0, 1\}$ *is a constraint indicator that specifies the modification sequences that can be applied to a goal recognition model.*

We assume all modifications have uniform cost and the cost of a sequence is equal to its length ($\mathcal{C}(\vec{m}) = |\vec{m}|$). Also, we assume a valid modification sequence cannot have an invalid prefix.

We let $app(R)$ represent the set of modifications that are applicable in R, *i.e.*, all modifications m such that $\phi(m, R) = 1$. In addition, we let $R^{\vec{m}}$ represent the result of applying a valid modification sequence $\vec{m}$ to model R. Applying any modification to $R_\emptyset$ results in an invalid model, *i.e.*, $\forall m \in \mathcal{M}$, $\delta(m, R_\emptyset) = R_\emptyset$.

Finally, we define a goal recognition design model.

Definition 3 *A goal recognition design (*GRD*) model is given by the pair $T = \langle R_0, D \rangle$ where*

- R_0 *is an initial goal recognition model, and*
- D *is a* design model.

The design model D imposes a set $\mathcal{R}^T \subseteq \mathcal{R}$ of goal recognition models reachable from the initial model R_0 by applying a valid modification sequence.

Given a *GRD* model $T = \langle R_0, D \rangle$, our objective is to find a modification sequence $\vec{m}^* \in \vec{\mathcal{M}}$ to apply to R_0 that minimizes *wcd* under the constraints specified by ϕ. We let $wcd^{min}(T)$ represent the minimal *wcd* achievable in T when applying an optimal modification sequence and $R_0^{\vec{m}}$ represent the model that results from applying $\vec{m}$ to R_0. The objective of the *GRD* problem is formulated as follows.

$$wcd^{min}(T) = \underset{\vec{m} \in \vec{\mathcal{M}} | \phi(\vec{m}, R_0) = 1}{\text{minimize}} \quad wcd(R_0^{\vec{m}}) \tag{1}$$

In particular, given the minimal *wcd* possible, we prefer solutions with minimal length.

The Redesign Process

We view the model redesign process as a search in the space of modification sequences $\vec{m} \in \vec{\mathcal{M}}$ (and their corresponding models $R_0^{\vec{m}}$) for a sequence that minimizes *wcd*. The operators are the modifications $\mathcal{M}$ that transition between goal recognition models.

A basic *wcd* reduction method uses a breadth first search (BFS), possibly replaced with a Dijkstra-based exploration in the case of non-uniform modification cost. The root node of the search is the empty sequence $\vec{m}_\emptyset$ and the initial model R_0. Each successor node appends a single modification to the sequence applied to the parent node. If the sequence is valid, it is added to a queue of sequences. For each explored node we calculate *wcd*, updating the minimal *wcd* and optimal modification sequence when needed. The search continues, increasing at each level of the tree the size of applied

modification sequences until a model in which $wcd = 0$ is found or until there are no more nodes to explore. The result is a modification sequence $\vec{m}^{min} \in \vec{\mathcal{M}}$ that minimizes the *wcd*.

In addition to satisfying the main objective of minimizing *wcd* under the specified constraints, this iterative approach fulfills our secondary requirement, ensuring finding a modification sequence of minimal length.

A key question involves the modifications to be considered at each stage. A naïve approach is to consider all valid combinations of modifications, which we refer to as *exhaustive-reduce*. In the worst case, *exhaustive-reduce* examines all valid modification sequences.

We let $\vec{\Pi}^{wcd}(R) \subseteq \vec{\Pi}^{nd}(R)$ represent the set of non-distinctive paths with maximal length. Seeking improvement through pruning, Lemma 1 shows that *wcd* of a model cannot decrease if at least one non-distinctive path in $\vec{\Pi}^{wcd}(R)$ remains non-distinctive in the modified model.

Lemma 1 *Given a goal recognition design model T and goal recognition models $R, R' \in \mathcal{R}^T$, if $\exists \vec{\pi} \in \vec{\Pi}^{wcd}(R)$ s.t. $\vec{\pi} \in \vec{\Pi}^{nd}(R')$ then* $\text{wcd}(R) \leq \text{wcd}(R')$.

Proof: The *wcd* of a model is the maximal cost over non-distinctive paths $\vec{\Pi}^{nd}(R)$, which is the cost of $\vec{\pi}$. If $\vec{\pi}$ is non-distinctive in R' the *wcd* is at least the cost of $\vec{\pi}$ and $wcd(R) \leq wcd(R')$. ∎

The *pruned-reduce* algorithm (Algorithm 1), presented first by Keren, Gal, and Karpas (2014) and extended here to partially observable settings, exploits Lemma 1 to avoid unnecessary computations by pruning sequences $\vec{m} \cdot m$ that are guaranteed not to have an effect on the wcd *plans* of a model $\Pi^{wcd}(R_0^{\vec{m}_{cur}})$, which are a pair of complete plans that lead to two different goals, and at least one has a *wcd* path $\vec{\pi} \in \vec{\Pi}^{wcd}(R_0^{\vec{m}_{cur}})$ as its prefix while the other has a prefix that shares an observable projection with $\vec{\pi}$.

Typically, the *wcd* calculation of a goal recognition model (performed for example using the methods of Keren, Gal, and Karpas (2014; 2016a; 2016b)) yields a pair of *wcd* plans. The effect of a modification on a path is characterized by the set of *affected actions* $A(m, R)$. Action a belongs to $A(m, R)$ either because m changes its implementation (by changing $pre(a)$, $add(a)$ or $del(a)$) or its observability (by changing the set of tokens a may emit). The *pruned-reduce* algorithm prunes modifications m for which the set of affected actions $A(m, R_0^{\vec{m}_{cur}})$ includes no action that belongs to $\Pi^{wcd}(R_0^{\vec{m}_{cur}})$ (Line 11).

Algorithm 1 *pruned-reduce*($T = \langle R_0, D \rangle$)

```
1: wcd^min = ∞ (init)
2: m⃗^min = m⃗_∅ (init with empty sequence)
3: Create a queue Q initialized to m⃗_∅
4: while Q is not empty: do
5:    m⃗_cur ← Q.dequeue()
6:    if wcd(R_0^{m⃗_cur}) < wcd^min then
7:       wcd^min = wcd(R_0^{m⃗_cur})
8:       m⃗^min = m⃗_cur
9:    end if
10:   for all m ∈ M_D do
11:      if {φ(m⃗_cur · m, R_0) = 1} and
         { ∃a ∈ A(m, R_0^{m⃗_cur}) s.t. a ∈ Π^wcd(R_0^{m⃗_cur})}
         then
12:         enqueue m⃗_cur · m onto Q
13:      end if
14:   end for
15: end while
16: return m⃗^min
```

Safe Pruning for *GRD* using Generalized Strong Stubborn Sets

To justify the pruning performed by *pruned-reduce* we observe it can be viewed as a form of partial order reduction used to reduce the size of the search space. Specifically, we show that for every node the modifications not pruned form a strong stubborn set (Valmari 1989), a subset of modifications that include the first modification in a sequence that minimizes the *wcd* of the goal recognition model represented by the node.

Specifically, we use the formulation of Generalized Strong Stubborn Sets (GSSS) (Wehrle and Helmert 2014)ł which considers planning tasks and adopt it to our optimization task of finding a valid modification sequence to apply to the initial goal recognition model to minimize the *wcd*.

We consider a modification sequence $\vec{m}$ to be *strongly optimal* for model R if it is a minimal length sequence that minimizes the *wcd* of R. A *terminal node* either minimizes the *wcd* or has no valid successor modifications. The set $Opt(T)$ represents the strongly optimal solutions for model R_0 in T. Finally, a successor function $\succ$: $\mathcal{R} \rightarrow 2^{\mathcal{M}}$ yields for every node R a set of successor modifications.

We let $\succ_{pr}$ represent the successor function of the *pruned-reduce* algorithm, that yields for every node R a set of successor modifications $\succ_{pr}(R) \subseteq app(R)$. According to Algorithm 1, $\succ_{pr}$ prunes transitions that either violate the constraints or have no effect on the set of *wcd* plans of the current model.

Since we want to minimize *wcd*, we need to make sure our pruning method is *safe*, *i.e.*, it is guaranteed that an optimal solution for the given goal recognition design task can still be found in the pruned search tree. Specifically, a successor function is safe if for every non-terminal model R, $\succ_{pr}(R)$ includes at least one operator that starts an optimal solution for R. As described by Wehrle and Helmert (2014), assuming all modifications have non-zero uniform cost, this is a necessary criterion for safety.

Given a *GRD* model T, we let T_R represent the model that is the same as T but with R as its initial goal recognition model. The set $\succ_{pr}(R)$ is a GSSS in R if it complies with three requirements. First, for every modification $m \in \succ_{pr}(R)$ that is not applicable in R, the set contains a *necessary enabling set* for m and $Opt(T_R)$, which are the modifications that are applied before m in all sequences in

$Opt(T_R)$ in which m is applied. Second, for every modification $m \in \succ_{pr}(R)$ that is applicable in R, the set contains all modifications that *interfere* with m in R. Two modifications $m_1, m_2 \in \mathcal{M}$ interfere in R if they either *disable* or *conflict* in R. Modification m_1 disables m_2 in R if both are applicable in R but $m_2 \notin app(\delta(m_1, R))$. Two applicable modifications conflict in R if applying them in any order to R is valid but yields different non-distinctive paths (*i.e.*, $\vec{\Pi}^{nd}(R^{m_1,m_2}) \neq \vec{\Pi}^{nd}(R^{m_2,m_1})$). The final condition requires that $\succ_{pr}(R)$ contains at least one modification from at least one strongly optimal solution for T_R.

Note that while the original formulation by Wehrle and Helmert (2014) requires non-interfering operators to yield the same state for different orders of application, we only require them to yield the same set of non-distinctive paths which is enough to guarantee both models share the same *wcd* value. Also, the original formulation of GSSSs considers an envelope of modifications that are sufficient to consider at each node. As can be seen in Line 10 of Algorithm 1, all modifications are considered at every step setting the envelope, hereon omitted from our description, to be the set of all modifications. Considering tighter envelopes that allow more pruning are beyond the scope of this paper but offer promising extensions for future work.

The definition of GSSSs guarantees that for every non-terminal node, at least one permutation of a strongly optimal solution is not pruned. Note that verifying a given set is a GSSS does not require complete knowledge of the sets of strongly optimal solutions. If the specified conditions can be verified for an over-approximation of the sets, they hold for the actual sets. Accordingly, we show that the pruning performed by the *pruned-reduce* uses an over-approximation of the optimal solutions for every explored node.

Theorem 1 ensures that successor pruning based on GSSSs is safe.

Theorem 1 *Let $\succ$ be a successor pruning function defined as $succ(R) = G(R) \cap app(R)$, where $G(R)$ is a generalized strong stubborn set in R. Then $\succ$ is safe.*

The proof is omitted due to space considerations but follows that of (Wehrle and Helmert 2014).

Independent, Persistent, Monotonic-nd *GRD* Models

We are now ready to present a characterization of a class of *GRD* models that can take advantage of the GSSS characterization and perform effective pruning, using Algorithm 1. Clearly, to make use of GSSSs, one must provide an efficient method for their computation. We show that in this case the computation of a GSSS is given with a low computational overhead, as part of the *wcd* computation (Line 6).

A useful observation here is that a *GRD* model and its constraint function induce a space of valid modification sequences. To guarantee $\succ_{pr}$ is safe *i.e.*, that the space state induced by $\succ_{pr}$ includes a strongly optimal sequence, we require the *GRD* model to be *independent*, *persistent*, and *monotonic-nd*. Independence ensures no two modifications interfere. Persistence ensures model constraints under modification permutation. Finally, monotonicity ensures no new non-distinctive paths are added as a result of model modification. After introducing these characterizations formally we show they are sufficient to guarantee the safety of $\succ_{pr}$.

Definition 4 (independent model) *A* GRD *model is* independent *if for any modification $m \in \mathcal{M}$:*
- *the necessary enabling set of m and $Opt(T)$ is empty, and*
- *there are no modification m' and goal recognition model R s.t. m' interferes with m in R.*

Definition 5 (persistent model) *A* GRD *model T is* persistent *if for any goal recognition model $R \in \mathcal{R}_T$ and modification sequences $\vec{m}, \vec{m}' \in \vec{\mathcal{M}}$:*
- *If $\vec{m}$ is a prefix of $\vec{m}'$, and $\phi(\vec{m}, R) = 0$ then $\phi(\vec{m}', R) = 0$*
- *If $\vec{m}'$ is a permutation of $\vec{m}$ and $\phi(\vec{m}, R) = 1$ then $\phi(\vec{m}', R) = 1$.*

Persistence and independence are sufficient for pruning invalid sequences or sequences that are guaranteed to have a permutation that is not pruned. To prune modifications that do not affect the pair of *wcd* plans when no new non-distinctive paths may be added to the model, we define *monotonic-nd* models as follows.

Definition 6 (monotonic-nd model) *A* GRD *model T is* monotonic-nd*, if for any goal recognition model $R \in \mathcal{R}_T$ and modification $m \in \mathcal{M}$, $\vec{\Pi}^{nd}(R^m) \subseteq \vec{\Pi}^{nd}(R)$.*

In a monotonic-nd model valid modifications may only remove paths from the set of non-distinctive paths. Therefore, applying them cannot increase *wcd*.

Lemma 2 *Given a* GRD *model T, if T is monotonic-nd then for every goal recognition model $R \in \mathcal{R}^T$ and modification $m \in \mathcal{M}$ s.t. $\phi(m, R) = 1$,*

$$\text{wcd}(R^m) \leq \text{wcd}(R)$$

Proof: According to Definition 6, modifications in a monotonic-nd model do not add non-distinctive paths. In particular, there are no non-distinctive paths with a cost higher than $wcd(R)$ that are added to the model and $wcd(R^m) \leq wcd(R)$. ∎

In a monotonic-nd model, applying a modification that does not modify the pair of *wcd* plans of the current model leaves the *wcd* unchanged.

Corollary 1 *Let T be a monotonic-nd* GRD *model. For every modification $m \in \mathcal{M}$ and goal recognition model $R \in \mathcal{R}_T$ s.t. $\phi(m, R) = 1$, if $\forall a \in A(m, R)$ $a \notin \Pi^{\text{wcd}}(R)$ then* $\text{wcd}(R^m) = \text{wcd}(R)$.

Proof: According to Lemma 2, in a monotonic-nd *GRD* model *wcd* cannot increase as a result of applying a

valid modification, *i.e.*, $wcd(R^m) \leq wcd(R)$. Since no action in the pair of plans $\Pi^{wcd}(R)$ is affected by m, both remain applicable in R^m and the *wcd* path $\vec{\pi}^{wcd}(R)$ remains non-distinctive in R^m. Moreover, since T is monotonic-nd no non-distinctive paths are added to the model. This, according to Lemma 1, means that the *wcd* cannot decrease *i.e.*, $wcd(R^m) \geq wcd(R)$, leaving *wcd* unchanged. ■

Finally, we specify the conditions under which the *pruned-reduce* algorithm is guaranteed to produce an optimal solution, or alternatively, the conditions under which $\succ_{pr}$ yields a GSSS for every model encountered in the search.

Theorem 2 *Given a* GRD *model $T = \langle R_0, D\rangle$ where R_0 is the initial model and $D = \langle \mathcal{M}, \delta, \phi\rangle$, if T is independent, persistent, and monotonic-nd then for every model $R \in \mathcal{R}_T$, the set $\succ_{pr}(R)$ is a GSSS.*

Proof (sketch) The proof, for which the complete version can be found in an extended techincal report (Keren, Gal, and Karpas 2018), shows that under the specified assumptions, for every strongly optimal sequence that is pruned, we can use the method suggested by Wehrle and Helmert (2014) to construct a strongly optimal sequence that is not pruned by $\succ_{pr}$. □

Theorem 1 and Theorem 2 guarantee that for independent, persistent, and monotonic-nd *GRD* models $\succ_{pr}$ generates a GSSS for every node $R \in \mathcal{R}_T$, which means that $\succ_{pr}$ is safe and the search will find an optimal modification sequence.

As a final note we observe that Algorithm 1 can be extended to support non-independent models in which case modifications that interfere with or enable modifications that affect the *wcd* plans are not pruned. In the next section we focus on creating a class of modifications that extends those proposed in previous work while complying with the specified requirements.

New Modifications for Independent, Persistent, Monotonic-nd *GRD* Models

Equipped with the conditions under which the pruning performed by the *pruned-reduce* algorithm is safe we now describe a general design model $D = \langle \mathcal{M}, \delta, \phi\rangle$ that complies with these conditions. We do this by specifying the constraints (ϕ) and modifications ($\mathcal{M}$) and prove their compliance with the requirements. In particular, we extend the state-of-the-art in *GRD* by offering new redesign modification types while preserving the completeness of the *pruned-reduce* algorithm.

We assume agents follow bounded sub-optimal paths, which means they have a limited budget for diverting from optimal behavior (or no diversion budget if they are optimal). The constraint function is *budget preserving*, enforcing a design budget that limits the number of modifications that can be applied. Such a budget can either be a limit on the overall number of modifications or a separate budget for each modification type. The constraint function is also *cost preserving*, requiring that the maximal cost of a legal plan to any of the goals $g \in G$ does not increase. Note that for settings where bounded sub-optimal paths are legal, existing methods for *wcd* calculation, e.g. (Keren, Gal, and Karpas 2014; 2016b), reveal the maximal cost to goal, meaning this condition can be verified with no overhead cost.

The modification set consists of four modification types, two of which were already presented in the literature, namely *action removal* (Keren, Gal, and Karpas 2014; Wayllace et al. 2016) and *sensor placement* (Keren, Gal, and Karpas 2015). We next present, in addition, *action conditioning* and *single action sensor refinement*. Letting $\mathcal{A}$ denote the set of all actions and $pre_R(a)$ the preconditions of action a in model R.

Definition 7 *A modification m is an* action conditioning *modification if for every goal recognition model $R \in \mathcal{R}$, R^m is identical to R except that for every action $a \in \mathcal{A}$ $pre_R(a) \subseteq pre_{R^m}(a)$.*

When conditioning (*i.e.*, adding preconditions to) an action from the model, some paths in the model may become invalid. Specifically, action conditioning can be used to disallow specific permutations of paths by forcing a partial order between actions. *Action removal* is an extreme special case, where actions are removed from the model. Removing an action is equivalent to adding an unsatisfiable precondition.

Action conditioning may potentially decrease *wcd* by removing non-distinctive paths. In general, however, action conditioning may increase the optimal cost to a goal (and the *wcd*) by eliminating all optimal paths that lead to it. We show that when the constraint function is cost-preserving, action conditioning modifications cannot add non-distinctive paths.

Lemma 3 *Let $T = \langle R_0, D\rangle$ be a* GRD *model. If ϕ is cost preserving, then for any action conditioning modification $m \in \mathcal{M}$ and model $R \in \mathcal{R}_T$, $\vec{\Pi}^{nd}(R^m) \subseteq \vec{\Pi}^{nd}(R)$.*

Proof: Let $\Pi^*_R(g)$ represent the optimal plans to goal g in R. Assume, by way of contradiction, that there is an action conditioning modification m s.t. there exists a goal recognition model R with a path $\vec{\pi} \in \vec{\Pi}^{nd}(R^m)$ but $\vec{\pi} \notin \vec{\Pi}^{nd}(R)$ (non-distinctive in R^m but distinctive in R). This implies there are at least two optimal plans $\pi' \in \Pi^*_{R^m}(g')$ and $\pi'' \in \Pi^*_{R^m}(g'')$ to two different goals $g' \neq g''$ one of which has $\vec{\pi}$ as its prefix and the other shares the observable projection of $\vec{\pi}$ in R^m but not in R (w.l.o.g we say that $\vec{\pi}$ is a prefix of π').

Under the assumption that m is an action conditioning modification, it is easy to see that both plans are valid in R (*i.e.*, $\pi' \in \Pi_R(g')$ and $\pi'' \in \Pi_R(g'')$). Moreover, since ϕ is cost preserving we know that both plans are optimal in R ($\pi' \in \Pi^*_R(g')$ and $\pi'' \in \Pi^*_R(g'')$). The sensor model of R and R^m is the same. Therefore, $\vec{\pi}$ is a prefix of π' and shares an observable projection with a prefix of π'' which means $\vec{\pi}$ is non-distinctive in R, thus contradicting our choice of $\vec{\pi}$. ■

The second newly introduced modification type is *single action sensor refinement*, which sets the observability of a

single action. We let $A_S[a] \subseteq \mathcal{A}$ represent the set of actions (excluding action a) that share an observation token with action a according to sensor model S and define single action sensor refinement as follows.

Definition 8 *A modification m is a* single action sensor refinement *modification if for every goal recognition model $R \in \mathcal{R}$, R and R^m are identical except that:*
(1) for every action $a \in \mathcal{A}$, if $o_\emptyset \in S_{R^m}(a)$ then $o_\emptyset \in S_R(a)$ and
(2) there exists an action $a_m \in \mathcal{A}$ s.t. for every action $a \in \mathcal{A}$ $a \neq a_m$, $A_{S_{R^m}}[a] = A_{S_R}[a] \setminus a_m$.

Note that the token set of a given action can include the empty token in the refined model, only if it was included in the original model. Single action refinement (hereon referred to as sensor refinement) modifications assign a unique token (or a set of tokens) to one of the actions, previously mapped to a set of other tokens.

A special case of sensor refinement is *sensor placement* (Keren, Gal, and Karpas 2016a) which *exposes* a non-observable action (previously mapped to the null token) by mapping it to a unique token. To show sensor refinement (and placement) modifications never add non-distinctive paths we show that for every path, the number of paths that may share its observable projection cannot increase due to sensor refinement.

Lemma 4 *Given a sensor refinement modification m and two goal recognition models R and R', if $R' = m(R)$ and $\phi(m, R) = 1$ then $\vec{\Pi}^{nd}(R') \subseteq \vec{\Pi}^{nd}(R)$.*

Proof (sketch) By definition, sensor refinement only changes the system's sensor model and not the set of legal paths to each goal. For every action a, $A_{S_{R^m}}[a] \subseteq A_{S_R}[a]$ (the set of actions that share a token with a is reduced in the modified model). Consequently, the set of paths that share an observable projection with $\vec{\pi}$ in R^m is a subset of the paths that share observability with $\vec{\pi}$ in R. Therefore, a path which is distinctive in R cannot become non-distinctive in R^m.

□

Before concluding our description we show that our model complies with the specified requirements that guarantee the safety of $\succ_{pr}$. We start by showing that a model with only action conditioning and sensor refinement modifications and a cost-preserving constraint function is guaranteed to be monotonic-nd.

Lemma 5 *Given a goal recognition design model $T = \langle R_0, D\rangle$ where R_0 is the initial model and $D = \langle \mathcal{M}, \delta, \phi\rangle$, if ϕ is cost-preserving and the modification set $\mathcal{M}$ consists only of action conditioning and sensor refinement modifications, then T is monotonic-nd.*

Proof: According to Lemma 3 and assuming the constraint function ϕ is cost preserving, action conditioning modifications never add paths to the model and non-distinctive paths in particular. Corollary 4 shows this is true for sensor refinement modifications in any goal recognition design model. Therefore, for any modification $m \in \mathcal{M}$ and model $R \in \mathcal{R}_T$ $\vec{\Pi}^{nd}(R^m) \subseteq \vec{\Pi}^{nd}(R)$ and T is monotonic-nd according to Definition 6. ■

Next, we show that when the modifications set $\mathcal{M}$ consists only of action conditioning and sensor refinement modifications, the model is independent.

Lemma 6 *Let $T = \langle R_0, D\rangle$ be a goal recognition design model where R_0 is the initial goal recognition model and $D = \langle \mathcal{M}, \delta, \phi\rangle$ is the design model. If $\mathcal{M}$ consists only of sensor refinement and action conditioning modifications, then T is independent.*

Proof (sketch) According to Definition 7 and 8 both action conditioning and sensor refinement modifications are applicable at any model $R \in \mathcal{R}_T$ and do not affect the applicability of other modifications. According to Definition 4, to show T is independent we are left with the need to show there are no modifications that conflict in some model R.

Assume to the contrary that $\exists R \in \mathcal{R}_T$ and modifications m_1 and m_2 s.t. the modifications conflict in R s.t. $\vec{\Pi}^{nd}(R^{m_1,m_2}) \neq \vec{\Pi}^{nd}(R^{m_2,m_1})$.

W.l.o.g we let $\vec{\pi}$ represent a path that belongs to $\vec{\Pi}^{nd}(R^{m_1,m_2})$ but not to $\vec{\Pi}^{nd}(R^{m_2,m_1})$. Because $\vec{\pi}$ is non-distinctive in R^{m_1,m_2} we know that there exists a path $\vec{\pi}'$ s.t. $\vec{\pi}$ and $\vec{\pi}'$ share an observable projection that satisfies at least two goals in R^{m_1,m_2} but not in R^{m_2,m_1} (otherwise $\vec{\pi}$ would be non-distinctive in both models). This can happen in one of two cases: either both $\vec{\pi}$ and $\vec{\pi}'$ are valid in R^{m_1,m_2} but one of them is invalid in R^{m_2,m_1} or the two paths, share an observation sequence in R^{m_1,m_2} but not in R^{m_2,m_1}. We show that the first case cannot happen since R^{m_1,m_2} and R^{m_2,m_1} have the same set of actions and valid paths. The second case cannot happen since action conditioning modifications do not change the sensor model, and for every action the set of possible tokens is the same in both $A_{S_{R^{m_1,m_2}}}$ and $A_{S_R}[a_i] \setminus a_{m_2} \setminus a_{m_1}$. Accordingly, the set of possible observable projections of every path is the same in both models, and any path that is non-distinctive in R^{m_1,m_2} and belongs to $\vec{\Pi}^{nd}(R^{m_1,m_2})$ is also non distinctive in R^{m_2,m_1} (and belongs to $\vec{\Pi}^{nd}(R^{m_2,m_1})$), thus contradicting our choice of $\vec{\pi}$ and concluding our proof. □

The third condition requires that our goal recognition design model is persistent. The persistency of a model depends on the definition of its constraint function.

Lemma 7 *Given a goal recognition design model $T = \langle R_0, D\rangle$ where R_0 is the initial model and $D = \langle \mathcal{M}, \delta, \phi\rangle$, if ϕ is cost-preserving and budget preserving and the modifications set $\mathcal{M}$ consists only of action constraining and sensor refinement modifications, then T is persistent.*

Proof: To show the model is persistent, we consider a sequence $\vec{m}$ in T. If $\vec{m}$ is pruned because it violates the design

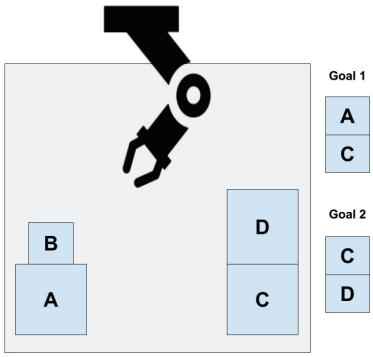

Figure 1: An example of a goal recognition design problem

budget then adding modifications is not allowed and there is no sequence that has $\vec{m}$ as its prefix that is valid. If $\vec{m}$ is valid, the sequence and all its permutations respect the budget constraints.

According to Definition 7 the set of actions affected by a modification is the same for any model $R \in \mathcal{R}$ (for sensor refinement the preconditions of all action are unchanged). Therefore, if sequence $\vec{m}'$ is a permutation of $\vec{m}$ it will have the same set of paths and optimal paths in particular. If $\vec{m}$ maintains the optimal cost to all goal, so will any of its permutations $\vec{m}'$. Similarly, if $\vec{m}$ causes the optimal cost of one of the goals to increase, appending modifications to $\vec{m}$ will not decrease the costs and therefore any sequence that has $\vec{m}$ as its prefix will be invalid, thus concluding our proof that T is persistent under the specified conditions. ■

Finally, we combine the ingredients specified above to describe a goal recognition design model for which the *pruned-reduce* algorithm is guaranteed to an optimal modification sequence.

Lemma 8 *Given a goal recognition design model $T = \langle R_0, D\rangle$ where R_0 is the initial model and $D = \langle \mathcal{M}, \delta, \phi\rangle$, if ϕ is both cost-preserving and budget-preserving and the modifications set $\mathcal{M}$ includes only action constraining and sensor refinement modifications then $\succ_{pr}(R)$ is safe.*

Proof: According to Theorem 1, if a pruning function is guaranteed to produce a GSSS for every node in the search, it is safe. According to Theorem 2, if T is independent, persistent and monotonic-nd the set $\succ_{pr}(R)$ is a GSSS for every model $R \in \mathcal{R}_T$. Lemma 5 guarantees that under the specified constraints T is monotonic-nd, Lemma 6 guarantees it is independent and finally Lemma 7 concludes our proof by guaranteeing the model is persistent. ■

Example 1 *As an example of a controllable environment consider Figure 1 which depicts a variation of the well known BlockWords domain where a robot uses a gripper to move blocks around in order to achieve one of two possible configurations. In our setting the blocks and gripper are hidden from the observer who knows the initial setting (depicted on the left) and possible goals (depicted on the right), but only knows when some action is performed. Also, to lift a box the gripper is adjusted to the width of the box.*

In the original setting, the only way to recognize the goal is by counting the number of actions performed, setting wcd $= 3$ *as the cost of the optimal path to* Goal 1. *By placing a sensor on* Block B *(indicating when the block is picked up by the gripper) the* wcd *is reduced to* 1 *since an optimal plan to both goals can start with placing* Block D *on the table. If moving* Block B *is the next action performed then the goal is* Goal 1 *(and* Goal 2 *otherwise). If, in addition to the sensor, the robot movement is limited so it can only expand its gripper, the only way to achieve* Goal 1 *is by first lifting* Block B. *In this case* $wcd = 0$ *and the first action reveals the goal. We cannot disallow moving any of the blocks while guaranteeing both goals can be achieved, making it necessary to use action conditioning rather than action removal to minimize the* wcd.

	FO				PO			
	Exhaustive		Pruned		Exhaustive		Pruned	
	Solved	Expanded	Solved	Expanded	Solved	Expanded	Solved	Expanded
GRID	48	135.25	60	48.5	132	137.42	160	52.57
GRID+	71	161.25	72	36.75	63	212.66	68	50.04
BLOCK	15	136.1	20	17.6	32	136.75	48	21.33
LOG	43	194.6	44	9.66	19	580.01	22	13.66
ISS	60	40.0	120	6.67	74	68.75	224	56.63

Table 1: Efficiency gain of pruning

	FO			PO			
	AR	AC	TOT	AR	AC	SR	TOT
GRID	0.81	0.56	0.81	1.66	0.4	1.02	1.88
GRID+	0.57	0.28	0.57	4.25	4.25	6.25	6.25
BLOCK	0.5	0	0.5	0.47	0	1.42	1.42
LOG	0	0	0	0	0	0.33	0.33
ISS	0	0	0	0	0	2.92	2.92

Table 2: Comparing modification methods

Empirical Evaluation

The objective of our empirical evaluation is twofold. First, we evaluate the efficiency gain brought by pruning by comparing the *exhaustive-reduce* and *pruned-reduce* methods for redesign. Our second objective is to evaluate the *wcd* reduction achievable with the various modifications. We first describe the datasets setup before presenting and discussing the results.

Datasets: We used four uniform cost plan recognition domains (Ramirez and Geffner 2009; Keren, Gal, and Karpas 2016b) namely: GRID-NAVIGATION (GRID), IPC-GRID$^+$ (GRID+), BLOCK-WORDS (BLOCK), and LOGISTICS (LOG)[1] - all based on PDDL domains from the deterministic track of the International Planning Competition. For non-uniform action costs we evaluated the ISS-CAD (ISS) domain (E-Martín, R-Moreno, and Smith 2015), which describes a space exploration setting where a robot tries to recognize the goal of an astronaut performing spaceship maintenance tasks. We considered two system observability settings (*i.e.*, two system sensor models) : Fully Observable (FO) and Partially Observable (PO) where actions

[1]The details of the dataset generation process are detailed in (Keren, Gal, and Karpas 2018).

are each mapped to a set of tokens. We evaluated 210 GRID, 168 GRID+, 216 BLOCK, 180 LOG and 354 ISS problems (we used different problems for FO and PO).

Setup: We implemented three modification methods, namely action removal (AR), action conditioning (AC), and single action sensor refinement (SR). Action conditioning was implemented by enforcing an order between a pair of actions, while action removal disallows the action altogether. Each problem is run with both the *exhaustive-reduce* and *pruned-reduce* algorithms, with a design budget of 4 assigned once for each modification method and once as an overall budget. The optimal cost to any of the goals could not increase. During execution, we kept track of intermediate results, allowing us to examine the maximal reduction achieved by each budget level. We used the Fast Downward planning system (Helmert 2006) running A^* with the LM-CUT heuristic (Helmert and Domshlak 2009) for all but the ISS domain for which the IPDB heuristic (Haslum et al. 2007) was used. Experiments were run on Intel(R) Xeon(R) CPU X5690 machines, with a time limit of 30 minutes and memory limit of 2 GB.

Results: Table 1 compares the pruning performed by the *exhaustive-reduce* and *pruned-reduce* algorithms. For each method and setting, the table shows the number of problems solved to completion within the time bound (*Solved*) and for those solved by both methods, the average number of modification sequences expanded by each method (*Expanded*). The evaluation shows that for all domains *pruned-reduce* solves more problems and expands less nodes then the exhaustive search, showing the efficiency gain brought by pruning. This computational gain is emphasized in domains with large branching factors, such as LOG and ISS where *pruned-reduce* examines less then 10% of the nodes examined by the exhaustive approach.

Table 2 shows the *wcd* reduction achieved by each modification method separately and a combination of all modifications (TOT) for problems completed for all design options. The results indicate the ability to reduce *wcd* by redesign, achieving an average *wcd* reduction of more that 6 in the GRID+ domain. For most of the settings a specific type of modification achieves best results, except for PO-GRID where combining modifications yields higher results then the individual methods. For both GRID and GRID+ the newly presented action conditioning managed to achieving the same *wcd* reduction as action removal for PO GRID+ settings, indicating its benefit to *wcd* minimization in settings where action removal cannot be applied.

Related Work

GRD, which is a special case of environment design (Zhang, Chen, and Parkes 2009), was first introduced by Keren et al. (2014) and later extended (Keren, Gal, and Karpas 2015; Son et al. 2016; Keren, Gal, and Karpas 2016a; 2016b; Wayllace et al. 2016) by offering tools to analyze a variety of *GRD* settings. Common to all previous work on *GRD* is the ad-hoc account of the design process, providing optimality guarantees to the specific model and the modifications discussed. We are the first to provide a general framework for characterizing goal recognition models for which partial order reduction in the form of strong stubborn sets can be used to improve the design process. Specifically, it allows us to define new classes of modifications that can be used to enhance recognition.

Conclusion

We frame existing pruning methods in the context of strong stubborn sets, allowing us to enrich the *GRD* framework with new modifications while preserving the completeness of the pruning performed. Our empirical evaluation showed both the efficiency gains of pruning and the *wcd* reduction achieved.

In future work we intend to explore new ways to enhance the design process by combining different search space reduction techniques. Also, while our evaluation used existing goal recognition benchmarks, we intend to enrich the *GRD* dataset with new settings for which design is relevant.

References

Boddy, M. S.; Gohde, J.; Haigh, T.; and Harp, S. A. 2005. Course of action generation for cyber security using classical planning. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS 2005)*.

E-Martín, Y.; R-Moreno, M. D.; and Smith, D. E. 2015. Practical goal recognition for iss crew activities. In *Proceedings of the International Workshop of Planning and Scheduling for Space (IWPSS 2015)*.

Fikes, R. E., and Nilsson, N. J. 1972. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence* 2(3):189–208.

Haslum, P.; Botea, A.; Helmert, M.; Bonet, B.; Koenig, S.; et al. 2007. Domain-independent construction of pattern database heuristics for cost-optimal planning. In *Proceedings of the Conference of the American Association of Artificial Intelligence (AAAI 2007)*.

Helmert, M., and Domshlak, C. 2009. Landmarks, critical paths and abstractions: What's the difference anyway? In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS 2009)*.

Helmert, M. 2006. The fast downward planning system. *J. Artif. Intell. Res. (JAIR)* 26:191–246.

Jarvis, P. A.; Lunt, T. F.; and Myers, K. L. 2004. Identifying terrorist activity with ai plan recognition technology. In *Proceedings of the National Conference on Innovative Applications of Artificial Intelligence (IAAI 2004)*.

Kaluza, B.; Kaminka, G. A.; and Tambe, M. 2011. Towards detection of suspicious behavior from multiple observations. In *AAAI Workshop on Plan, Activity, and Intent Recognition (PAIR 2011)*.

Kautz, H.; Etzioni, O.; Fox, D.; Weld, D.; and Shastri, L. 2003. Foundations of assisted cognition systems. *Technical Report: University of Washington, Computer Science Department*.

Keren, S.; Pineda, L.; Gal, A.; Karpas, E.; and Zilberstein, S. 2017. Equi-reward utility maximizing design in stochas-

tic environments. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2017)*.

Keren, S.; Gal, A.; and Karpas, E. 2014. Goal recognition design. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS 2014)*.

Keren, S.; Gal, A.; and Karpas, E. 2015. Goal recognition design for non optimal agents. In *Proceedings of the Conference of the American Association of Artificial Intelligence (AAAI 2015)*.

Keren, S.; Gal, A.; and Karpas, E. 2016a. Goal recognition design with non oservable actions. In *Proceedings of the Conference of the American Association of Artificial Intelligence (AAAI 2016)*.

Keren, S.; Gal, A.; and Karpas, E. 2016b. Privacy preserving plans in partially observable environments. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2016)*.

Keren, S.; Gal, A.; and Karpas, E. 2018. Goal recognition design for optimal agents. In *Technical report: IE/IS-2018-02, Technion Israel Institute of Technology (March 2018)*.

Levine, S. J., and Williams, B. C. 2014. Concurrent plan recognition and execution for human-robot teams. In *Proceedings of the Conference on Automated Planning and Scheduling (ICAPS 2014)*.

Ramirez, M., and Geffner, H. 2009. Plan recognition as planning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2009)*.

Son, T. C.; Sabuncu, O.; Schulz-Hanke, C.; Schaub, T.; and Yeoh, W. 2016. Solving goal recognition design using asp. In *Proceedings of the Conference of the American Association of Artificial Intelligence (AAAI 2016)*.

Valmari, A. 1989. Stubborn sets for reduced state space generation. In *International Conference on Application and Theory of Petri Nets*. Springer.

Wayllace, C.; Hou, P.; Yeoh, W.; and Son, T. C. 2016. Goal recognition design with stochastic agent action outcomes. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2016)*.

Wehrle, M., and Helmert, M. 2014. Efficient stubborn sets: Generalized algorithms and selection strategies. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS 2014)*.

Zhang, H.; Chen, Y.; and Parkes, D. C. 2009. A general approach to environment design with one agent. *Morgan Kaufmann Publishers Inc.*