Generalising Planning Environment Redesign

Alberto Pozanco[1*], **Ramon Fraga Pereira**[2*], **and Daniel Borrajo**[1†]
[1]J.P. Morgan AI Research
[2]University of Manchester, UK
{alberto.pozancolancho,daniel.borrajo}@jpmorgan.com
ramon.fragapereira@manchester.ac.uk

Abstract

In *Environment Design*, one interested party seeks to affect another agent's decisions by applying changes to the environment. Most research on planning environment (re)design assumes the interested party's objective is to facilitate the recognition of goals and plans, and search over the space of environment modifications to find the minimal set of changes that simplify those tasks and optimise a particular metric. This search space is usually intractable, so existing approaches devise metric-dependent pruning techniques for performing search more efficiently. This results in approaches that are not able to generalise across different objectives and/or metrics. In this paper, we argue that the interested party could have objectives and metrics that are not necessarily related to recognising agents' goals or plans. Thus, to generalise the task of *Planning Environment Redesign*, we develop a general environment redesign approach that is *metric-agnostic* and leverages recent research on top-quality planning to efficiently redesign planning environments according to any interested party's objective and metric. Experiments over a set of environment redesign benchmarks show that our general approach outperforms existing approaches when using well-known metrics, such as facilitating the recognition of goals, as well as its effectiveness when solving environment redesign tasks that optimise a novel set of different metrics.

Introduction

In *Environment Design* (Zhang, Chen, and Parkes 2009), one interested party (usually referred to as observer) seeks to affect another agent's decisions by applying a minimal set of changes to the environment. Most research on planning environment (re)design has focused on cooperative and adversarial settings where the observer aims to facilitate *Goal or Plan Recognition* (Ramírez and Geffner 2009), i.e., infer the agent's goal or plan as soon as possible. These tasks are known as *Goal Recognition Design* (GRD) (Keren, Gal, and Karpas 2014) and *Plan Recognition Design* (PRD) (Mirsky et al. 2019), respectively, and they have been studied under different interested party objectives, metrics, and observer's capabilities (Keren, Gal, and Karpas 2016b; Kulkarni, Srivastava, and Kambhampati 2019; Shvo and McIlraith 2020), as well as agents' intentions and environment assumptions (Keren, Gal, and Karpas 2021).

Existing research on planning environment design defines a *metric* that is able to assess how long (in terms of action progress, i.e., number of observations) an agent can act in an environment without revealing its intended goal or plan to the observer (Keren, Gal, and Karpas 2021). Optimising these metrics can force the agent's behaviour to be more *transparent*, *ambiguous*, or endow *predictability* (Chakraborti et al. 2019). Most approaches assume that the environment can only be modified by removing actions. So, they search over the space of actions' removal to compute the best set of environment changes for a given metric (Keren, Gal, and Karpas 2021). Since this space is usually intractable, existing approaches devise different heuristics and pruning techniques to perform search efficiently depending on the specific metric to be optimised. This results in *metric-dependent* approaches that are not robust enough to generalise across different environment redesign metrics.

In this paper, we propose a *metric-agnostic* approach to redesign *fully observable* and *deterministic* planning environments. Our main contributions are twofold, as follows:

- *Environment Redesign* has mainly focused on promoting or impeding the recognition of goals/plans. We argue that the interested party could have other objectives that are not necessarily related to identifying the agent's goal/plan. For example, the interested party might want to redesign the environment such that the agent is constrained to follow plans that keep certain relationships with some states. This can be beneficial in many planning settings, such as *Anticipatory Planning* (Burns et al. 2012), *Counterplanning* (Pozanco et al. 2018), *Risk Avoidance and Management* (Sohrabi et al. 2018), or *Planning for Opportunities* (Borrajo and Veloso 2021). Thus, we propose *novel metrics* that can be used to redesign environments for these other settings.
- To generate new environments that optimize our *novel metrics*, as well as existing metrics in the literature, we propose GER, a *General Environment Redesign* approach that employs an anytime *Breadth-First Search* (BFS) algorithm. It exploits recent research on *top-quality* planning (Katz, Sohrabi, and Udrea 2020) to improve efficiency. While previous approaches have also used BFS to explore the space of environment modifications, they

[*]These authors contributed equally.
[†]On leave from Universidad Carlos III de Madrid.

assume the extremes of the spectrum. Keren, Gal, and Karpas (2014) do not assume a *plan-library*, so they have to explore the state space and reason over the quality of the plans and the metric value in the environment induced by the current modifications. This yields very costly approaches that are not able to scale to complex environments with many goals. In contrast, Mirsky et al. (2019)'s approach assumes a hand-crafted plan-library is provided as input, which allows the algorithm to reduce the action's removal space by just considering the actions appearing in the plan-library. We propose a middle-ground approach, in which the action space is pruned by a plan-library that is not explicitly given as input, but computed using *top-quality* planning (Katz, Sohrabi, and Udrea 2020).

We evaluate GER in a set of benchmarks for environment redesign, and show that it outperforms existing approaches (Keren, Gal, and Karpas 2014) (being orders of magnitude faster) in known redesigning tasks such as GRD. We also show its effectiveness when solving environment redesign tasks that optimise a novel set of different metrics.

Background

Planning is the task of devising a sequence of actions (i.e., a *plan*) to achieve a goal state from an initial state (Geffner and Bonet 2013). We follow the formalism and assumptions of the *Classical Planning* setting, and assume that an environment is *discrete*, *fully observable*, and *deterministic*.

A *planning domain* $\mathcal{D}$ is defined as $\langle \mathcal{F}, \mathcal{A} \rangle$, where: $\mathcal{F}$ is a set of *facts*; $\mathcal{A}$ is a set of *actions*, where every action $a \in \mathcal{A}$ has a set of preconditions, *add* and *delete* effects, $pre(a), add(a), del(a)$, and a *positive cost*, denoted as $cost(a)$. We define a *state* $\mathcal{S}$ as a finite set of positive facts $f \in \mathcal{F}$ by following the *closed world assumption*, so that if $f \in \mathcal{S}$, then f is true in $\mathcal{S}$. We also assume a simple inference relation $\models$ such that $\mathcal{S} \models f$ iff $f \in \mathcal{S}$, $\mathcal{S} \not\models f$ iff $f \notin \mathcal{S}$, and $\mathcal{S} \models f_0 \wedge ... \wedge f_n$ iff $\{f_0, ..., f_n\} \subseteq \mathcal{S}$. An action $a \in \mathcal{A}$ is applicable to a state $\mathcal{S}$ iff $\mathcal{S} \models pre(a)$, and it generates a new successor state $\mathcal{S}'$ by applying a in $\mathcal{S}$, such that $\mathcal{S}' = (\mathcal{S} \setminus del(a)) \cup add(a)$.

A *planning problem* $\mathcal{P}$ is defined as $\langle \mathcal{D}, \mathcal{S}_\mathcal{I}, G \rangle$, where: $\mathcal{D}$ is a planning domain as we described above; $\mathcal{S}_\mathcal{I} \subseteq \mathcal{F}$ is the *initial state*; and $G \subseteq \mathcal{F}$ is the *goal state*. A *solution* to $\mathcal{P}$ is a *plan* $\pi = [a_0, a_1, ..., a_n]$ that maps $\mathcal{S}_\mathcal{I}$ into a state $\mathcal{S}$ that holds G, i.e., $\mathcal{S} \models G$. The cost of a plan π is $cost(\pi) = \Sigma\ cost(a_i)$, and a plan π^* is *optimal* (with minimal cost) if there exists no other solution π for $\mathcal{P}$ such that $cost(\pi) < cost(\pi^*)$. We use $h^*(s, G)$ to refer to the cost of an optimal plan of achieving G from s.

We refer to $\Pi(\mathcal{P}, b)$ as the *set of all plans* that solve a planning problem $\mathcal{P}$ *within a sub-optimality bound* b (Katz, Sohrabi, and Udrea 2020). This bound is defined as the cost of a plan π with respect to the cost of an optimal plan π^*, i.e., $b = \frac{\pi}{\pi^*}$. Therefore, $\Pi(\mathcal{P}, 1.0)$ will give us all the optimal plans that solve $\mathcal{P}$, and $\Pi(\mathcal{P}, 1.5)$ will give us all the plans that solve $\mathcal{P}$ within a sub-optimality bound of 1.5. When $b > 1$, plans in $\Pi(\mathcal{P}, b)$ might contain loops, i.e., they visit at least one state more than once. In the rest of the paper, we assume that $\Pi(\mathcal{P}, b)$ only contains loop-less plans (von Tschammer, Mattmüller, and Speck 2022).

Planning Environment Redesign

Planning Environment Redesign is the task in which an interested party (observer) aims to perform off-line modifications to a *planning environment* (or just *environment*) where another agent will be acting, in order to constraint its potential behaviour. Following the formalism of (Keren, Gal, and Karpas 2014), we define a *planning environment* with deterministic actions under fully observability, as follows:

Definition 1. *A* ***planning environment*** *is a tuple* $\mathcal{E} = \langle \mathcal{P}_\mathcal{E} = \langle \mathcal{F}, \mathcal{A}, \mathcal{S}_\mathcal{I} \rangle, \mathcal{G} \rangle$ *where* $\mathcal{F}, \mathcal{A}$ *and* $\mathcal{S}_\mathcal{I}$ *are the same as in a planning problem, and* $\mathcal{G}$ *is a set of possible reachable goals* $\{G_0, G_1, ..., G_n\}$ *that are of interest to either the observer or the agent.*

We define the *planning environment redesign* problem in Definition 2, and its solutions in Definitions 3 and 4.

Definition 2. *A* ***planning environment redesign problem*** *is a tuple* $\mathcal{R} = \langle \mathcal{E}, M_b \rangle$ *where* $\mathcal{E}$ *is the current planning environment, and* M_b *is a metric to be optimised in order to get the new redesigned environment, assuming the agent's behaviour sub-optimality is bounded by a constant* b*.*

This definition is more general than the one in (Keren, Gal, and Karpas 2019; Mirsky et al. 2019), as we include the metric M_b in the definition, making the problem definition *metric-agnostic*. We do not make any assumption on the relation between the observer and the agent, i.e., they could be competing, cooperating, or indifferent.

Definition 3. *A* ***solution*** *to an environment redesign problem* $\mathcal{R} = \langle \mathcal{E}, M_b \rangle$ *is a new redesigned environment* $\mathcal{E}' = \langle \mathcal{P}'_\mathcal{E} = \langle \mathcal{F}, \mathcal{A}', \mathcal{S}_\mathcal{I} \rangle, \mathcal{G} \rangle$ *where* $\mathcal{E}'$ *contains a new set of actions,* $\mathcal{A}'$*, and all goals in* $\mathcal{G}$ *are still reachable using* $\mathcal{P}'_\mathcal{E}$*.*

Although environments could be redesigned by adding or removing any element in $\mathcal{E}$, we follow (Keren, Gal, and Karpas 2019) and (Mirsky et al. 2019), and assume that environments are redesigned through *action removal*. Thus, $\mathcal{A}' = \mathcal{A} \setminus \mathcal{A}_\neg$, where $\mathcal{A}_\neg$ is the removed actions from $\mathcal{A}$.

Definition 4. *An* ***optimal solution*** *to a planning environment redesign problem* $\mathcal{R} = \langle \mathcal{E}, M_b \rangle$ *is a redesigned planning environment* $\mathcal{E}^* = \langle \mathcal{P}'_\mathcal{E} = \langle \mathcal{F}, \mathcal{A}', \mathcal{S}_\mathcal{I} \rangle, \mathcal{G} \rangle$ *that optimises the given redesign metric* M_b *while minimising* $|\mathcal{A}_\neg|$*.*

An optimal solution to an environment redesign problem $\mathcal{R}$ optimises the given metric M_b, breaking ties in favour of solutions requiring fewer changes to the environment.

Environment Redesign Metrics

Before proceeding to define the redesign metrics, we first provide some common notation and introduce the running example we use throughout the paper.

The *metrics* we use for environment redesign rely on reasoning about sets of plans for the possible goals $\mathcal{G}$ in $\mathcal{R}$, and we refer to these sets of plans as a *plan-library* $\mathbb{P}$ (following the terminology of *set of plans* defined in Section). We formally define a *plan-library* $\mathbb{P}$ in Definition 5.

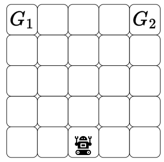


Figure 1: GRID environment where the agent located at cell $(2, 0)$ and has two possible goals: $G_1 = (0, 4)$; $G_2 = (4, 4)$.

Definition 5. *Given an environment $\mathcal{E}$ and a sub-optimality bound b, a **plan-library** of a planning environment with a bound b is defined as $\mathbb{P}(\mathcal{E}, b) = \bigcup_{G_i \in \mathcal{G}} \Pi(\langle \mathcal{P}_\mathcal{E}, \{G_i\}\rangle, b)$.*

Redesign metrics often relate to **plan prefixes** of a given size n, i.e., the first n actions of a plan π. We use $\vec{\pi}_n$ to refer to the first n actions of a plan π. Similarly, we use Π_n to denote all the plan prefixes of size n of a given set of plans Π. We abuse the notation and say that a plan prefix is inside a set of plans ($\vec{\pi}_n \in \Pi$) iff there exists a plan $\pi \in \Pi$ for which $\vec{\pi}_n$ is a plan prefix. We assume the actions $\mathcal{A}$ have a uniform cost equal to 1, but the metrics we define here are not limited to uniform cost. In order to simplify notation, we use x', x'' when referring to two different elements in a set, i.e., $x' \neq x''$. We also use $x \in (X', X'')$ to denote that $x \in X' \wedge x \in X''$.

As a *running example*, we use the GRID environment shown in Figure 1, where a robot can move in the four cardinal directions, and its possible goals consist of reaching the cells depicted by G_1 and G_2. We use (x, y) coordinates when referring to cells in the grid. When formalising the redesign metrics, we assume optimal agents' behaviour, so agents only follow optimal plans to achieve their goals ($b = 1.0$ when computing sets of plans).

Redesign Metrics

We now formally define a set of environment redesign metrics, in which two of them are well-known in the literature (Keren, Gal, and Karpas 2014; Mirsky et al. 2019), and the other ones are our *novel redesign metrics*.

Goal Transparency (GT). *Goal Transparency* (equivalent to GRD) aims at redesigning an environment such that an observer can infer agents' (or humans') true intended goal as soon as possible. This is useful in many applications such as *transparent planning* (MacNally et al. 2018), *human-robot collaboration* (Kulkarni et al. 2020), or *counterplanning* (Pozanco et al. 2018). *Goal Transparency* can be achieved by minimising the *worst case distinctiveness* (*wcd*) of an environment $\mathcal{E}$ (Keren, Gal, and Karpas 2014, 2019). We adapt the notation of Keren, Gal, and Karpas (2014, 2019) and formally define *wcd* as follows:

Definition 6. *Given a planning environment $\mathcal{E}$ in $\mathcal{R} = \langle \mathcal{E} = \langle \mathcal{P}_\mathcal{E}, \mathcal{G}\rangle, M_b\rangle$, let $\Pi' = \Pi(\langle \mathcal{P}_\mathcal{E}, G'\rangle, b)$, and $\Pi'' = \Pi(\langle \mathcal{P}_\mathcal{E}, G''\rangle, b)$ for $G', G'' \in \mathcal{G}$. The **worst case distinctiveness (wcd)** of a pair of goals G', G'' is the length of the longest plan prefix $\vec{\pi}$ that is present in Π' and Π'':*

$$wcd(G', G'') = \max\{n \mid \vec{\pi}_n \in (\Pi' \cap \Pi'')\}$$

*Thus, the **worst case distinctiveness of a planning environment** is denoted as wcd($\mathcal{E}$), and defined as:*

$$wcd(\mathcal{E}) = \max_{G', G'' \in \mathcal{G}} wcd(G', G'')$$

The *wcd* of the original environment shown in Figure 1 is 4, and the agent can execute 4 different actions (moving up 4 times) without revealing its actual goal. Figure 2a shows an optimal solution of the environment redesign problem where *Goal Transparency* is optimised ($M_{1.0}$), $wcd = 0$, and $|\mathcal{A}_\neg| = 1$. In this new environment, an observer will be able to recognise the agent's goal as soon as it executes the first action because the removal of the action to move from $(2, 0)$ to $(2, 1)$ forces the agent to move left or right, thus revealing its goal. *Goal Transparency* can also accommodate sub-optimal agents by adjusting the bound b, thus being useful for related tasks such as avoiding/preventing *goal obfuscation* (Bernardini, Fagnani, and Franco 2020) and *deception* (Masters and Sardiña 2017; Price et al. 2023).

Plan Transparency (PT). One could aim to redesign an environment such that an observer can infer the agents' (or humans') intended plans as soon as possible. We define such a task as *Plan Transparency* (equivalent to PRD). This is a stricter variant of *Goal Transparency*, so its applications are the same, and it can be achieved by minimising the *worst case plan distinctiveness* (*wcpd*) of an environment $\mathcal{E}$ (Mirsky et al. 2019). We adapt the notation in (Mirsky et al. 2019) and formally define *wcpd* as follows:

Definition 7. *Given $\mathcal{R} = \langle \mathcal{E} = \langle \mathcal{P}_\mathcal{E}, \mathcal{G}\rangle, M_b\rangle$, let $\pi', \pi'' \in \mathbb{P}(\mathcal{E}, b)$. The **worst case plan distinctiveness (wcpd)** of π', π'' is the length of the longest plan prefix $\vec{\pi}$ in π' and π'':*

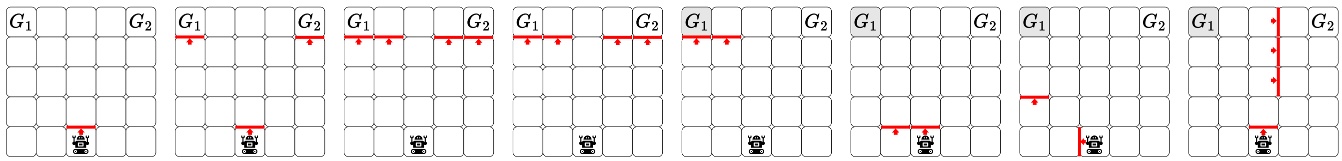
$$wcpd(\pi', \pi'') = \max\{n \mid \vec{\pi}_n \in (\{\pi'\}, \{\pi''\})\}$$

*Thus, the **worst case plan distinctiveness of a planning environment**, denoted as wcpd($\mathcal{E}$), is defined as:*

$$wcpd(\mathcal{E}) = \max_{\pi', \pi'' \in \mathbb{P}(\mathcal{E}, b)} wcpd(\pi', \pi'')$$

The $wcpd$ of the original environment shown in Figure 1 is 4: the agent can execute 4 actions (moving up 4 times) without revealing its plan. Figure 2b shows an optimal solution for this problem, where *Plan Transparency* is optimised ($M_{1.0}$ = PT), $wcpd = 0$, and $|\mathcal{A}_\neg| = 3$. In this new environment, an observer will be able to recognise the agent's plan as soon as it executes the first action, as now the agent has only one optimal plan available to achieve the goals.

Goal Privacy (GP). Sometimes, autonomous agents or humans plan and act in an environment in order to keep their goals *private*. To endow *Goal Privacy*, one could redesign an environment to allow agents (or humans) to keep their goals as private as possible during the execution of their plans. *Goal Privacy* can prevent goal recognition and be useful in adversarial settings (Kulkarni, Srivastava, and Kambhampati 2019) such as *goal obfuscation* (Bernardini, Fagnani, and Franco 2020). We introduce a novel metric called *worst case non-distinctiveness* (*wcnd*). Then, *Goal Privacy* optimization will be equivalent to maximising *wcnd*. We define *wcnd* as follows:



(a) *Goal Transparency.* (b) *Plan Transparency.* (c) *Goal Privacy.* (d) *Plan Privacy.* (e) *Min. Avg. Distance.* (f) *Max. Avg. Distance.* (g) *Min. Max. Distance.* (h) *Max. Min. Distance.*

Figure 2: Redesigned environments for different metrics by using our approach with a time limit of 15 minutes and a maximum number of removed actions $|\mathcal{A}_\neg| = 4$. Red arrows and lines indicate removed actions $\mathcal{A}_\neg$. Intended goals are depicted in grey.

Definition 8. *Given $\mathcal{R} = \langle \mathcal{E} = \langle \mathcal{P}_\mathcal{E}, \mathcal{G} \rangle, M_b \rangle$, let $\Pi' = \Pi(\langle \mathcal{P}_\mathcal{E}, G' \rangle, b)$, and $\Pi'' = \Pi(\langle \mathcal{P}_\mathcal{E}, G'' \rangle, b)$ for $G', G'' \in \mathcal{G}$. The* ***worst case non-distinctiveness (wcnd)*** *of a pair of goals G', G'' is the length of the shortest plan prefix $\vec{\pi}$ for which the symmetric difference of the plans sets Π'_n and Π''_n of size n is empty:*

$$wcnd(G', G'') = \min\{n \mid (\Pi'_n \triangle \Pi''_n) \neq \emptyset\}$$

Thus, the ***worst case non-distinctiveness of a planning environment*** *is denoted as wcnd($\mathcal{E}$), and defined as:*

$$wcnd(\mathcal{E}) = \min_{G', G'' \in \mathcal{G}} wcnd(G', G'')$$

The *wcnd* of the environment shown in Figure 1 is 0, and the agent can execute actions that might reveal its intended goal (moving left or right). Figure 2c shows an optimal solution of the environment redesign problem, where *Goal Privacy* is optimised ($M_{1.0}$ = GP), $wcnd = 4$, and $|\mathcal{A}_\neg| = 4$. As a result, the agent is forced to execute four actions without revealing its true intended goal. This metric and the resulting redesigned environment, are different (and more strict) than just maximising wcd. The original environment already has a maximum wcd of 4, so an optimal solution to maximise wcd would be an empty solution; i.e., do not apply any modification to the environment. However, this solution would allow the agent to execute actions that could reveal its intended goal earlier, so it would not be a solution to *Goal Privacy*.

Plan Privacy (PP). When facing specific situations that entail continuous monitoring, we may want to preserve our privacy by concealing what we are doing or aim to do. To do so, one could redesign an environment such that agents can keep their executed plans as private as possible. We define this task as *Plan Privacy*. This is a variant of *Goal Privacy*, so its applications are essentially the same, and the redesign metrics for *Goal and Plan Privacy* may have similar values depending on the problem. To achieve *Plan Privacy*, we define a novel metric called *worst case plan non-distinctiveness*, denoted as *wcpnd*, and it can be optimised by maximising *wcpnd*. We formally define *wcpnd* as follows:

Definition 9. *Given $\mathcal{R} = \langle \mathcal{E} = \langle \mathcal{P}_\mathcal{E}, \mathcal{G} \rangle, M_b \rangle$, let $\pi', \pi'' \in \mathbb{P}(\mathcal{E}, b)$. The* ***worst case plan non-distinctiveness (wcpnd)*** *of π', π'' is the length of the shortest plan prefix $\vec{\pi}$ for which $\pi' \neq \pi''$:*

$$wcpnd(\pi', \pi'') = \min\{n \mid \pi'_n \neq \pi''_n\}$$

Thus, the ***worst case plan non-distinctiveness of a planning environment model****, denoted as wcpnd($\mathcal{E}$), is defined as:*

$$wcpnd(\mathcal{E}) = \min_{\pi', \pi'' \in \mathbb{P}(\mathcal{E}, b)} wcpnd(\pi', \pi'')$$

The *wcpnd* of the original environment shown in Figure 1 is 0, in which the agent can act freely without being private about its executed plans. Figure 2d shows an optimal solution of the environment redesign problem where *Plan Privacy* is optimised ($M_{1.0}$ = PP), $wcpnd = 4$, and $|\mathcal{A}_\neg| = 4$. In this resulting environment, the agent can act by executing at least four actions in an optimal plan without revealing its intended plan. *Plan Privacy* can also accommodate sub-optimal agents' behaviour by adjusting the bound b, thus being useful for related planning applications where sub-optimal plans play an important role, such as *deceptive planning* (Masters and Sardiña 2017; Price et al. 2023).

Minimise Average Distance (MINAVGD). Certain situations require that agents (or humans) act in an environment to stay as close as possible to certain states. To accomplish this, one could redesign an environment such that an agent would be forced to "stay" *as close as possible* to a set of partial states whilst acting for achieving its true goal. We define this task as *Minimise Average Distance*, and its applications may include *anticipatory planning* (Burns et al. 2012) or *planning for opportunities* (Borrajo and Veloso 2021). More concretely, following the example in (Keren, Gal, and Karpas 2014), the airport operator might be interested in forcing passengers to pass through some shops on the way to their gates. It can also be useful in surveillance tasks, where one might want to constrain the surveillance agent's behaviour to pass through places where potential monitoring tasks might dynamically arrive. To practically endow this, we adapt the definition of planning *centroids* (Pozanco et al. 2019; Karpas 2022) to work over plans, rather than just single states. We define the average distance of an environment for a set of partial states and a goal state as *avgD*, as follows:

Definition 10. *Given $\mathcal{R} = \langle \mathcal{E} = \langle \mathcal{P}_\mathcal{E}, \mathcal{G} \rangle, M_b \rangle$, where $G_t \in \mathcal{G}$ is a true goal, and $\mathcal{G}_S = \mathcal{G} \setminus \{G_t\}$ is a set of partial states of interest to reason about. Let $\mathcal{S}_\Pi$ be all the states traversed by the plans in $\Pi = \langle \mathcal{P}_\mathcal{E}, G_t \rangle$. The* ***average distance of a planning environment*** *is denoted as avgD, and defined as:*

$$avgD(\mathcal{E}) = \frac{\sum_{s_i \in \mathcal{S}_\Pi, G_i \in \mathcal{G}_S} h^*(s_i, G_i)}{|\mathcal{S}_\Pi| \times |\mathcal{G}_S|}$$

The *avgD* of the original environment shown in Figure 1 is 5. Figure 2e shows a solution of the original environment redesign problem where the average distance is minimised ($M_{1.0}$ = *minAvgD*). In the resulting environment, two actions are removed ($|\mathcal{A}_\neg| = 2$), and the agent is forced to "stay" as close as possible to G_2 whilst following an optimal plan to achieve its intended goal G_1. This optimal plan involves moving north four steps, followed by 2 east steps, and traverses 7 states, yielding $avgD = \frac{6+5+4+3+2+3+4}{7} = 3.86$. Even if the example only shows one special goal to reason about, G_2, the metric works for any set of goals.

Maximise Average Distance (MAXAVGD). One could aim to redesign an environment such that agents would be forced to "stay" *as far as possible* from a set of potential *risks* whilst achieving their goals (Perny, Spanjaard, and Storme 2007). This can be useful in evacuation domains, such as in the event of a volcano eruption, where the goal is to move people to a safe place while staying as far as possible from a set of dangerous areas; or in financial planning, where the aim is to achieve the user's financial goal while staying far from financial risks such as high debt. In these cases, it is usually impossible to completely eliminate the risk (block the goal), so our assumption about all the goals being reachable (Def. 3) still holds in practice. To do so, we can *Maximise Average Distance* (*maxAvgD*) using Def. 10.

Figure 2f shows a solution for the environment redesign problem in Figure 1 when using *maxAvgD*, where average distance is maximised $M_{1.0}$ = *MaxAvgD*, $maxAvgD = \frac{6+7+8+7+6+5+4}{7} = 6.15$, and $|\mathcal{A}_\neg| = 2$. In this case, the agent is forced to stay as far as possible from G_2 while following an optimal plan to achieve G_1.

Minimise Maximum Distance (MINMAXD). *Minimise Maximum Distance* aims at redesigning an environment such that agents are "forced" to never stay too far from a set of partial states whilst achieving its true intended goal. It can be used in the same previous examples. We adapt the definition of planning *minimum covering states* (Pozanco et al. 2019) over plans, and define the maximum distance of an environment *maxD* as:

Definition 11. *Given $\mathcal{R} = \langle \mathcal{E} = \langle \mathcal{P}_\mathcal{E}, \mathcal{G} \rangle, M_b \rangle$, where $G_t \in \mathcal{G}$ is a true goal, and $\mathcal{G}_S = \mathcal{G} \setminus \{G_t\}$ is the set of partial states to reason about. Let $\mathcal{S}_\Pi$ be all the states traversed by the plans in $\Pi = \langle \mathcal{P}_\mathcal{E}, G_t \rangle$. The* ***maximum distance of a planning environment*** *is denoted as maxD, and defined as:*

$$maxD(\mathcal{E}) = \max_{s_i \in \mathcal{S}_\Pi, G_i \in \mathcal{G}_S} h^*(s_i, G_i) \quad (1)$$

The *maxD* of the environment shown in Figure 1 is 8, which is achieved when the agent visits the cell $(0, 0)$. Figure 2g shows a solution for this environment redesign problem where the maximum distance is minimised ($M_{1.0}$ = *MinMaxD*), then we have $maxD = 6$ and $|\mathcal{A}_\neg| = 2$. This metric is different from minimising average distance. While the solution in Figure 2e also has a *maxD* of 6, the solution in Figure 2g does not minimise *avgD*.

Maximise Minimum Distance (MAXMIND). One could redesign an environment such that agents are compelled to avoid getting too close to a set of partial states whilst achieving their true goal. Redesigning environments to optimise this metric can be useful in the same risk avoidance and evacuation domains we already mentioned. We define this task as *Maximise Minimum Distance* (*maxMinD*). We define the minimum distance of $\mathcal{E}$ as *minD*, as follows:

Definition 12. *Given $\mathcal{R} = \langle \mathcal{E} = \langle \mathcal{P}_\mathcal{E}, \mathcal{G} \rangle, M_b \rangle$, where $G_t \in \mathcal{G}$ is a true goal, and $\mathcal{G}_S = \mathcal{G} \setminus \{G_t\}$ is the set of partial states to reason about. Let $\mathcal{S}_\Pi$ be all the states traversed by the plans in $\Pi = \langle \mathcal{P}_\mathcal{E}, G_t \rangle$. The* ***minimum distance of a planning environment*** *is denoted as minD, and defined as:*

$$minD(\mathcal{E}) = \min_{s_i \in \mathcal{S}_\Pi, G_i \in \mathcal{G}_S} h^*(s_i, G_i) \quad (2)$$

The *minD* of the environment shown in Figure 1 is 2, which is achieved when the agent visits the cell $(2, 4)$. Figure 2h shows a solution to the environment redesign problem, where the minimum distance is maximised $M_{1.0}$ = *MaxMinD*, $minD = 6$, and $|\mathcal{A}_\neg| = 2$.

Environment Redesign via Search

We now present GER, a general environment redesign approach that is *metric-agnostic* and employs an anytime *Breadth-First Search* (BFS) (Russell and Norvig 2005, Section 3.3.1) algorithm that exploits recent research on *top-quality* planning to improve the search efficiency.

GER is described in Algorithm 1, and takes as input an environment redesign problem $\mathcal{R} = \langle \mathcal{E}, M_b \rangle$ and a stopping condition C. GER searches the space of environment modifications by iteratively generating and evaluating environments where an increasing number of actions is removed. GER returns the set of best solutions found $\mathcal{M}$ until C is triggered, i.e., the set of different environment modifications that optimises a redesign metric M_b, yielding a redesigned environment with metric value m^+.

Compute Plan-Library $\mathbb{P}$ (Line 1). GER first computes a plan-library $\mathbb{P}(\mathcal{E}, b)$ for the given environment redesign problem $\mathcal{R} = \langle \mathcal{E}, M_b \rangle$ by calling a TOPQUALITYPLANNER.

Compute Allowed Modifications (Line 2). After computing $\mathbb{P}(\mathcal{E}, b)$, GER then computes the set of allowed modifications for the given environment by using the GETALLOWEDMODIFICATIONS function, taking as input a plan-library $\mathbb{P}$, a set of actions $\mathcal{A}$, and a metric M_b to be optimised. Depending on the metric, this function can either return all the actions in $\mathcal{A}$, or only the subset of actions that appear in the plan-library $\mathbb{P}$, thus pruning the space of modifications. GER only reasons over the actions in the plan-library $\mathbb{P}$ for optimising GT, GP, PT or PP, as removing actions that do not appear in the plan-library does not affect these metrics. In addition, GER reasons over all the possible actions in $\mathcal{A}$ when optimising the distance-related metrics (*minAvgD*, *maxAvgD*, *minMaxD*, *maxMinD*), as removing actions that do not appear in the agent's optimal plans that achieve the true intended goal might affect and influence directly these metrics (see Figure 2h, where $\mathcal{A}_\neg$ includes actions that are not part of any optimal plan that achieves G_1).

Algorithm 1: GER: A General Environment Redesign Approach

Input: Redesign problem $\mathcal{R} = \langle \mathcal{E}, M_b \rangle$, C stopping condition.
Output: Set of solutions found $\mathcal{M}$, metric value found m^+.

1: $\mathbb{P}(\mathcal{E}, b) \leftarrow$ TOPQUALITYPLANNER$(\mathcal{E}, b)$
2: $amod \leftarrow$ GETALLOWEDMODIFICATIONS$(\mathbb{P}(\mathcal{E}, b), \mathcal{A}, M_b)$
3: $s_0 \leftarrow \emptyset$, OPEN $\leftarrow s_0$, $\mathcal{M} \leftarrow \{s_0\}$
4: $m_0, m^+ \leftarrow$ EVALUATE$(s, M, \mathcal{E}, b)$
5: **while** $\neg C$ **do**
6: $\quad s \leftarrow$ OPEN.DEQUEUE() {/* *State s with lowest* $|\mathcal{A}_\neg|$ */}
7: $\quad$ **for** a **in** $amod$ **do**
8: $\quad\quad s' \leftarrow s \cup a$
9: $\quad\quad$ **if** ISVALID(s') **then**
10: $\quad\quad\quad$ OPEN.QUEUE(s')
11: $\quad\quad\quad m' \leftarrow$ EVALUATE$(s', M, \mathcal{E}, b)$
12: $\quad\quad\quad$ **if** ISBETTER(m', m^+) **then**
13: $\quad\quad\quad\quad m^+ \leftarrow m'$, $\mathcal{M} \leftarrow \{s'\}$
14: $\quad\quad\quad$ **else if** $m' = m^+$ **and** $|s'| = |s''|,\ st.\ s'' \in\ \mathcal{M}$ **then**
15: $\quad\quad\quad\quad \mathcal{M} \leftarrow \mathcal{M} \cup \{s'\}$
16: **return** $\mathcal{M}, m^+$

Searching Process (Lines 3–16). With the computation of the plan-library $\mathbb{P}(\mathcal{E}, b)$ and the allowed modifications properly in place, GER initialises the search structures, and then conducts a BFS search until the stopping condition C is met (Line 5). Most existing algorithms only stop when the best possible value for a metric is achieved (Keren, Gal, and Karpas 2019; Mirsky et al. 2019). While defining this best possible value is easy for some metrics, i.e., $wcd = 0$ when optimising GT, this value is not easy to be properly defined for all metrics. Hence, we generalise the stopping conditions in the literature and assume C can represent any formula, such as a time limit or memory limit, a bound on the number of removed actions, or an improvement ratio of the metric with respect to its original value.

In each iteration, GER gets the best node from the open list OPEN according to its $g\text{-}value$, defined as the size of the removed actions set $|\mathcal{A}_\neg|$. Then, GER generates the successors of the current node s by adding removable actions in $amod$ to the current node's removed actions' set (Line 8). Before appending the new node s' to OPEN, GER checks if it is valid, verifying that all the goals in $\mathcal{G}$ are still achievable in the resulting environment after removing the actions in s'. If s' is a valid node, GER computes the value of the metric M_b for that node, m', using the EVALUATE function. This function assesses the quality of the environment obtained after removing the actions in s', using for example any of the metrics proposed in Definitions 6–12. If m' ISBETTER than the best metric value m^+ found so far (lower when minimising, higher when maximising), then this value is replaced, and the set of environment modifications $\mathcal{M}$ is updated (Lines 12–15). If m' is equal to m^+ and node s' has the same number of modifications (same $g\text{-}value$) as those nodes in $\mathcal{M}$, then node s' is included in the set of environment modifications $\mathcal{M}$ (Line 15). Finally, GER terminates when the condition C is met (Line 5), returning the best solutions found (environment modifications $\mathcal{M}$), and the best value m^+ for the redesign metric M in these solutions. Theoretical properties of GER can be found in the extended version of this paper (Pozanco, Pereira, and Borrajo 2024).

Experiments and Evaluation

We now present the experiments carried out to evaluate GER. The aim of the evaluation is twofold: (1) compare GER against state-of-the-art approaches for GRD (Keren, Gal, and Karpas 2014) when optimising *wcd*; and (2) show GER's performance when optimising the new redesign metrics.

Benchmarks and Setup. We have created a benchmark set that contains 300 planning environment problems equally split across the five well-known domains: BLOCKS words, DEPOTS, GRID, IPC-GRID, and LOGISTICS. The number of possible goals varies in size, having on average 4 possible goals over the different benchmarks. For the metrics where this is relevant, the true goal G_t is selected as the first goal in $\mathcal{G}$. The environments are encoded in PDDL (Planning Domain Definition Language) (McDermott et al. 1998). We generate 8 redesign problems for each environment by varying the metric M_b that should be optimised, using the metrics defined in Definitions 6 to 12. This gives us $300 \times 8 = 2400$ planning environment redesign problems.

GER uses SYM-K (von Tschammer, Mattmüller, and Speck 2022) to compute the plan-library. We run SYM-K with a bound of 1.0, i.e., aiming for optimal plans, although all our metrics support arbitrary sub-optimality bounds. We also set a limit of $1,000$ plans to prevent disk overflows and avoid GER spending all the time computing the plan-library in redesign problems with a large number of optimal plans. For the subset of 300 environment redesign problems, where the aim is minimising *wcd*, we compare GER against the most efficient GRD approach (*latest-split*) of Keren, Gal, and Karpas (2014), denoted as GRD-*LS*. We have run all experiments using 4vCPU AMD EPYC 7R13 Processor 2.95GHz with 32GB of RAM, and run GER with $C = \{$*time limit* = 900s or *memory limit* = 4GB$\}$. We used the same stopping condition C for both GER and GRD-*LS*. Benchmarks and GER's code are available on GitHub[1]. Further results and analysis can be found in the extended version of this paper (Pozanco, Pereira, and Borrajo 2024).

Results. Table 1 shows our results, using the following metrics: T, the time (seconds) to find the best solution; m_0, the metric value of the original environment; and m^+, the metric value of the environment returned as a solution. We only report results for the subset of problems for which the given metric could be improved within the time and memory limits. As for GT, we only report results for problems in which both GER and GRD-*LS* can improve the given metric.

As we can see in the GT columns (first inner table), GER yields the same results (same redesigned environments) as GRD-*LS* (Keren, Gal, and Karpas 2014) but two orders of magnitude faster. GRD-*LS* needs 119.9 seconds on average to find the best solution in 8 out of 60 IPC-GRID problems, whereas GER only needs 1.1 seconds on average. This performance gap can be explained by two factors. First, GER uses SYM-K to compute a plan-library before searching in the space of the environment's modifications. By only removing the actions appearing in this library, GER needs to explore much fewer nodes than GRD-*LS*. Second, GRD-*LS*

[1] https://github.com/ramonpereira/general-environment-redesign

	GT ⇓						PT⇓			GP⇑			PP⇑		
	GER			GRD-*LS*			GER			GER			GER		
# *domain*	T	m_0	m^*	T	m_0	m^*	T	m_0	m^*	T	m_0	m^*	T	m_0	m^*
BLOCKS	1.0	5.2	3.7	63.7	5.2	3.7	1.0	5.3	3.8	1.0	2.4	4.4	1.0	2.4	4.4
DEPOTS	1.0	5.0	4.0	69.0	5.0	4.0	1.2	6.5	5.2	1.1	0.0	4.0	-	-	-
GRID	8.1	4.2	1.8	345.0	4.2	2.2	60.1	4.0	1.8	19.1	0.0	1.4	1.4	0.0	1.4
IPC-GRID	1.1	11.1	8.5	119.9	11.1	8.5	1.1	11.1	7.7	0.9	2.0	3.0	0.9	2.0	4.0
LOGISTICS	-	-	-	-	-	-	-	-	-	150.2	0.0	10.0	-	-	-

	MINAVGD⇓			MAXAVGD⇑			MINMAXD⇓			MAXMIND⇑		
	GER			GER			GER			GER		
# *domain*	T	m_0	m^*	T	m_0	m^*	T	m_0	m^*	T	m_0	m^*
BLOCKS	114.7	8.2	7.9	84.7	8.4	8.9	91.2	12.7	11.2	70.1	3.2	4.8
DEPOTS	52.8	7.0	6.6	104.3	7.1	7.5	48.3	12.1	9.8	54.0	2.6	4.0
GRID	266.0	4.3	4.0	383.8	4.3	4.9	152.0	7.9	6.7	232.3	0.7	2.0
IPC-GRID	167.4	13.5	13.2	173.8	11.1	11.6	29.8	19.3	17.7	167.7	3.1	4.2
LOGISTICS	409.8	10.1	9.6	410.6	9.9	10.5	-	-	-	314.0	3.3	5.0

Table 1: Each cell represents *avg* values for the redesign metrics. Cells with "-" mean that the metric could not be improved for the problems in the domain for the time limit of 900 seconds. ⇓ represents reducing m_0, whereas ⇑ represents increasing m_0.

needs to generate and solve new planning problems for each node in order to compute the *wcd* of the new environment, resulting in a huge computational overhead. On the contrary, GER can compute the *wcd* very efficiently by just analysing the common prefixes of the plans in the plan-library.

GER's execution time increases when redesigning environments to optimise the distance-based metrics. Two factors influence this: (1) the space of action's removal is larger, as GER is not constrained to only remove actions in the plan-library; and (2) evaluating the metric of each search state is more costly than for the other metrics. For GT, PT, GP, and PP, GER only reasons over plans and their common prefixes, while for the distance-based metrics GER computes the optimal costs from each state traversed by optimal plans that achieve G_t and the other states in $\mathcal{G}_S = \mathcal{G} \setminus \{G_t\}$.

Related Work

Most approaches to environment redesign assume the observer's objective is to modify the environment to facilitate recognizing goals and plans (Keren, Gal, and Karpas 2014; Son et al. 2016; Mirsky et al. 2019). Later works in GRD frame and solve this task under different observability settings (Keren, Gal, and Karpas 2015, 2016a,b), environment assumptions (Wayllace et al. 2016, 2020; Wayllace and Yeoh 2022), or observer's capabilities (Shvo and McIlraith 2020; Gall, Ruml, and Keren 2021). Unlike these works, we assume the interested party might want to modify the environment for tasks different than recognising goals and plans.

On the algorithmic side, most works use search algorithms to explore the space of actions' removal (Keren, Gal, and Karpas 2021). While GER searches in the same space using similar algorithms, it differs from these works as follows. First, GER presents a good compromise between approaches that do not use plan-libraries (Keren, Gal, and Karpas 2019) and those that need pre-defined plan-libraries (Mirsky et al. 2019). GER exploits recent advances in top-quality planning to efficiently compute plan-libraries for pruning the space of modifications. Second, GER is *metric-agnostic*. Previous approaches are *metric-dependent*, devising pruning techniques and stopping conditions tailored to specific metrics, whereas GER is more general and can accommodate a wide variety of metrics. Third, GER is able to return all the best solutions found until a stopping condition is met. This is usually a desirable feature in applications with *humans-in-the-loop* (Boddy et al. 2005; Sohrabi et al. 2018), as humans prefer to have diverse solutions to choose from.

Conclusions

In this paper, we extended the definition of environment design from previous work, and we introduced a more general task for *Planning Environment Redesign*. We defined a new set of environment redesign metrics that endows and facilitates not only the recognition of goals and plans, but also other tasks, such as deception, risk avoidance, or planning for opportunities. We showed that our general environment redesign approach GER is *metric-agnostic*, and can optimise a wide variety of redesign metrics. Our experiments show that GER is efficient to optimise different metrics, and it outperforms (being orders of magnitude faster) the most efficient GRD approach of Keren, Gal, and Karpas (2014).

We intend to expand this work in two directions: improving GER's performance and making the problem definition even more general. Regarding performance, we aim to develop heuristics to improve the search process. Namely, one could prioritise removing actions belonging to a higher number of plans in the plan-library when optimising GT. We also aim to study how to balance the time allocated to compute the plan-library and the time to search in the space of modifications. As for the problem formulation, we envisage allowing other modifications such as removing or adding objects or predicates from the initial state. Finally, we aim to investigate how to jointly optimize sets of these metrics.

Acknowledgements

This paper was prepared for informational purposes in part by the Artificial Intelligence Research group of J.P. Morgan Chase & Co and its affiliates (J.P. Morgan) and is not a product of the Research Department of J.P. Morgan. J.P. Morgan makes no representation and warranty whatsoever and disclaims all liability, for the completeness, accuracy, or reliability of the information contained herein. This document is not intended as investment research or investment advice, or a recommendation, offer, or solicitation for the purchase or sale of any security, financial instrument, financial product, or service, or to be used in any way for evaluating the merits of participating in any transaction. It shall not constitute a solicitation under any jurisdiction or to any person if such solicitation under such jurisdiction or to such person would be unlawful.

References

Bernardini, S.; Fagnani, F.; and Franco, S. 2020. An Optimization Approach to Robust Goal Obfuscation. In *KR*.

Boddy, M. S.; Gohde, J.; Haigh, T.; and Harp, S. A. 2005. Course of Action Generation for Cyber Security Using Classical Planning. In *ICAPS*.

Borrajo, D.; and Veloso, M. 2021. Computing Opportunities to Augment Plans for Novel Replanning during Execution. In *ICAPS*.

Burns, E.; Benton, J.; Ruml, W.; Yoon, S. W.; and Do, M. B. 2012. Anticipatory On-Line Planning. In *ICAPS*.

Chakraborti, T.; Kulkarni, A.; Sreedharan, S.; Smith, D. E.; and Kambhampati, S. 2019. Explicability? Legibility? Predictability? Transparency? Privacy? Security? The Emerging Landscape of Interpretable agent Behavior. In *ICAPS*.

Gall, K. C.; Ruml, W.; and Keren, S. 2021. Active Goal Recognition Design. In *IJCAI*.

Geffner, H.; and Bonet, B. 2013. *A Concise Introduction to Models and Methods for Automated Planning*. Morgan & Claypool Publishers.

Karpas, E. 2022. A Compilation Based Approach to Finding Centroids and Minimum Covering States in Planning. In *ICAPS*.

Katz, M.; Sohrabi, S.; and Udrea, O. 2020. Top-quality planning: Finding practically useful sets of best plans. In *ICAPS*.

Keren, S.; Gal, A.; and Karpas, E. 2014. Goal Recognition Design. In *ICAPS*.

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

Keren, S.; Gal, A.; and Karpas, E. 2016a. Goal Recognition Design with Non-Observable Actions. In *AAAI*.

Keren, S.; Gal, A.; and Karpas, E. 2016b. Privacy Preserving Plans in Partially Observable Environments. In *IJCAI*.

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

Keren, S.; Gal, A.; and Karpas, E. 2021. Goal Recognition Design - Survey. In *IJCAI*.

Kulkarni, A.; Sreedharan, S.; Keren, S.; Chakraborti, T.; Smith, D. E.; and Kambhampati, S. 2020. Designing environments conducive to interpretable robot behavior. In *IROS*.

Kulkarni, A.; Srivastava, S.; and Kambhampati, S. 2019. A unified framework for planning in adversarial and cooperative environments. In *AAAI*.

MacNally, A. M.; Lipovetzky, N.; Ramírez, M.; and Pearce, A. R. 2018. Action Selection for Transparent Planning. In *AAMAS*.

Masters, P.; and Sardiña, S. 2017. Deceptive Path-Planning. In *IJCAI*.

McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. 1998. PDDL – The Planning Domain Definition Language. In *AIPS*.

Mirsky, R.; Gal, K.; Stern, R.; and Kalech, M. 2019. Goal and Plan Recognition Design for Plan Libraries. *ACM Transactions on Intelligent Systems and Technology*, 10(2): 14:1–14:23.

Perny, P.; Spanjaard, O.; and Storme, L. S. 2007. State space search for risk-averse agents. In *IJCAI*.

Pozanco, A.; E-Martín, Y.; Fernández, S.; and Borrajo, D. 2018. Counterplanning using Goal Recognition and Landmarks. In *IJCAI*.

Pozanco, A.; E-Martín, Y.; Fernández, S.; and Borrajo, D. 2019. Finding Centroids and Minimum Covering States in Planning. In *ICAPS*.

Pozanco, A.; Pereira, R. F.; and Borrajo, D. 2024. Generalising Planning Environment Redesign. arXiv:2402.07799.

Price, A.; Pereira, R. F.; Masters, P.; and Vered, M. 2023. Domain-Independent Deceptive Planning. In *AAMAS*.

Ramírez, M.; and Geffner, H. 2009. Plan Recognition as Planning. In *IJCAI*.

Russell, S.; and Norvig, P. 2005. AI a Modern Approach. *Learning*, 2(3): 4.

Shvo, M.; and McIlraith, S. A. 2020. Active Goal Recognition. In *AAAI*.

Sohrabi, S.; Riabov, A.; Katz, M.; and Udrea, O. 2018. An AI planning solution to scenario generation for enterprise risk management. In *AAAI*.

Son, T. C.; Sabuncu, O.; Schulz-Hanke, C.; Schaub, T.; and Yeoh, W. 2016. Solving Goal Recognition Design Using ASP. In *AAAI*.

von Tschammer, J.; Mattmüller, R.; and Speck, D. 2022. Loopless top-k planning. In *ICAPS*.

Wayllace, C.; Hou, P.; Yeoh, W.; and Son, T. C. 2016. Goal Recognition Design with Stochastic Agent Action Outcomes. In *IJCAI*.

Wayllace, C.; Keren, S.; Gal, A.; Karpas, E.; Yeoh, W.; and Zilberstein, S. 2020. Accounting for Observer's Partial Observability in Stochastic Goal Recognition Design. In *ECAI*.

Wayllace, C.; and Yeoh, W. 2022. Stochastic Goal Recognition Design Problems with Suboptimal Agents. In *AAAI*.

Zhang, H.; Chen, Y.; and Parkes, D. C. 2009. A General Approach to Environment Design with One Agent. In *IJCAI*.