DISSERTATION

INVESTIGATING THE APPLICATIONS OF MODEL REASONING FOR HUMAN-AWARE AI SYSTEMS

Submitted by

Turgay Caglar

Department of Computer Science

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Summer 2025

Doctoral Committee:

Advisor: Sarath Sreedharan

Nathaniel Blanchard
Nikhil Krishnaswamy
Anne Cleary

ABSTRACT

INVESTIGATING THE APPLICATIONS OF MODEL REASONING FOR HUMAN-AWARE AI SYSTEMS

This dissertation investigates how intelligent agents can reason over their models to better support, explain, and adapt to human users. Traditional AI planning assumes that the underlying model—describing the agent's actions, goals, and environment—is fixed and complete. However, in real-world deployments, these models often diverge from users' expectations, leading to confusion, mistrust, or failure. To address this, I propose a shift from reasoning within a model to reasoning about the model itself, using a framework called model-space search. Through four interconnected works, I demonstrate how model reasoning enables agents to operate more effectively in human-aware settings. First, I show how agents can proactively support users by detecting likely failure due to model misalignment and suggesting minimal corrections. Second, I extend explanation frameworks to include the intentions of system designers, revealing hidden influences on agent behavior. Third, I introduce Actionable Reconciliation Explanations, which combine model reconciliation and excuse generation to help users both understand and influence agent behavior. Finally, I explore how Large Language Models can enhance model-space search by guiding it toward more plausible and interpretable updates. Together, these contributions establish model reasoning as a foundation for building AI systems that are not only autonomous but also transparent, adaptable, and aligned with the people they serve.

ACKNOWLEDGEMENTS

I am deeply grateful to my advisor Sarath Sreedharan for his continuous support throughout my PhD journey. There is so much I could say about Sarath, and I truly believe his guidance deserves more than just a few lines. I am thankful for the opportunity to join his lab, for the knowledge he has generously shared, and for always being there when I needed direction. Before meeting Sarath, I never thought of myself as particularly lucky, but having the honor to pursue my PhD under his mentorship changed that. His commitment to understanding complex ideas, his kindness, and his dedication to his students have shaped not only how I think, but also how I want to treat others. He has set a standard of excellence and compassion that I will always strive to follow.

I am also thankful to my committee members. I am especially grateful to Nathaniel Blanchard for offering opportunities that pushed me out of my comfort zone and helped me grow in many ways. Anne Cleary's support during the Deja Vu project helped expand my perspective on research and collaboration. Although I did not work closely with Nikhil Krishnaswamy, I appreciated his thoughtful feedback and the fresh insights he contributed to my dissertation.

Throughout this journey, I have had the privilege of collaborating with incredible researchers. Working with Tathagata Chakraborty on my first project with Sarath was a great experience. His openness, clarity, and thoughtful feedback helped shape how I view collaboration. Although our time working together was brief, his influence stayed with me. Much of the literature I explored during my dissertation is based on the foundational work of both Tathagata and Sarath. I am also very grateful for the opportunity to work with Mor Vered. Her ability to make social science topics accessible and relevant to our field was inspiring. Her clear thinking and structured approach to designing studies and conveying ideas helped me grow as a researcher. I would like to thank Zahra Zahedi for her valuable feedback and support during our user studies. Her input made a big difference in how we approached our work. I am also thankful for the chance to collaborate with Michael Katz, who has been supportive both to me and to the AI planning community. It was an honor to work with him and learn from his experience.

I also want to thank my lab mates and friends who have been part of this journey. Huma Jamil, Yongxin (Cin) Liu, Changsoo Jung, Sonu Dileep, Sanhith Rangaraju, Chirag Kandoi, and Caspian Siebert were always there for me, and I will always remember our gatherings and the moments we shared. I am especially thankful to Kelsey Sikes and Malek Mechergui, with whom I spent the most time in the lab. Kelsey's talent for remembering everything and her openness to talk about any topic made our conversations fun and engaging. Malek's kindness and positivity always helped lift my spirits. I feel lucky to have had such supportive people around me throughout this experience.

DEDICATION

To my mother, Fatma Çağlar

All thanks to you, all for you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

The field of Artificial Intelligence (AI) continues to evolve, driving innovation in domains where intelligent systems must autonomously solve complex problems. One of the foundational capabilities enabling such behavior is AI planning, which focuses on generating sequences of actions that lead agents from an initial state to a desired goal. Traditional approaches in this area typically assume that the underlying planning model is complete, correct, and static. Within such assumptions, the core task is to find an optimal or feasible plan that satisfies the goal condition under the given model.

However, these assumptions rarely hold in real-world applications. In practice, planning models are often incomplete, outdated, or misaligned with human expectations. The environment may change, users may interpret the task differently, and designers may embed objectives into the system that are not fully transparent. These discrepancies can lead to plans that appear inexplicable, unsafe, or unintuitive to human collaborators, ultimately reducing trust and system effectiveness.

To build AI systems that are more adaptable, transparent, and user-aligned, we must move beyond simply generating plans within a fixed model. We must instead empower agents to reason over the model itself to consider what assumptions are embedded in the model, how those assumptions might differ from a user's expectations, and how the model might be revised or interpreted differently to support collaboration. This broader perspective gives rise to the framework of model-space search, where the objective is not just to find a plan, but to identify or adapt the model in ways that improve performance, understanding, or alignment.

Model-space search reframes planning tasks as problems situated in a space of possible models. Instead of asking "What is the best plan for this model?", we can ask "What model would make this plan valid?", "What minimal changes to the model would help a user understand the agent's actions?", or "What assumptions need to be revised for a failed plan to succeed?" This shift in

focus enables a wide range of functionalities that are essential for real-world AI systems, including explanation, intervention, adaptation, and model correction.

This work is grounded in the broader research area of Human-Aware AI Planning, which emphasizes the importance of designing planning agents that explicitly account for the mental models, capabilities, and goals of human users. Human-Aware AI Planning extends classical planning by requiring agents to consider not only what is optimal in their own model but also how that behavior will be interpreted or assisted by humans. It highlights the role of model divergence as a source of misunderstanding and presents new planning objectives centered around explicability, transparency, and assistance.

My dissertation builds upon the Human-Aware AI Planning framework by focusing specifically on reasoning over models—both the agent's and the human's—as a way to operationalize these objectives. I argue that model reasoning via model-space search provides a powerful and generalizable tool for achieving the goals of human-aware AI systems. Rather than treating models as fixed inputs to the planning process, I treat them as variables to be explored, updated, and aligned with stakeholders' expectations.

This dissertation presents four works that illustrate how model reasoning can improve collaboration, understanding, and adaptability in AI systems. These chapters are connected by a common methodological foundation—model-space search—but they explore different applications, challenges, and extensions of the idea.

These contributions are structured as four chapters (Chapters 3–6), each focusing on a different aspect of model-space reasoning. Collectively, they contribute to a broader vision in which reasoning over models becomes a core capability of intelligent, human-aware agents.

To support these contributions, Chapter 2 provides the necessary theoretical and conceptual background. It introduces formal notions of planning, model-space search, and the core ideas from Human-Aware AI Planning that underpin the work in later chapters.

Chapter 3 initiates the exploration by addressing the problem of knowledge asymmetry—situations where users have flawed or incomplete models of the task. I propose a method that combines prob-

abilistic goal recognition with model-space search to detect when users are likely to fail and to identify minimal model updates for recovery. This chapter introduces the foundational need for agents to reason about others' models to provide timely and context-sensitive support.

Building on the need for explanation in asymmetric settings, this chapter considers cases where an agent's behavior is influenced by hidden design goals. While Chapter 3 focused on helping users based on their misunderstandings, Chapter 4 explores how design-driven system behavior can appear opaque, even when users understand the agent's model. I extend model reasoning to include designer-aware explanations, offering insights into how agents and environments were constructed to produce particular behaviors.

Having shown how agents can explain themselves and their designer goals, this chapter shifts toward interactive explanations—enabling users not just to understand the system but to influence its behavior. I introduce *Actionable Reconciliation Explanations,* which blend model reconciliation and excuse generation into a single explanation that shows both why the agent acted as it did and how the model could be changed to support a user-preferred alternative. This work marks a transition from explanatory systems to collaborative, modifiable agent plans.

The preceding chapters demonstrate the expressiveness of model-space search, but also expose some limitations. While model-space search provides a principled framework, it often suffers from computational bottlenecks and lacks preference sensitivity among logically equivalent model edits. In Chapter 6, we investigate how Large Language Models (LLMs) can be leveraged to improve model-space reasoning. We explore LLMs as standalone model reasoners and also as complementary components that guide combinatorial search toward more likely or natural model edits. This work marks an important step toward scaling model reasoning and incorporating broader commonsense signals into planning tasks.

Contribution This dissertation advances the field of Human-Aware AI by positioning model reasoning—reasoning about, rather than only within, planning models—as a foundational capability for intelligent systems. By treating models as entities that can be interrogated, explained, and

modified, this work enables a shift from purely action-generating systems to truly collaborative agents that can engage with users' mental models, designers' intentions, and dynamic task contexts.

- This dissertation shows how model reasoning enables AI agents to detect, explain, and resolve misalignments between their internal models and the expectations of users or designers, ultimately improving transparency, collaboration, and trust.
- It introduces new forms of explanation—including designer-aware and actionable explanations—that go beyond traditional plan justification to support deeper user understanding and meaningful interaction.
- It demonstrates how model-space search provides a unifying framework for enabling proactive assistance, rich explanations, and adaptive behavior in human-aware AI systems.
- It enhances the scalability and human-likeness of model reasoning by integrating large language models, making model-space search more efficient and better aligned with real-world expectations.

1.1 Publications Covered in the Dissertation

The core of this dissertation includes four peer-reviewed publications: two papers published in AAMAS, one published in HRI, and one published in AAAI.
The following chapters are based on these publications and reflect the core contributions of the dissertation:

- Chapter 3 is based on the paper presented at AAMAS-24 [1].
- Chapter 4 is based on the paper presented at AAMAS-25 [2].
- Chapter 5 is based on the paper presented at HRI-25 [3].
- Chapter 6 is based on the paper published in AAAI-24 [4].

Chapter 2

BACKGROUND

In our chapters, we will focus on goal-directed deterministic planning problems, i.e., classical planning or STRIPS [5] style planning problems. We can represent such planning problems or equivalently models using tuples of the form $\mathcal{M} = \langle F, A, I, G, C \rangle$, where $\mathcal{M}$ represents a planning model. In this tuple, F is a set of propositional fluents used to define the set of possible unique states within the model. A is the set of actions that can be executed. Each action $a \in A$ is represented as a tuple $\langle pre(a), add(a), del(a) \rangle$, where $pre(a) \subseteq F$, specifies the preconditions for the action. An action a is available in a state s if $pre(a) \subseteq s$. The sets $add(a) \subseteq F$ and $del(a) \subseteq F$ represent the add and delete effects of the action, respectively. We will use the function $\delta_M : 2^F \times A \rightarrow 2^F$ to represent the transition function that captures resulting of executing an action. As such, the result of executing an action a in a given state $s \subseteq F$ is represented as $\delta_M(s, a) = (s \cup add(a)) \setminus del(a)$, where $pre(a) \subseteq s$. $I \subseteq F$ defines the initial state, and $G \subseteq F$ specifies the goal description. Finally, C is a cost function that maps actions to their costs, which are real values.

The solution to this problem is a plan (sequence of actions), $\pi = \langle a_1, a_2, ..., a_n \rangle$, such that $\delta_M(I, \pi) \supseteq G$ (where $\delta_M(I, \pi) = \delta_M((\delta_M(I, a_1), a_2), ...a_n)$). The cost of the plan π is $C(\pi) = \sum_{a_i \in \pi} C(a_i)$. We call a plan π optimal if no other plan can satisfy the goal description G with less cost. We represent such optimal plans in model M as π^*, and similarly, the cost for the optimal plan will be C^*_M. The original work on excuses focused on unsolvable planning problems, i.e., no action sequence whose execution leads to the goal exists. To capture such cases uniformly, we introduce a new action a_{-1}, which has empty preconditions, whose add effects contain the goal description, and has an infinite cost. Thus, a problem is unsolvable if its optimal plan is $\pi_{-1} = \langle a_{-1} \rangle$.

We are interested in setting where the robot's model of the task $\mathcal{M}^R = \langle F, A^R, I^R, G^R, C^R \rangle$ may be different from the human's beliefs about the task $\mathcal{M}^H = \langle F, A^H, I^H, G^H, C^H \rangle$. Keeping with the foundational settings in human-aware planning [6], we will consider a case where the robot is the primary actor, and the human uses their beliefs to make sense of the robot's actions. For the

sake of simplifying the technical discourse, we will assume the two models share the fluent space and action labels (but not necessarily the action definitions). One of the basic problems that could arise here is when the plan followed by the robot (π^R) is not the same as the one expected by the human (π^H). To simplify the discussion, we will assume that the expectation mismatch arises from the fact that the human might not think the robot plan is optimal or even valid (per $\mathcal{M}^H$) and the robot might have similar views about the human plan[1]. Both model reconciliation explanations [7] and excuses [8] provide two orthogonal ways of addressing this same problem.

Both these methods use model space search to update either of the two models. In particular, this model space is defined over a space of model parameters $\mathcal{F}^{(F,A)}$, which is defined over the fluents and action labels shared between the two models, and this model parameter set is given as where

$$\mathcal{F}^{(F,A)} = \{\textit{init-has-}f \mid f \in F\} \cup \{\textit{goal-has-}f \mid f \in F\} \cup \bigcup_{a \in A} \{a\textit{-has-positive-prec-}f a\textit{-has-add-fluent-}f, a\textit{-has-del-fluent-}f \mid f \in F\}.$$

[1]It is worth noting that most of the following discussion still holds if the human is not performing optimal planning

We will now use a parameterization function ($\Gamma(\cdot)$) that will map each model to a subset of the model parameters (we will follow the conventions set by [9])

$$\tau_I = \{\textit{init-has-f} \mid f \in I\}$$

$$\tau_G = \{\textit{goal-has-g} \mid g \in G\}$$

$$\tau_{pre(a)} = \{a\textit{-has-prec-fluent-}f \mid f \in pre(a)\}$$

$$\tau_{add(a)} = \{a\textit{-has-add-fluent-}f \mid f \in add(a)\}$$

$$\tau_{del(a)} = \{a\textit{-has-del-fluent-}f \mid f \in del(a)\}$$

$$\tau_a = \tau_{pre(a)} \cup \tau_{add(a)} \cup \tau_{del(a)}$$

$$\tau_A = \bigcup_{a \in A^{\mathcal{M}}} \tau_a$$

$$\Gamma(\mathcal{M}) = \tau_I \cup \tau_G \cup \tau_A$$

Similarly, we will use the function Γ^{-1} to map a set of model parameters into a model, and we will represent the set of all possible models that can be represented using the model parameters as $\mathbb{M}^{(F,A)}$. Now, under explanation model reconciliation, the goal is to update the human model ($\mathcal{M}^H$), such that the robot plan (π^R) will be optimal in the updated model. Here, the changes made to the human model align with the robot model; that is, the human is provided with true information about the robot model (that may not have been known previously). We will represent this process of updating the human model as: $\widehat{\mathcal{M}}^H = \mathcal{M}^H + \mathcal{E}$

Where $\widehat{\mathcal{M}}^H$ is the updated human model, '+' is the update operator, and $\mathcal{E}$ is the model information (i.e., the explanation). Note that the use of the '+' operator is one of convenience since the update may provide the human with information about the robot model they previously didn't know (as such, add new parameters to $\Gamma(\mathcal{M}^H)$), and possibly correct misconception they may have about the robot model (as such, remove parameters from $\Gamma(\mathcal{M}^H)$). We will use the cost function $\mathcal{C}^{\mathcal{E}}$ to return the non-negative cost of providing an explanation (i.e., a piece of model information) and will refer to an explanation set with the lowest cost as a minimally complete explanation or *MCE*.

We believe that that this background introduces the key concepts required for the upcoming chapters. Although conditional effects are relevant, we introduce them later in Chapter 3 as they are not essential for the foundational understanding provided here and are specific to the context discussed there.

Chapter 3

HELP! PROVIDING PROACTIVE SUPPORT IN THE PRESENCE OF KNOWLEDGE ASYMMETRYS

In this chapter, I explore how model reasoning can support users who operate under incorrect or incomplete assumptions about a task. In such cases, the plans they generate may be invalid in the actual environment, leading to unintended or even dangerous outcomes. This work focuses on enabling agents to detect when these model misalignments may result in user failure and to intervene early with minimal and targeted support. The core idea is to estimate the likelihood that a user's plan—though not directly observable—will lead to failure due to differences in task models. To do this, I frame the problem as a form of probabilistic goal recognition, allowing the agent to infer risk from observed actions. Once risk is detected, I apply model-space search to identify the smallest set of model updates that would correct the user's misconceptions and enable a valid plan. Finally, I introduce a decision-theoretic approach to guide early intervention, ensuring the agent acts before failure becomes irreversible. This chapter highlights how reasoning over user models can proactively support success without assuming full knowledge of user plans or goals.

3.1 Introduction

There is a long history within AI for developing proactive personal assistants [10–17]. These are automated agents that are meant to keep track of the activities of a user and their eventual goals and offer help whenever the agent determines that the user may benefit from it. While many previous works have looked at such support problems, most of the works that propose formal support models seem to focus on a few specific settings. In particular, there is a lot of focus on what might be considered serendipitous support,' where the user tries to achieve their goal, and the agent tries to

reduce the burden placed on the user (cf. [18–22]). Works have also looked at supporting users with cognitive disabilities [23–26].

However, a direction of work that has gotten relatively less attention is the use of disembodied assistants in the presence of knowledge assymmetry i.e., support users when their estimate about the task may have changed or differs from the agent's true estimate. As such, the plans users devise may not be valid. The potential use cases here could range from safety-critical, imagine a system alerting a rescue worker in a disaster scenario about a collapsed wall blocking their path, to the quotidian, a system that informs the user about heavy traffic in their normal route to work. Also, the setting simplifies the kinds of intervention required from the agent's end to merely informing the user about task information they might not have been aware of previously.

The primary challenge with these works is identifying the degree to which (if at all) the difference in task knowledge impacts the user's ability to generate valid plans. If they do, the agent should be able to recognize what information about the underlying model should be provided to the user so they can choose a new valid plan. As with other works in this direction, we will assume that the user's actual plan is not known upfront to the assistive agent. Additionally, the difference in the user's estimate of the task and the agent's might be significant. Depending on the plan the user selects, only a subset of differences between the agent's estimate and the user's own estimate might be relevant, where relevancy is determined by whether it impacts the plan's validity. As such, we will start by proposing a way to estimate the probability that a user may follow an invalid plan in the agent's task model. We will introduce a novel planning compilation that will map the problem of detecting failure into that of probabilistic goal recognition and then employ existing goal recognition tools to estimate the probabilities.

Additionally, for a given state, we also show how one could adapt model space search [27], to generate the minimal set of information about model changes that need to be passed to the user so they are guaranteed not to follow a potentially incorrect plan. One could see such information being provided to the user once the confidence the agent has in the user making a mistake crosses a certain threshold.

However, as we will see, relying purely on the probability of failure has downsides. One would want the assistive agent to make suggestions or intervene early enough that the user can take corrective actions and avoid potential mistakes. For example, in the case of heavy traffic, you would expect the agent to notify the user before they enter the road and not after they are stuck in the traffic (even though the agent can be very confident that the user is following a bad plan in the latter case). In mission-critical non-ergodic domains, such early intervention can be quite critical. As we will see in Section 3.6, while the true problem of generating pre-emptive agent intervention may be best expressed as a Partially Observable Markov decision process (POMDP), we can approximate the problem that is guaranteed to generate a more cautious policy.

To summarize, the main contributions of this chapter are as follows:

- We develop a novel planning compilation that maps the problem of estimating failure probability to a goal recognition problem. (Section 3.4)
- We update the model search algorithm to find the minimal set of model updates that need to be provided to the user to avoid potential failures. (Section 3.5)
- We develop a decision-theoretic method to identify scenarios that require early intervention. (Section 3.6)
- We evaluate the various proposed solutions on several planning benchmarks. (Section 3.8)

3.2 Technical Background

In this chapter, we extend the action definition in Chapter 2 to introduce conditional effects. Hence, each action $a \in A$ is further defined by the tuple
$a = \langle pre(a), \mathit{ceff}(a), add(a), del(a) \rangle$, where $pre(a)$ is a conjunctive propositional formula defined over F, $\mathit{ceff}(a)$ are the set of conditional effects and $add(a) \subseteq F$, $del(a) \subseteq F$ the unconditioned add and delete effects associated with the action. Conditional effects are only applied when the state meets a set of conditions, and unconditioned add and delete effects are always applied when the action is executable. Each conditional effect $e \in \mathit{ceff}(a)$, can be further defined using a tuple of the

form $\langle \mathcal{C}(e), add(e), del(e) \rangle$. Here $\mathcal{C}(e)$, a propositional logical formula, represents the conditions under which the specific effect is applied as part of action execution. The components $add(e)$ and $del(e)$ are the add and delete effects applied when the overall conditional effect is applied. For a given planning model, we can define a transition function $\gamma_{\mathcal{M}} : 2^F \rightarrow 2^F$ as follows:

$$\gamma_{\mathcal{M}}(s) = \begin{cases} (s \setminus \text{del\_set}(s, a)) \cup \text{add\_set}(s, a), & \text{if } s \models \textit{pre} \\ \textit{undefined} & \text{otherwise} \end{cases}$$

Where,

$$\text{add\_set}(s, a) = add(a) \bigcup_{\langle \mathcal{C}(e), add(e), del(e) \rangle \text{ and } \mathcal{C}(e) \models s} add(e)$$

and

$$\text{del\_set}(s, a) = del(a) \bigcup_{\langle \mathcal{C}(e), add(e), del(e) \rangle \text{ and } \mathcal{C}(e) \models s} del(e)$$

Note that when we use a state s in the context of the entailment operator, we are effectively considering the logical formula obtained by considering the conjunction of all positive literals corresponding to fluents that are part of that state and the negative literals corresponding to the fluents that are absent from that state. We will also overload the transition function to apply to action sequences as well. For models where the conditional effect set is empty, we will simplify the entire action representation and represent it using the tuple $\langle pre(a), add(a), del(a) \rangle$. We will generally consider settings where actions have unit costs. However, the proposed approach can easily be extended to cases where actions have differing action costs. A solution to a planning problem is a plan, where a plan $\pi = \langle a_1, ..., a_k \rangle$, is simply an action sequence that satisfies the requirement $\gamma_{\mathcal{M}}(\pi, I) \models G$. We will refer to the plan with the lowest cost (in this case, being equivalent to the shortest) as the optimal plan. Additionally, we will sometime refer to the tuple consisting of just the fluent and action set (i.e., $\langle F, A \rangle$ as the domain of the planning problem.

This chapter also involves finding model updates to generate planning models with the required properties. To do this, we will follow the model-space search paradigm followed by methods like

model reconciliation [9, 27, 28], and similar to these earlier works, we will rely on a model parameterization function to represent each possible model by a set of propositional factors. Since we will be using these parameterization methods over the original models, we will define a parameterization function that only supports model types without conditional effects.

Specifically, we will follow the conventions set by Sreedharan et al . [28] and define a model parameterization function Γ for a set of fluents F and action labels A, which is defined over a set of model parameters $\mathcal{F}^{(F,A)}$, where

$$\begin{aligned}\mathcal{F}^{(F,A)} = \; & \{\textit{init-has-}f \mid f \in F\} \cup \{\textit{goal-has-}f \mid f \in F\} \cup \\ & \bigcup_{a \in A} \{a\textit{-has-positive-prec-}f, a\textit{-has-negative-prec-}f, \\ & \quad a\textit{-has-add-fluent-}f, a\textit{-has-del-fluent-}f \mid f \in F\}.\end{aligned}$$

And the model parameterization function itself is defined as $\Gamma(\mathcal{M})$, which is a mapping to a state $s \subseteq \mathcal{F}$, is defined by

$$\begin{aligned}\tau_I &= \{\textit{init-has-}f \mid f \in I\} \\ \tau_G &= \{\textit{goal-has-}g \mid g \in G\} \\ \tau_{pre_+(a)} &= \{a\textit{-has-positive-prec-}f \mid f \in pre_+(a)\} \\ \tau_{pre\_(a)} &= \{a\textit{-has-negative-prec-}f \mid f \in pre\_(a)\} \\ \tau_{add(a)} &= \{a\textit{-has-add-fluent-}f \mid f \in add(a)\} \\ \tau_{del(a)} &= \{a\textit{-has-del-fluent-}f \mid f \in del(a)\} \\ \tau_a &= \tau_{pre_+(a)} \cup \tau_{pre\_(a)} \cup \tau_{add(a)} \cup \tau_{del(a)} \\ \tau_A &= \bigcup_{a \in A^{\mathcal{M}}} \tau_a \\ \Gamma(\mathcal{M}) &= \tau_I \cup \tau_G \cup \tau_A\end{aligned}$$

Where $pre_+(a)$ is the positive literals that are part of the precondition for a and $pre\_(a)$ the negative ones.

In our work, we map the problem of proactive intervention to that of solving a Partially Observable Markov Decision Process (POMDP) [29–31]. Generally, POMDPs can be described with 8-tuples $\langle S, A, O, T, C, \Omega, \mu_0, \delta \rangle$, where S, A, and O represent the agent's state, action, and observation space, respectively. In each step, the agent takes an action $a \in A$ at a state s. This results in the agent receiving an observation $o \in O$ and the environment state transitioning to a state s'. The transition probabilities for the state are captured by T, which is a set of conditional probabilities of the form $T(s'|s, a)$. C is a cost function that maps all state and action pairs to a cost. Similarly, Ω captures observation probabilities $\Omega(o|s', a)$. μ_0 is the initial distribution over the states and finally, $\delta \in [0, 1)$ is the discount factor. The goal here would be to generate behavior that minimizes the expected discounted total cost. Since the agent doesn't know the exact environment state, it can track its current state estimate by tracking the whole history of observations received or converting them into a posterior distribution over the likelihood of possible states (starting with μ_0 as the prior beliefs). We will refer to the latter representation as a belief state and use $\mathbb{B}$ to represent the set of all belief states. A policy for a POMDP can take the form of a function that maps belief states to actions. In this case, the value function ($V^\pi : \mathbb{B} \to \mathbb{R}$) for a policy π returns the expected total discounted cost received by executing a policy at a belief state, and the Q value function ($Q^\pi : \mathbb{B} \times A \to \mathbb{R}$) returns the expected total discounted cost received by executing an action at a belief state and then following the policy. An optimal policy is the one with the lowest expected total discounted cost, and the corresponding value and Q value functions are represented as V^* and Q^*.

3.3 Running Example

Imagine an AI agent observing a human in a kitchen setup. The agent is tasked with the responsibility of ensuring the safety of the human. This robot monitors the human's actions, gauges task failure risks, and steps in to prevent errors. Figure 3.1 depicts our "Kitchen Domain" example,

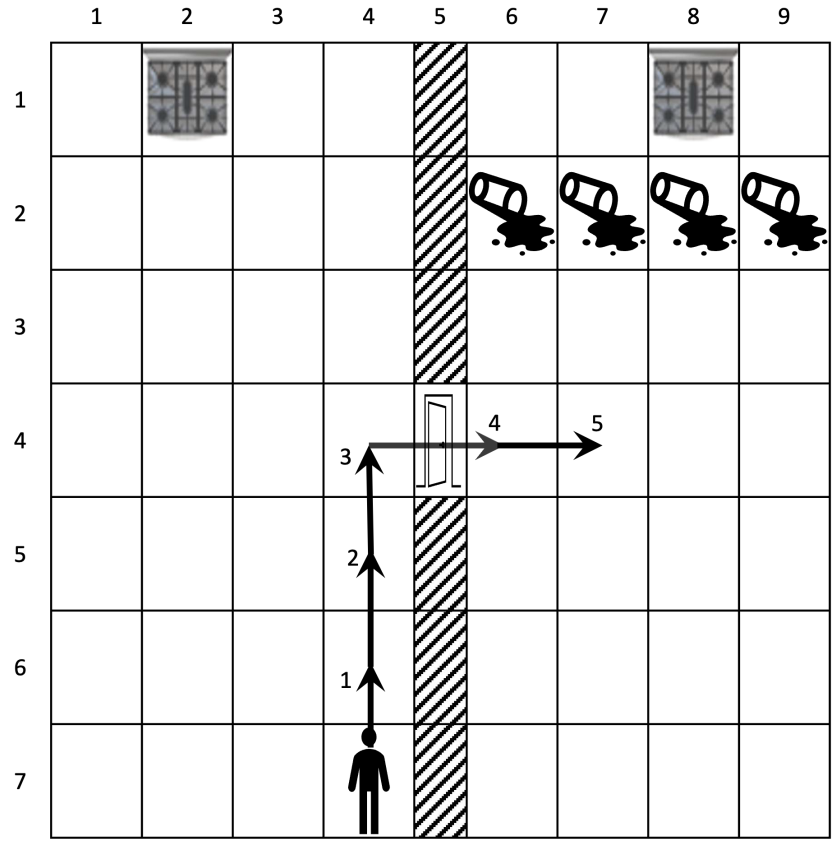


Figure 3.1: A graphical representation of our running example involving an AI agent observing a human operating in a kitchen.

showing a human navigating a specific 9x7 grid kitchen layout. The human starts at point (4,7). The human can move diagonally, vertically, or horizontally through the kitchen, each move being a unit cost action. A wall, marked by diagonal arrows along the y-axis, divides the kitchen. A one-way door at (5,4) allows movement only from the kitchen's left to the right side. The human's objective is to switch on one of the ovens at (2,1) or (8,1). However, unknown to the human, there are oil spills blocking paths to the right oven, and stepping on the spill could potentially lead to the human injuring themselves. The human ignorance about the oil spill may come from the fact that the spill just happened and wasn't present the last time the human visited the room. On the other hand, the assistive agent uses information received from sophisticated sensors placed all around the kitchen to generate an accurate estimate of the overall state. Arrows in Figure 3.1 indicate a set of possible

observations the agent could have received at various points in time. The objective of this chapter is to build an approach that can identify when the human may be headed to a possibly unsafe situation and intervene before they make a mistake.

To ground this example within the framework of our problem, we can see that this example corresponds to a scenario with knowledge asymmetry. Here, the human is unaware of the oil spills, while the AI agent knows about its locations. As such, if we encode the knowledge of the human and the agent into planning models, we would get two distinct models that allow for different valid plans. Particularly in the human model, there are additional plans that involve the human going to the stove on the right.

The first goal of the agent would be to use the observations received from the human activity to determine whether the human is currently executing a plan that involves moving through one of the cells with oil in it. Given the exponential blow-up of plan space for factored planning settings, it would be infeasible to iterate over all potential human plans that could fail explicitly and calculate the probability of the human following one of these invalid plans. As such, we need to develop methods that can approximate this likelihood without the need to enumerate all possible plans.

Once the system has built enough confidence that the human may in fact be heading to an unsafe state, they need to intervene. In this case, the reason why the human is engaging in this unsafe behavior is the underlying knowledge asymmetry; as such, one way to intervene would be to help resolve this asymmetry. Specifically, the agent could inform the user about the oil spill so they don't move into those cells. However, it is worth noting that the agent should only inform the user about the oil spills if it has high confidence the human is headed towards it. If the human received this information when they had no intention of heading to the room on the right, the agent would just be introducing cognitive load on the human's end for no reason (which could potentially lead to the human not trusting the system and potentially ignoring it in the future). Such considerations become even more important in cases where the difference between the human model and the robot model could be quite significant. In such cases, it also becomes important that the agent should be able to recognize what is the minimal set of information that it can give to the human to avoid

potential mistakes. Dumping all the model differences onto the human would again result in the human getting overwhelmed and even potentially ignoring important information.

The question of when to provide the human with the information could be as important as what information to provide. One possibility might be to wait until the agent's confidence crosses a certain threshold or, at the very least, the agent is more confident about the human making a mistake than not making one. However, in the example laid out here, that point is reached once the human crosses through the door. While intervening after that point could help prevent the human from stepping on the oil, it would leave the human trapped in the right room and unable to complete their task. As such, at every time step, the system needs to reason about the possible cost of giving model information when unnecessary (there by incurring some penalty associated with adding cognitive load at the human's end) and the possibility that the user might take an action that would prevent them from ever reaching the goal. To do this form of reasoning, the agent not only needs to consider the probability that the human is following an invalid plan but also consider what exact next steps the user could follow (along with their likelihood). The user would need to use these probabilities with the costs associated with each outcome to determine the right course of action. In this case, this would correspond to the agent informing the human about the oil spill, as soon as the human reaches the door.

3.4 Model Compilation for Failure Detection

We will consider a basic setting where a human operates in an environment. The human maintains some beliefs about the task, which can be represented mathematically via a model $\mathcal{M}^H = \langle F, A^H, I^H, G^H \rangle$. Now, we have an assistive agent tracking human actions and trying to evaluate the likelihood of success. The agent maintains its estimate of the task that we will denote as $\mathcal{M}^R = \langle F, A^\alpha, I^\alpha, G^\alpha \rangle$. We will refer to this pair of models $\mathbb{M}^\alpha = \langle \mathcal{M}^H, \mathcal{M}^R \rangle$, as the *assistive model pair*. To simplify the discussions, we will assume that the action definitions in both models have empty conditions effect sets and share the same set of action labels. Additionally, we will assume that the $\mathcal{M}^R$ is a more accurate task representation than $\mathcal{M}^H$ and that they might differ over

any of the model components. For the running example, the difference between the two models is primarily in the initial state, and the initial state in the human model is missing propositions related to the oil spills next to the right oven.

At a given timestep t, let O_t be the series of actions that the human has performed, now the first step would be to estimate the likelihood that the plan being pursued by the human (i.e., π_H) will fail. More formally, we are trying to estimate

Definition 1. *For a given assistive model pair,* $\mathbb{M}^\alpha = \langle \mathcal{M}^H, \mathcal{M}^R \rangle$*, the* likelihood of failure *given an observation sequence* O_t*, or* $\mathcal{P}_F(O_t, \mathbb{M}^\alpha)$*, is equal to the probability* $\sum_{\pi \in \Pi_F^H} P_H(\pi | O_t, \mathcal{M}^H)$*, where* P_H *gives the probability of the human selecting a plan given their current understanding of the task, and* Π_F^H *gives the set of plans that succeeds in the human model but fails in the agent model, i.e.,* $\Pi_F^H = \{\pi | \gamma_{\mathcal{M}^H}(I^H, \pi) \models G^H, \gamma_{\mathcal{M}^R}(I^\alpha, \pi) \not\models G^R\}$.

Likelihood of failure, thus captures the marginal probability of the human selecting a plan they think will succeed but fails as per the agent's environment model.

Now to calculate this probability, we will employ a compiled model that combines both the human model estimate and the agent one. The basic intuition is that we want to build a model that supports generating all valid plans in the human model but can also track their status in the agent model.

More formally, we will represent this model as $\mathcal{M}^\mathcal{C} = \langle F^\mathcal{C}, A^\mathcal{C}$
$, I^\mathcal{C}, G^\mathcal{C} \rangle$. Here the new fluent $F^\mathcal{C}$ set consists of all the original fluents, a copy for each fluent that will be used to track the agent state, and finally, a proposition called `plan_fail` to detect plan failure, i.e.,

$$F^\mathcal{C} = F \cup \alpha(F) \cup \{\texttt{plan\_fail}\}$$

Where $\alpha(F)$ are the copies of the fluent made for the agent. This means that the compiled model will contain two copies for each original propositional fluent, and we will use $\alpha()$ as a function to map the original fluent to the agent copy. For example, in our running example, the compiled model will have two `stove_on` propositions, the original one and a new proposition $\alpha(\texttt{stove\_on})$.

Coming now to the actions $A^{\mathcal{C}}$, we create a copy for each human action with the same preconditions and effects but now include two new sets of conditional effects, one that corresponds to a case where the action succeeds in the agent model too (and isn't following some previous action that may have failed in the agent model) and one that corresponds to the failure in the agent model. More specifically, for action $a^H \in A^H$ (with a corresponding action a^α in the agent model), we will have a corresponding action $a^{\mathcal{C}} = \langle pre(a^{\mathcal{C}}), \mathit{ceff}(a^{\mathcal{C}}), add(a^{\mathcal{C}}), del(a^{\mathcal{C}})\rangle$, such that

$$pre(a^{\mathcal{C}}) = pre(a^H),\ add(a^{\mathcal{C}}) = add(a^H), \text{and } add(a^{\mathcal{C}}) = del(a^H)$$

Now for the conditional effects, we have

$$\begin{aligned}\mathit{ceff}(a^{\mathcal{C}}) = \{&\langle \alpha(pre(a^\alpha)) \wedge \neg \texttt{plan\_fail}, \\ &\alpha(add(a^\alpha)), \alpha(del(a^\alpha))\rangle, \\ &\langle \neg\alpha(pre(a^\alpha)), \{\texttt{plan\_fail}\}, \{\}\rangle\}\end{aligned}$$

The initial state here consists of the original human initial state and then the copy of the agent's estimate of the initial state

$$I^{\mathcal{C}} = I^H \cup \alpha(I^\alpha)$$

Finally, coming to the goal, the first goal we will consider if one where we are trying to find a plan where the human goal is met and the proposition `plan_fail` is also satisfied.

$$G^{\mathcal{C}} = G^H \cup \{\texttt{plan\_fail}\}$$

A plan that will satisfy this goal would be one that will work on the human model but not on the agent one. This brings us to the first proposition, namely that the failure set (Π_F^H) only consists of plans that satisfy this goal

Proposition 1. *For a plan π is part of Π_F^H, if and only if, $\gamma(I^H, \Pi^{\mathcal{C}}) \models G^{\mathcal{C}}$.*

The proof for this proposition is relatively straightforward. Given the structure of the preconditions and the the add effects, any plan that satisfies the human goal will still result in a state where the part of the state defined with F will satisfy the human goal specification. Additionally, if the plan wasn't valid in the agent model, there must be at least one action in the plan where the failure conditional effect will be executed and thus producing `plan_fail`. The 'only if' part can be proved using a similar reasoning line.

Now to show that we can use the compiled model to calculate the probability, we need to show that the distribution of plans follows the original distribution in the human model. This requires us to adopt a model of decision-making for the human. A natural choice is the noisy-rational model [32, 33], which has been widely used to model human-AI interaction. Under this model, the likelihood of the human selecting an action sequence is given as

$$P_H(\pi|\mathcal{M}^H) \propto e^{-1\times C_{\mathcal{M}^H}(\pi)}$$

Where $C_{\mathcal{M}^H}(\pi) = |\pi|$ if the action sequence is valid (hence a plan) in $\mathcal{M}^H$, else it is equal to ∞. Now if we created a new model $\mathcal{M}^{\mathcal{C}'} = \langle F^{\mathcal{C}}, A^{\mathcal{C}}, I^{\mathcal{C}}, G^H\rangle$, we can see that the distribution plans under decision-making model will match the original distribution for $\mathcal{M}^H$.

Proposition 2. *For noisy rational decision-making models, we can see that*

$$P_H(\pi|\mathcal{M}^H) = P_H(\pi|\mathcal{M}^{\mathcal{C}'})$$

for every action sequence π.

This follows from the fact that every action sequence that is valid in $\mathcal{M}^H$ is valid in $\mathcal{M}^{\mathcal{C}'}$) and vice-versa. Similarly, any action sequence invalid in $\mathcal{M}^H$ is invalid $\mathcal{M}^{\mathcal{C}'}$) and vice-versa. Thus the normalization is done over the same set, and since the cost of valid plans is conserved across the two models, the probabilities stay the same.

Given these two propositions, we can assert that the probability of failure is the probability that the human follows a plan that satisfies the goal $G^H \cup \{\texttt{plan\_fail}\}$, i.e., $G^{\mathcal{C}}$ given an observation

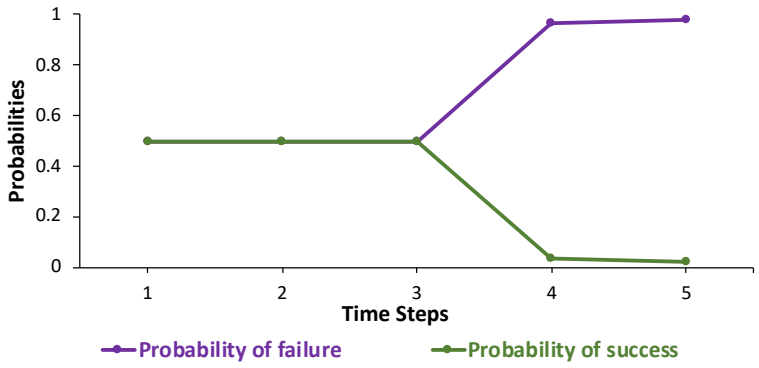


Figure 3.2: Probabilities for goal failure, and not goal failure, as derived for our running example.

O_t, more formally,

$$\mathcal{P}_F(O_t, \mathbb{M}^\alpha) = \sum_{\pi, \gamma(\pi, I^{\mathcal{C}}) \models G^{\mathcal{C}}} P(\pi | O_t, \mathcal{M}^{\mathcal{C}'})$$

Which in turn can be formulated as a probabilistic goal recognition problem between two mutually exclusive goals $G^H \cup \{\texttt{plan\_fail}\}$ and $G^H \cup \{\neg\texttt{plan\_fail}\}$, for the planning domain $\langle F^{\mathcal{C}}, A^{\mathcal{C}} \rangle$, initial state $I^{\mathcal{C}}$ and observation sequence $O^{\mathcal{C}}$. For this purpose, we can direct use methods like the one proposed by Ramírez and Geffner [34], which implicitly uses a noisy rational model. Figure 3.2 illustrates the probabilities associated with task failure and task success for each observed time slice in the running example.

3.5 Minimal Interventional Information

As discussed in the previous sections, the source of human confusion is their misunderstanding regarding the task. Additionally, the agent has access to a more accurate representation of the task. As such, a way to avoid human mistakes might be by informing them about potential differences in tasks. However, informing them about all the differences might be overwhelming and unnecessary. For example, in the running example, knowing the fact that the weather outside the house has changed will not deter the user from going down the wrong path. The goal thus becomes to find the minimal set of model updates that need to be made to the human model, per the agent model, that will ensure that in the resulting model, the human can't select a plan that will result in failure.

Reader's familiar with model reconciliation [9, 27, 35, 36] literature will note the similarity of the description with that of MCE explanations. However, unlike MCE explanations, where the goal is to identify a set of model updates that will ensure that a specific plan is optimal in the updated model, our objective is to ensure that the probability of the human selecting a failing plan is zero, or more formally

Definition 2. *For a given assistive model pair, $\mathbb{M}^{\alpha} = \langle \mathcal{M}^{H}, \mathcal{M}^{R} \rangle$ and an observation sequence O_t, the Minimal Intervention Information or MII is given by a pair of model updates of the form $\mathcal{E} = (\mathcal{E}^{+}, \mathcal{E}^{-})$, such that*

C1 $\mathcal{E}^{+} \subseteq (\Gamma(\mathcal{M}^{R}) \setminus \Gamma(\mathcal{M}^{H}))$ *and* $\mathcal{E}^{-} \subseteq (\Gamma(\mathcal{M}^{H}) \setminus \Gamma(\mathcal{M}^{R}))$

C2 The probability of failure for the updated model pair $\bar{\mathbb{M}}^{\alpha} = \langle \bar{\mathcal{M}}^{H}, \mathcal{M}^{R} \rangle$ is zero, where $\bar{\mathcal{M}}^{H} = \Gamma^{-1}(\bar{\mathcal{M}}^{H} \setminus \mathcal{E}^{-}) \cup \mathcal{E}^{+}$

C3 There exists no pair $\widetilde{\mathcal{E}}$ that satisfies C1 and C2, such that $|\widetilde{\mathcal{E}}^{+}| + |\widetilde{\mathcal{E}}^{-}| < |\mathcal{E}^{+}| + |\mathcal{E}^{-}|$

The model updates which satisfy C1 and C2, but not C3 (hence aren't minimal) will be referred to as simply valid interventional information (or VII)

Following the discussion in the previous section, this is equivalent to finding the minimal set of model updates to be applied to the human model so that the corresponding compiled model is unsolvable when the initial state is set to the one obtained by applying the observations (we will only consider observation sequences that are valid in both human and agent model).

Now we can identify such MII model updates by selecting any existing model-space search or MCE generation algorithm (cf. [9]) and replacing it with the new goal condition. However, unlike the traditional MCE algorithm, the fact that the model updates here are detected for an observation sequence gives rise to two interesting properties.

Proposition 3. *For a model pair $\mathbb{M}^{\alpha}$, if a model update pair $\mathcal{E}$ is VII for an observation sequence $\mathcal{O}$, then it must be a VII for any observation sequence $\hat{\mathcal{O}}$ that contains $\mathcal{O}$ as a prefix. However, the reverse is not true.*

This proposition is rather straightforward given the fact that for a model update pair to be VII, it needs to eliminate all failing plans the human may consider. Any possible failing plan that the human may consider after committing to additional steps must be a subset of this set. However, a VII that merely removes a subset need not work for the original set.

Proposition 4. *For a model pair $\mathbb{M}^{\alpha}$, if a model update pair $\mathcal{E}$ is MII for an observation sequence $\mathcal{O}$, and $\widetilde{\mathcal{E}}$ an MII for an observation sequence $\hat{\mathcal{O}}$ that contains $\mathcal{O}$ as a prefix, then we must have*

$$|\widetilde{\mathcal{E}}^{+}| + |\widetilde{\mathcal{E}}^{-}| \leq |\mathcal{E}^{+}| + |\mathcal{E}^{-}|$$

This proposition can be proved by following a similar line of reasoning as the previous proposition. This brings up an interesting question about trade-off, is it better to just generate a VII in advance that works for all possible actions the human can take in the beginning and just use it when the agent has enough confidence or is it better to wait until the point of failure and calculate an MII [2]. We will be evaluating this trade-off empirically in our evaluation. For the running example, the MII at every point involves informing the human about all the cells with the oil spill.

3.6 Pre-Emptive Intervention

Looking back at the running example (and Figure 3.2), the system is only confident about failure after step 4. At step 4, it can give information about the spilled oil and hopefully stop the human from making the mistake. As mentioned earlier, this would also leave the human stuck inside the room. They can no longer go back out of the room and switch on the other stove. If the goal of the system was to empower the human to actually achieve the goal, as opposed to just preventing them from making a mistake, the information should have been provided much earlier.

This could be the case with many non-ergodic domains, where waiting until the system is confident could result in the human receiving the information too late to actually be used effectively.

[2] One could in theory also argue for a rather conservative approach of always giving a VII upfront. However, in cases where the actual likelihood of the human making a mistake is low, this will not only place an unnecessary cognitive load on the human but could also result in humans losing trust in the system and potentially ignoring future warnings.

Instead of merely reasoning about the likelihood of the human making mistake, it needs to reason about the expected costs involved and use it to drive the decision-making. A natural way to express this reasoning problem would be in terms of a POMDP.

Framed from the point of view of the agent, the actions of the POMDP are limited to a NOOP action (i.e., the agent doesn't intervene), and the action to provide the model updates ($a_{\mathcal{E}}$). The state here includes the current task state as evaluated by the agent, the current task state as evaluated by the human, the human's current belief about the task model, and finally, the plan that is currently being pursued by the user. In our setting, all elements except the current plan are considered observable. The transition function for NOOP action simply involves the execution of the human action and updating both the agent and human's estimate of the state. On the other hand, the choice to give model updates, would update the human model and maybe also the plan they might pursue. The likelihood of the plan selection can be determined by the noisy rational model described before. The cost function could correspond to a failure cost c_f if the system transitions to a state that corresponds to a failure state, cost of model update $c_{\mathcal{E}}(s)$ if the system provides one at state s, and zero for all other transitions (with $c_f >> c_{\mathcal{E}}(s)$ for all s). Note that we will use a more expansive definition of a failure state than a state obtained by the application of an action whose preconditions are not met. In fact, we will consider any state from which the human goal cannot be achieved to be a failure state and also treat it as an absorbing state. We will denote this POMDP as $\mathcal{M}^{(H,\alpha)}$.

Even with the advances in POMDP solvers, exactly solving this problem may not be feasible or practically effective. In fact, even building this model is an expensive process given the need to identify all the states from which the goal is not reachable. However, as we will see, the setting lends itself to be approximated easily.

First, instead of considering individual plans as part of the state, we will aggregate them into a binary variable that represents whether the human is pursuing a plan currently that is bound to fail or not fail (F or $\neg F$). We will use the notation S_t to capture the observable part of the state at a given time step t. Finally, we will leverage the intuition that once the model updates are given, the human is guaranteed to succeed. Thus there is no reason to ever repeat this information.

At any timestep t and state S_t, let the probability of the human following plan that fails be $\mathcal{P}_F$ (shortened from $\mathcal{P}_F(O_t, \mathbb{M}^\alpha)$ for convenience). For the current step, the cost of giving a model update will be just $c_\mathcal{E}$. For NOOP, in cases where the human may be pursuing a plan that will fail, we will look at what the next potential states are and then try to approximate the cost from that state. If no failure has happened in the next state, we will approximate the future cost by using ($c_\mathcal{E}$). This is equivalent to saying that if no error occurs in the next step, the agent will take the safer option of giving a model update. The cost for NOOP action is given as

$$\hat{C}_{NOOP}(S_t) = \mathcal{P}_F \times \sum_{S_{t+1}} P(S_{t+1}|S_t) \times C(S_{t+1})$$

In the above equation, $P(S_{t+1}|S_t)$ effectively considers the probability of transition under each possible plan, along with the likelihood of the plan. However, the formulation provided in Section 3.4 provides us with the tools to calculate it without explicitly enumerating it over all plans. Note that for states S_{t+1} which are failure states, the cost $C(S_{t+1})$ will automatically be c_f and $c_\mathcal{E}(S_{t+1})$ for the rest of the states. We can ignore the $(1 - \mathcal{P}_F)$ term, since there is no cost associated with following a plan that will not fail.

We can show that the above cost calculation is a cautious approximation, as it always overapproximates the cost of performing a NOOP action. More formally, we can state this as

Theorem 1. *For any belief state $B^{(H,\alpha)}$ of the original POMDP, we have $\hat{C}_{NOOP}(S) \geq Q^*(B^{(H,\alpha)}, NOOP)$, where S is the common observable part shared by all states with non-zero probability in $B^{(H,\alpha)}$, and Q^* is optimal Q-value for the corresponding belief space MDP.*

Proof Sketch. This theorem can be proved easily by considering two facts. First, the aggregation maintains all the individual probabilities (as proved in the previous section). Secondly, we can consider a QMDP approximation [37] of the POMDP. For QMDP, we can already see that for NOOP, the Q-value cost will be higher. Secondly, we see that for the MDP estimates where the current state corresponds to a failing plan, the optimal plan would always involve waiting until the

failure step and then providing the model updates. This will be smaller than the value provided here. □

Our approximate decision-making procedure would involve choosing the explanation action, whenever $\hat{C}_{NOOP}(S) > c_{\mathcal{E}}$. Given the results of Theorem 1 and the fact that $Q^*(B^{(H,\alpha)}, a_{\mathcal{E}}) = c_{\mathcal{E}}$, we can guarantee that there will never be a state where the optimal policy would choose to perform the model update action ($a_{\mathcal{E}}$) and the approximate method would choose to perform a NOOP action. This further means that our method is always guaranteed to provide model updates before or at the same time as the optimal policy.

3.7 Related Work

Designing effective pro-active decision-support/assistive systems requires the creation of adaptive systems that can seamlessly comprehend the user's requirements, goals, and limitations. Then be able to directly use that information to come up with suggestions and even corrective actions to help the user achieve their intended objective. As such many of the early works in this direction have focused on the problem of tracking user actions and identifying goals, and providing assistance when necessary.

There currently exists a rich set of works on activity, plan, and goal recognition [38]. In particular, the methods discussed in this chapter are particularly connected to the goal-recognition literature [34, 39] as defined within the context of classical planning. Some works have tried to incorporate the problem of goal recognition directly into the assistance framework (cf. [40–42]). There are also frameworks like CIRL [43], where there is an expectation that humans might take an active role in helping communicate their objectives. On the other hand works in active goal recognition [44], looks at endowing the observer/recognizer agency to improve recognition. There have also been similar systems designed to support people with cognitive disabilities [45]. There is also a wider literature on intervention in both adversarial and cooperative settings [46, 47]. Our proposed algorithm can also be seen as a special case of the intervention problem defined by [48].

However, one set of scenarios where knowledge asymmetry has been explored to a degree is within the problem of providing decision support during the planning phase. Here the human is in the middle of coming up with a plan of action and using a specific decision support interface to formalize their plan. The agent could keep track of this plan and then provide suggestions on other courses of action to take and even provide explanations. Some prominent examples of such systems are the RADAR decision-support systems [49–51]. Such systems have also been explored in the context of risk management [52, 53]. One important example from this group of work is the one used in enterprise settings, which makes use of diverse planning.

In the context of human-robot Interaction, related approaches have been investigated in the context of shared autonomy [54]. In these cases, the control of a system is shared between the user and some form of automated decision-making system. In many of these systems, inferring the user goal is important to ensure the system is helping the user effectively [55]. Other relevant works in this direction include human-robot joint actions [56, 57] and coordination [58]. Most of these works involve social interaction in which two or more individuals/agents coordinate their actions in a shared space to effect a change in the environment, the emphasis of our research is on preventing the human actor from performing actions that could lead to failure. A pivotal element of the proposed proactive assistance system is its ability to reason about human mental states. Consequently, our work aligns closely with the Theory of Mind concepts in AI planning [59–61]. This includes the concept of Perspective Taking—a human ability that enables individuals to see things from another's viewpoint [62].

3.8 Evaluation

For empirical evaluation, our primary goal was to evaluate the two proposed approaches over a set of standard IPC benchmarks. The first method (M1) merely assesses the likelihood of goal failure through observation and intervenes when the likelihood of failure is higher, while the second method (M2) adopts a preemptive approach to identifying points at which to intervene. Through this evaluation, we are interested in testing three main hypotheses.

H1 M2 can help prevent failures that might happen under M1.

H2 The use of satisficing planners for approximating goal likelihood will result in lower runtime without serious degradation in performance with regard to the failure detection step.

H3 We expect the model update size to go down with observation length.

The code for the evaluation and the supplementary files can be found at https://github.com/cglrtrgy/Help.

Setting: We evaluated our method on four IPC domains. For each domain, the original domain description acted as the agent model, while we created the human domain model by randomly deleting five preconditions and delete effects. We created five distinct human models for each domain. Next, we selected five problems per domain. Thus per domain, we had 25 unique pairs of human and agent models (please note that in our terminology, a model contains both the domain and problem information). The optimal planner consisted of the FastDownward implementation [63] of A* search with hmax heuristic and used Lama as our satisficing planner. The observations were also generated by a satisficing planner (lazy-greedy search). All experiments were performed on a machine powered by an Intel Processor running at 3.10 GHz and with 128 GB RAM [64]. In our experiment, the planners were allowed to operate without any time or memory constraints, and we only considered unit cost actions. The cost of failure was taken to be double that of giving a model update.

Results: Table 3.1 gives an overview of the computational characteristics of the proposed methods and also provides all the information relevant to the first two hypotheses. Due to space limitations, we have only reported the average across all 25 instances, but you can find the full data in the supplementary file. Here for each problem instance, we present the results averaged across all five pairs of human and agent models. Specifically, for H1, we are interested in determining whether M2 provides any advantages over M1. Here we see clearly that, the failure rate of M1 (no of times the method intervened after the human made an error) is pretty high throughout all the domains. We skip reporting the failure rate for M2 because it never failed to catch a potential

Table 3.1: Comparison of performance metrics for the methods (M1 and M2) using both optimal and satisficing planners across different domains with varying problems. Here O corresponds to the average observation length; F.S - the average step at which the observation would have resulted in a failure; FR - the ratio of instances in which the method failed to prevent the human from taking a step that results in failure; Int. Step - the average step number in which the method intervened (along with the std deviation). Finally, time reports the time taken for the approach as a whole.

Domains	O	F.S	Optimal					Satisficing				
			M1			M2		M1			M2	
			Int. Step	FR	Time(s)	Int. Step	Time(s)	Int. Step	FR	Time(s)	Int. Step	Time(s)
Elevator	5.40	4.04	3.2 ± 1.8	3.6/5	3.2 ± 2.5	0.95 ± 0.8	77.2 ± 62.3	3.7 ± 1.7	4.4/5	3.6 ± 2.3	0.95 ± 0.8	72.9 ± 61.0
Rovers	9.24	7.04	6.9 ± 1.6	4.8/5	16.6 ± 7.0	2.9 ± 0.8	1800.7 ± 2036.7	6.1 ± 1.9	3.2/5	15.3 ± 9.3	2.3 ± 1.1	674.7 ± 659.5
Gripper	5.60	3.44	3.2 ± 1.6	4.8/5	6.1 ± 7.1	1.2 ± 0.8	154.4 ± 178.7	3.0 ± 1.7	4/5	3.1 ± 2.8	0.8 ± 0.7	104.5 ± 82.7
Zenotravel	5.96	4.16	3.5 ± 1.6	4.2/5	7.1 ± 5.8	0.5 ± 0.4	1552.5 ± 2091.6	2.4 ± 1.7	2.4/5	3.9 ± 4.7	0.3 ± 0.5	903.8 ± 735.7

failure. These results support H1. However, we do see that M2 does take more time than M1 (here, the time doesn't involve the time taken for model update generation). This increase in time could be explained by the large branching factors of IPC domains and the fact that you are calculating the probability of each potential outcome.

This brings us to H2. Here we do see that even when using a satisficing planner, M2 never allowed the user to take a step that fails. Additionally, using a satisficing planner does result in some reduction in the time taken. The domain with the most significant reduction in time was the Rover domain. This again supports hypothesis H2.

For H3, we considered three of the above domains and plotted how the minimal model update size changed with observation length. In Figure 3.3, each graph shows the average length of the Minimal Interventional Information (MII) across the five human-agent model pairs for each problem instance. In the domains Rover and Elevator, we saw that there was a reduction in the minimal intervention information size, with the reduction being most significant in Rover. However, we do not see the same reduction in Gripper, where MII size stayed the same across all observation sizes and problem instances. This might be explained by the fact that Gripper is an extremely compact domain and removing any model component results in an invalid domain. While these results do give some support for H3, it does point to the need to run more evaluation to better characterize how the model update size changes with observation length. Finally, we say that the average time taken across all domains was 278.7 seconds with a standard deviation of 200.5.

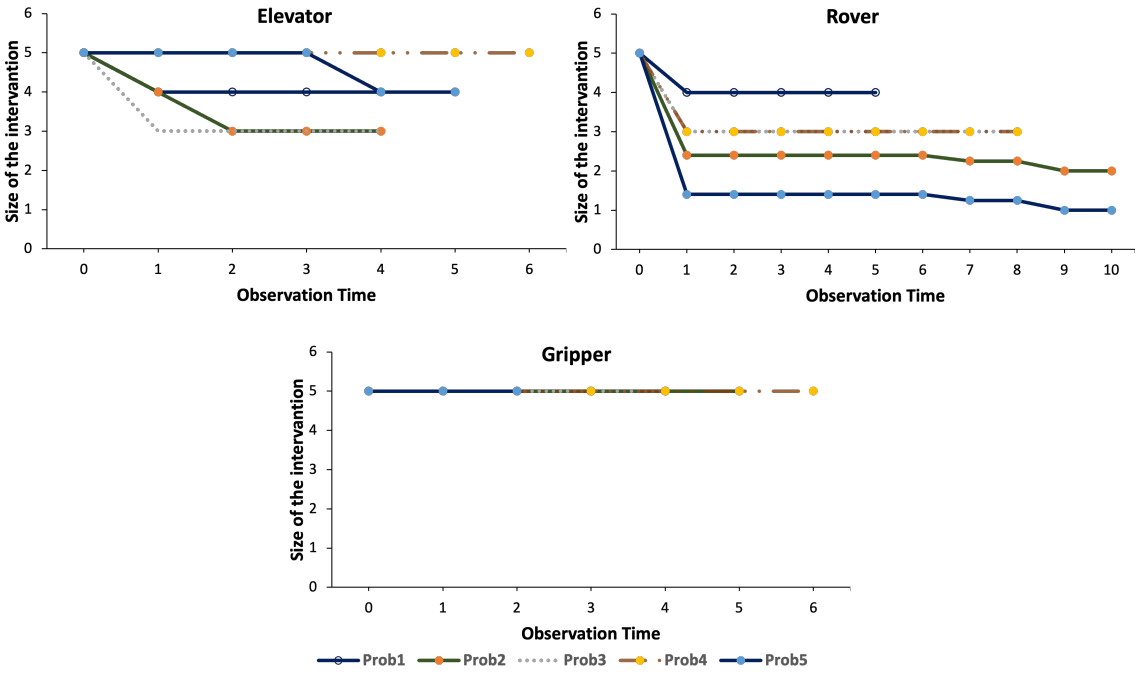


Figure 3.3: Average size of the minimal intervention across three domains as a function of observation length.

3.9 Conclusion

The work presents the first-of-its-kind pro-active assistance system which is able to identify potential human errors in the presence of knowledge asymmetry. We show how we can turn the problem of finding the probability of failure into a goal recognition problem. We additionally build on this basic formalism to support pre-emptive assistance to prevent humans from taking action that may result in them never reaching their goal. We also looked at how we can use model-space search to identify what information should be provided to the user to prevent such errors. In the future, we would like to look at problems where the assistive agent may be embodied or have limitations on how they could intervene. This would mean the agent would need to identify an error early enough that they can actually intervene. In future research, we plan to investigate user preferences regarding the timing of early intervention—such as whether support is preferred at the beginning of the task or just before failure becomes inevitable or costly. We also aim to study how users respond to explanations of varying sizes, including those that may contain task-unrelated information.

Chapter 4

WHO AM I DEALING WITH? EXPLAINING THE DESIGNER'S HIDDEN INTENTIONS

In the previous chapter, I focused on reasoning about the user's mental model and introduced methods for providing proactive support when users hold incorrect or incomplete beliefs about a task. While that work addressed the knowledge asymmetry, it did not consider a critical third stakeholder in real-world AI deployments: the designer. In this chapter, I extend the model reasoning framework by introducing a third stakeholder in human-agent interaction: the system designer. While many explanation techniques in AI focus on aligning the agent's model with the user's expectations—particularly through model reconciliation—this approach overlooks a critical influence on agent behavior: the intentional design choices made by the system's developer. These choices shape both the environment and the agent's decision-making processes, yet they are typically hidden from the end user. I argue that focusing solely on the agent's model in explanations can mislead users, especially when the agent's apparent behavior is influenced by goals or constraints originating from the designer. To address this, I propose a new explanation framework that incorporates designer-centered reasoning, enabling users to understand not only what the agent is doing, but also why the agent was designed to behave that way. This framework builds on existing model-space reasoning by extending the explanation space to include the designer's goals as a distinct model to be reasoned about. To illustrate and evaluate this approach, I formalize a multi-actor explanation model and empirically study its explanatory performance. I also present findings from a user study exploring how people interpret designer-driven explanations and discuss implications for developing more transparent and human-aligned AI systems.

4.1 Introduction

Human agent collaboration enables a varied, distributed set of actors to work together to address problems of greater complexity than those able to be addressed by each actor alone. However, the field of user-agent interaction presents several challenges, including issues related to trust, utilization, and varying degrees of reliance, which can hinder effective collaborations [65–67]. The problem of eXplanaining AI (XAI), often equated with interpretable AI [68], has been developed to address these challenges by enhancing users' understanding of AI systems and fostering more optimal interactions [1, 69]. XAI approaches have aimed to improve people's understanding of the agent model, help people recognize model uncertainty, and support people's calibrated trust in the agent [70, 71].

One XAI approach that aims to bridge this gap is through applying model-reconciliation, ascribing the agents with an approximation of the human's task and goal models [6]. In this approach, the agent explains its actions by leveraging the differences between its own model and the human's mental model of the agent. By reconciling the differences between the agent's model and the human's perception of that model, this method aims to align the human's expectation with the outcomes of the agent's behavior.

However, we posit that focusing on explaining the *agent model*, be it through model reconciliation or otherwise, should not be the goal and may even (unintentionally) create user deception. There is a third actor about whose intentions the user needs to reason. That is the *designer*, the actor who created the AI agent and made key choices about it's design and performance (Figure 4.1). It is rare for AI agents to be deployed and operate in a completely uncontrolled environment. In most cases, at least some aspects of the environment would have been controlled or designed to promote certain forms of agent behavior. Obviously, such a design process would also be applied to the agent itself. When an end-user comes in contact with an AI agent and asks it to achieve certain objectives, the responding agent actions would be heavily dependent on these prior design choices. For the most part the user is oblivious to the design decisions taking place behind the scenes. And while the designer's intentions may overlap with the agent's professed intentions (for example a search and

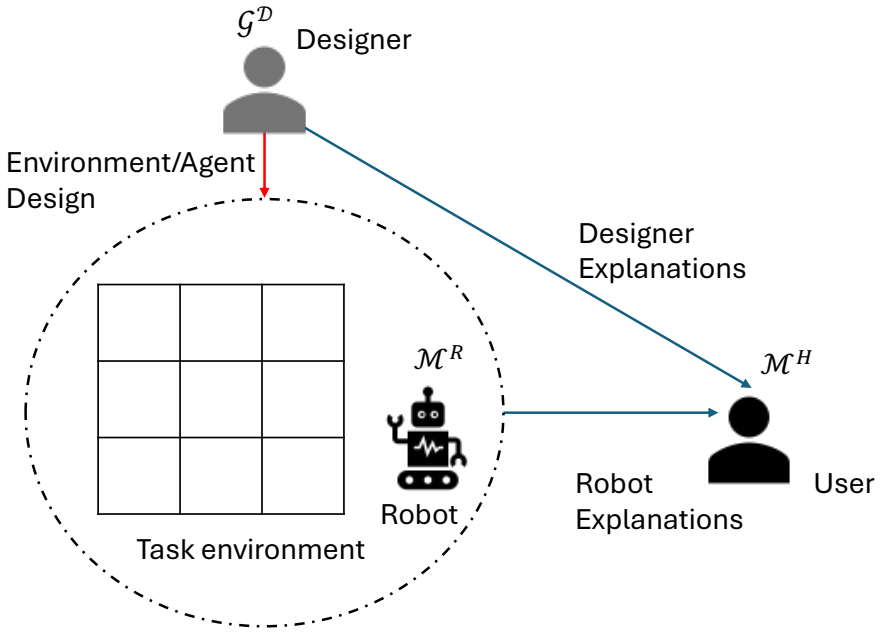


Figure 4.1: An overview of the interactions captured in our framework. The designer tries to modify the environment so that the robot's behavior achieves its underlying goal $\mathcal{G}^{\mathcal{D}}$.

rescue robot's intention is the same as the designers' intention for it - finding the human), they also may not. Consider recommendation systems: the agent's professed intention is to recommend a product that is similar to other products that the user likes and would therefore enjoy. On the other hand the designer's intention is for the user to ultimately spend more money, thereby increasing its utility. It is these goals that are often consciously or unconsciously hidden from the user through the mask of the agent. Same as in deception by misdirection, in which a person is deceived by focusing their attention in the wrong place - when explaining the robot's actions rather than the designer's, these explanations mask the true intention by shifting the user attention to reason about the acting agent [72]. In effect, the explanations people are receiving are not the explanations they need.

To bridge this gap we propose a new explanation framework which explains both designer intentions and acting agent actions. We begin by formalising a multi-actor explanation framework, this time considering the additional aspect of the designer/stakeholder (Section 4.4). We then instantiate our framework on the classical planning Sokoban environment and empirically evaluate the explanation generation performance and efficiency (Section 4.5). Finally, we conduct a proof-

of-concept user study in which users are presented with designer explanations (Section 4.6). We discuss the results and propose ideas for future work.

To summarize, the main contributions of this chapter are as follows:

- We identify and formalize the role of the system designer as a distinct and influential stakeholder in AI explanations, whose intentions may differ from those of the agent or the user.
- We propose a new explanation framework that incorporates reasoning over designer goals, extending traditional model reconciliation beyond user-agent dynamics.
- We instantiate this framework in a classical planning domain and demonstrate how designer-aware explanations can clarify agent behavior shaped by hidden design choices.
- We conduct a user study to evaluate how designer explanations impact trust, understanding, and user perception, providing initial evidence for their value in human-agent collaboration.

4.2 Related Work

As AI applications become more widespread and even deployed in safety-critical settings, there is increasing recognition that these systems need to be capable of explaining their decisions [73]. While some of the early works in explanations go back to expert systems [74, 75], the more recent interest has been particularly spurred by the inscrutability of the state-of-the-art AI models [70]. Even in a subfield of AI, like planning, we see a pretty large number of works related to explanations (cf. [76, 77]). However, recent works have also highlighted how explanations could lead to misuse of the system. This includes how even simple explanations could cause people to place unwarranted trust in the system [78] and even accept system decisions against their best interest [79]. Despite the rapid advancements in the field, most explanation efforts remain focused on a straightforward dyadic settings [6]. To the best of our knowledge, no other works in XAI has looked at incorporating the influence the designer has on the final behavior.

4.3 Running Example

To illustrate our approach consider Figure 4.2. In this scenario, there are three actors; (1) *the operator* who is tasked with helping robots navigate from a start position to a goal location by choosing one of several possible routes; (2) the robot whose goal is to reach the flag safely and with the least possible actions. The robot can move one square at a time, but only in the direction its tires are facing. It can rotate by changing its facing direction 90 degrees at a time and each action has a uniform cost; and (3) the designer of the robot and environment. The designer's goal is for the robot (and therefore, operator) to view the ads. To achieve this, the designer positioned the robot facing left and placed the boxes to make the passage too narrow, rendering the purple path invalid. Note that the operator does not know that the robot is too wide to safely pass through narrow passages.

4.4 Environment Design Problem and Explanations

To solve the problem of designer explanations we adopt a two-level explanation strategy. First we explain why, given the current agent model and environment state, the current plan is the right course of action. Then, we explain how aspects of the current model which were under the designer's control, gave rise to agent behavior that helped achieve the designer's goal.

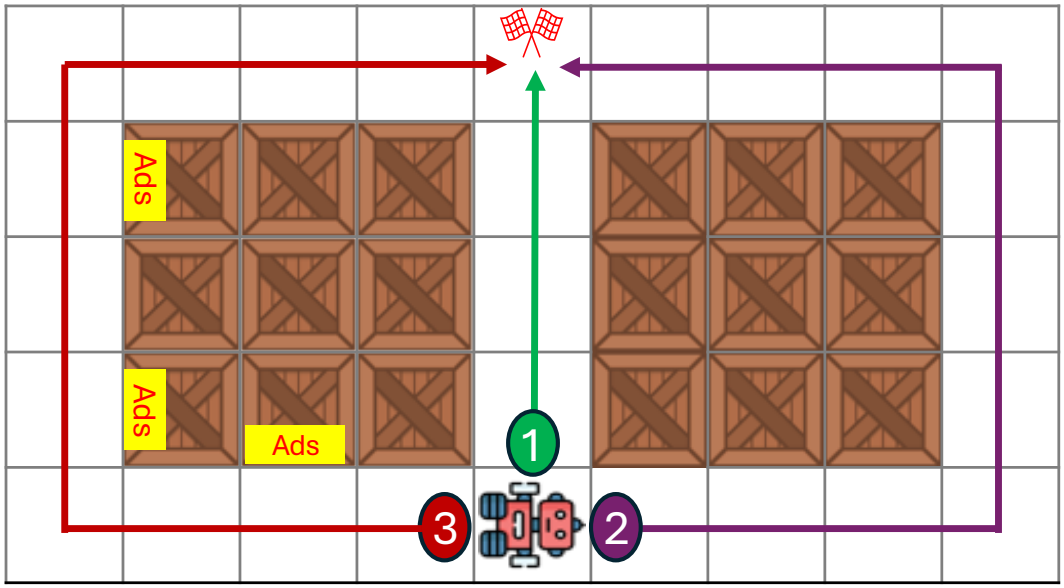


Figure 4.2: Example of decision making task

We begin by formally defining the underlying design problem that influenced the environment and gave rise to the particular agent behavior. We define the underlying designer goal as ($\mathcal{G}^{\mathcal{D}}$), the goal the designer wants the AI agent (henceforth referred to as the robot[3]) to achieve in the course of its operation (defined by a set of goals $\mathbb{G}^{\mathcal{R}}$). The designer has access to a set of actions ($\mathcal{A}^{\mathcal{D}}$) that they can employ to change both the environment and the robot. In our running example, $\mathbb{G}^{\mathcal{R}}$ ensures that the robot pass near 'Ads,' while $\mathcal{A}^{\mathcal{D}}$ includes placing boxes action (changing the environment) and rotating the robot's facing direction action (changing the robot).

The solution to the design problem is a sequence of designer actions that results in an environment where the behavior selected by the robot will satisfy the designer's goal. For convenience, we assume that the designer's actions only change the initial state of the robot and that the robot is an autonomous agent, using an independent optimal decision-making process to identify the optimal course of action to achieve its goal, given the environment. Note that the influence that the designer asserts on the agent is an indirect one, where they set up the environment so that the autonomous agent ends up also achieving the designer's hidden intent. This is a special case of mechanism design, which is an important topic within game theory.

We define the design problem, and solution, as follows:

Definition 3. *An environment design problem is characterized by the tuple,*
$\mathcal{DP} = \langle F, A^R, I^0, \mathcal{G}^{\mathcal{D}}, \mathcal{A}^{\mathcal{D}}, \mathbb{G}^{\mathcal{R}}, \Lambda \rangle$*, where:*

- $F, A^R, \mathbb{G}^{\mathcal{R}}$ *- Fluents used in the robot model, robot actions, and potential goals the robot might come across, respectively.*
- I^0 *- Initial, unedited, state.*
- $\mathcal{G}^{\mathcal{D}}, \mathcal{A}^{\mathcal{D}}$ *- Designer goal and actions.*
- Λ *- Transition function related to the application of designer action in the initial state such that,* $\Lambda : 2^F \times \mathcal{A}^{\mathcal{D}} \rightarrow 2^F$.

[3]Even though we refer to the AI agent as a robot, no part of this approach is limited to physically embodied agents.

Definition 4. *A solution to the environment design problem is a designer plan, which consists of a sequence of designer actions $\pi^{\mathcal{D}} = \langle a_1^{\mathcal{D}}, ..., a_k^{\mathcal{D}} \rangle$, such that the resultant initial state is $I^{\mathcal{D}} = \Lambda(I^0, \pi^{\mathcal{D}})$, and allows that for every $G' \in \mathbb{G}^{\mathcal{R}}$, the resultant robot model $\mathcal{M}' = \langle F, A^R, I^{\mathcal{D}}, G' \rangle$ is of the form that every plan π' optimal in $\mathcal{M}'$ satisfies $\mathcal{G}^{\mathcal{D}}$, i.e, $\delta^{\mathcal{M}'}(I^{\mathcal{D}}, \pi') \models \mathcal{G}^{\mathcal{D}}$.*

Following this definition, we can define the form of explanations we would want for this setting. We refer to these explanations as the *robot-designer explanations*. We use model reconciliation [9, 27, 35] as our base explanatory framework and extend it to support explaining the role of design choices. One aspect to remember here is that, as discussed before, the designer may want to hide the design influence from the user. Therefore an objective explainer can rarely be the designer or a system sanctioned or built by the designer. A more plausible role these systems could take would be that of an external, post-hoc system being employed by the user to make sense of the decision of an automated system. This necessarily restricts the information the explanation generation system might have access to. We generally adhere to information that can either be learned (possibly through observation of the robot and environment) or can be hypothesized directly from observed behavior.

The robot-designer explanations are based on the robot model ($\mathcal{M}^R$), the user's mental model of the robot ($\mathcal{M}^H$), the plan to be explained (π^R), the set of fluents whose values the designer could potentially influence ($\mathcal{F}^{\mathcal{D}} \subseteq F$), and the potential designer goal ($\mathcal{G}^{\mathcal{D}}$). As this is the first work considering designer intentions, we will focus on cases where a single fluent set and an individual hypothesis for the designer's goal are provided initially.

As discussed, our final explanation consists of two parts. (1) Explaining the robot behavior given the current environment, leveraging model reconciliation explanations [9]. Here, the explanation consists of information about the robot model, $\mathcal{M}^R$, which, when incorporated into the human model, $\mathcal{M}^H$, will allow the user to correctly evaluate the robot plan against the robot goal. (2) Model updates and counterfactual explanation [80]. The model updates will establish the fact that in the current model, the optimal plan always includes achieving the designer's goal, $\mathcal{G}^{\mathcal{D}}$. The counterfactual explanation will point out a set of initial state fluent values whose value change could

result in an optimal plan for the robot goal that no longer achieves $\mathcal{G}^{\mathcal{D}}$. More formally, we define the explanation problem as follows:

Definition 5. *A robot-designer explanation problem is represented as a tuple* $\mathcal{DEP} = \langle \mathcal{M}^R, \mathcal{M}^H, \pi^R, \mathcal{F}^{\mathcal{D}}, \mathcal{G}^{\mathcal{D}} \rangle$*. Here, the primary components for the robot explanation include the robot model* $\mathcal{M}^R$*, the human model* $\mathcal{M}^H$*, and the robot plan* π^R*. The designer information being used includes the fluents that can be changed by the designer (*$\mathcal{F}^{\mathcal{D}}$*) and the designer goal (*$\mathcal{G}^{\mathcal{D}}$*). The* $\mathcal{M}^R$ *used here is assumed to be a result of an environment design process.*

This defines a robot-designer explanation problem, and now we can formally define what a solution to this problem, i.e., an explanation, looks like.

Definition 6. *For a given robot-designer explanation problem* $\mathcal{DEP} = \langle \mathcal{M}^R, \mathcal{M}^H, \pi^R, \mathcal{F}^{\mathcal{D}}, \mathcal{G}^{\mathcal{D}} \rangle$*, a valid explanation consists of tuple of the form* $\mathbf{E} = \langle \mathcal{E}_R, E_{\mathcal{D}} \rangle$*, where* $\mathcal{E}_R$ *is the robot explanation and* $E_{\mathcal{D}}$ *is the designer explanation of the form* $E_{\mathcal{D}} = \langle \mathcal{E}_\mu, \mathcal{E}_\kappa \rangle$*, such that the following conditions are met*

C1- For $\mathcal{E}_R = \langle \mathcal{E}_R^+, \mathcal{E}_R^- \rangle$, $\mathcal{E}_\mu = \langle \mathcal{E}_\mu^+, \mathcal{E}_\mu^- \rangle$ *and* $\mathcal{E}_\kappa = \langle \mathcal{E}_\kappa^+, \mathcal{E}_\kappa^- \rangle$*, are such that,*

1. $\mathcal{E}_R^+ \subseteq \Gamma(\mathcal{M}^R) \setminus \Gamma(\mathcal{M}^H)$ *and* $\mathcal{E}_R^- \subseteq \Gamma(\mathcal{M}^H)$
2. $\mathcal{E}_\mu^+ \subseteq \Gamma(\mathcal{M}^R) \setminus \Gamma(\mathcal{M}^H)$ *and* $\mathcal{E}_\mu^- \subseteq \Gamma(\mathcal{M}^H)$
3. $\mathcal{E}_\kappa^+ \cup \mathcal{E}_\kappa^- \subseteq \mathcal{F}^{\mathcal{D}}$.

C2- The plan π^R *is optimal for model* $\mathcal{M}^H + \mathcal{E}_R$.

C3- For the model $\mathcal{M}^H + \mathcal{E}_R + \mathcal{E}_\mu$ *there exists no optimal plan such that it doesn't satisfy* $\mathcal{G}^{\mathcal{D}}$ *and* π^R *is still optimal.*

C4- for model $\mathcal{M}' = \mathcal{M}^H + \mathcal{E}_R + \mathcal{E}_\mu + \mathcal{E}_\kappa$*, there exists an optimal plan* π'*, such that* $\delta^{\mathcal{M}'}(I', \pi') \not\models \mathcal{G}^{\mathcal{D}}$ *and* $C^{\mathcal{M}'}(\pi') \leq C^{\mathcal{M}'}(\pi')$.

In the case of the designer's explanation $\mathcal{E}_\mu$ captures the model update part, and $\mathcal{E}_\kappa$ the counterfactual part. Condition C1 sets the requirements for the model updates provided as part of

the robot explanation to be consistent with $\mathcal{M}^R$. Similarly, it states that the designer explanation component should only consider changing fluents that are under the designer's control. The next three conditions, C2-C4, ensure that each explanation component meets its required purpose. The robot's explanation on its own shows why the current plan is optimal in the given environment. Returning to Figure 2, the robot's explanation would be: 'The robot is wide. The robot can only move through spaces that are wide.' This would help the user understand why the red path is optimal rather than the green path.

The first part of the designer's explanation will establish the fact that achieving the designer's goal will always be part of any optimal strategy in the given environment. The second part of the designer's explanation identifies some initial state fluents that the designer could influence. If changed, these fluents' values could allow for optimal plans that might not have met the designer's goals. This communicates the counterfactual cases where the designer's goals could have been avoided. Consider Figure 2, if the designer only changed the robot's facing direction from right to left without placing boxes on the left side of the green path, the robot would follow the green path which would be optimal.

Note that not all valid explanations may be equally effective or preferred by the user. After all, selectivity has been widely recognized as one of the central characteristics of preferred explanations [70]. As such, we need to minimize the amount of information passed to the user. Rather than optimizing the two components separately, we aim to minimize the total amount of information passed to the user.

Definition 7. *A given explanation pair,* $\mathbf{E} = \langle \mathcal{E}_R, E_{\mathcal{D}} \rangle$*, is considered a minimal explanation for the robot-designer explanation problem* $\mathcal{DEP}$*, if (a) it is valid explanation for* $\mathcal{DEP}$*, i.e., it meets the conditions listed in Definition 6 with respect to* $\mathcal{DEP}$ *and (b) there exists no other valid explanation* $\hat{E} = \langle \hat{\mathcal{E}}_R, \hat{E}_{\mathcal{D}} \rangle$*, such that*

$$|\hat{\mathcal{E}}_R^+| + |\hat{\mathcal{E}}_R^-| + |\hat{\mathcal{E}}_\mu^+| + |\hat{\mathcal{E}}_\mu^-| + |\hat{\mathcal{E}}_\kappa^+| + |\hat{\mathcal{E}}_\kappa^-| <$$
$$|\mathcal{E}_R^+| + |\mathcal{E}_R^-| + |\mathcal{E}_\mu^+| + |\mathcal{E}_\mu^-| + |\mathcal{E}_\kappa^+| + |\mathcal{E}_\kappa^-|$$

We measure the cost associated with each explanation by the number of model updates it communicates. This measure was motivated both by its intuitiveness and generality. However, it is worth noting that we can easily associate an arbitrary cost function with each model update with minimal changes to the formulation and the explanation generation algorithm we will discuss next.

Before we discuss the algorithm we will first introduce some important properties of the robot-designer explanation.

Proposition 5. *For a given given robot-designer explanation problem* $\mathcal{DEP} = \langle \mathcal{M}^R, \mathcal{M}^H, \pi^R, \mathcal{F}^{\mathcal{D}}, \mathcal{G}^{\mathcal{D}} \rangle$*:*

1. *We can guarantee that a model update* $\mathcal{E}_R$ *exists, such that, both conditions C1 and C2 are met.*

2. *We can guarantee that a set of model update* $\mathcal{E}_\mu$ *exists, such that, both conditions C1 and C3 are met.*

3. *There might not exist a model update* $\mathcal{E}_\kappa$ *that meets C1, & C4.*

The first property arises from the fact that one can guarantee to meet $C1$ and $C2$, by just communicating the complete model $\mathcal{M}^R$. After all, $\mathcal{M}^R$ identified π^R as the optimal plan.

Similarly, since the current problem is the result of a design process, communicating the entire model will ensure that all possible optimal plans will satisfy the designer's goal, thus guaranteeing the existence of $\mathcal{E}_\mu$.

However, the counterfactual part of the designer explanation isn't guaranteed because the designer's goal could have been of the form that the initial state already guarantees its achievement (i.e., the solution to the original design problem was an empty sequence). Additionally, $\mathcal{F}^{\mathcal{D}}$ could be empty or might not have influenced the behavior of the plan.

The next property will deal with comparing robot explanations found as part of the minimal robot-designer explanation and minimally complete explanation (MCE) for the plan [9]:

Proposition 6. *Let $\mathcal{E}_R$ be part of a minimal robot-domain explanation for a problem $\mathcal{DEP} = \langle \mathcal{M}^R, \mathcal{M}^H, \pi^R, \mathcal{F}^{\mathcal{D}}, \mathcal{G}^{\mathcal{D}} \rangle$. We can guarantee that $|\mathcal{E}_R| \geq |\mathcal{E}_{MCE}|$, where $\mathcal{E}_{MCE}$ is the MCE for π^R given the models $\mathcal{M}^R$ and $\mathcal{M}^H$.*

This property follows from the fact that any robot explanation for $\mathcal{DEP}$ meets the criteria for an MCE and, as such, can be a candidate for an MCE explanation. Note that the robot explanation here is found as part of the overall minimal solution to $\mathcal{DEP}$. Since these are not required for MCE, it is possible to find smaller model updates set that meets the requirement for MCE but not those for the minimal explanation for $\mathcal{DEP}$.

Finally, another explanation property in the model reconciliation framework is that of *monotonicity* [9]. Namely, an explanation is non-monotonic if additional model updates can invalidate it. In our case, it means adding new model updates to the robot explanation and/or designer explanation component, causing it to violate conditions C2 and or C3.

Proposition 7. *A minimal robot-designer explanation E^* for a problem $\mathcal{DEP}$ need not be monotonic.*

The non-monotonicity of the robot explanation component directly follows the argument proposed through the concept of model reconciliation. New information about new preconditions (while missing information about other actions add effects), might cause one to think a plan previously thought to be optimal is now invalid (thus violating C2). Similar arguments can also be made for the model update part of the designer explanation. For the counterfactual component, C4 might have been satisfied by initial state changes because it now made new plans possible. However, additional changes to the initial state could invalidate those plans, hence violating C4.

4.4.1 Identifying Minimal Robot-Designer Explanation Set

To identify the minimal robot-designer explanation we will use model space search which has been used previously for both model reconciliation [9] and design [81]. We will start by focusing on a breadth-first search. However, one could convert it into an informed search by incorporating various model-space search heuristics considered in the literature (cf. [9]). Algorithm 1 provides the

pseudo-code for our method. Note that the goal of the robot-designer explanation is not the same as other model reconciliation explanations, such as MCE. Here, we are tracking three sets of model updates, one of which is counterfactual changes. Here, the successor generator corresponding to the last explanation component is only considered in states where the first two components are found. Finally, unlike the MCE, we can't guarantee that an explanation always exists.

We start by initializing the search queue with empty explanations. At each node expansion step, we test for conditions C2, C3 and C4 from Definition 6: Test_for_C2($\mathcal{M}^H + \hat{E}_R$) checks to see if π^R is optimal in the resultant model; Test_for_C3($\mathcal{M}^H + \hat{\mathcal{E}}_R + \hat{\mathcal{E}}_\mu$) tests if in the model resulting from applying $\hat{\mathcal{E}}_R + \hat{\mathcal{E}}_\mu$ all optimal plans here would satisfy $\mathcal{G}^{\mathcal{D}}$; Finally, Test_for_C4($\mathcal{M}^H + \hat{\mathcal{E}}_R + \hat{\mathcal{E}}_\mu + \hat{\mathcal{E}}_\kappa$) checks if the introduction of $\hat{\mathcal{E}}_\kappa$, results in at least one optimal plan that doesn't satisfy $\mathcal{G}^{\mathcal{D}}$. We don't need to test for C1 explicitly because our successor function guarantees that any model updates considered will satisfy it. At the end of the search, if no robot-designer explanation was identified, a minimal robot explanation will be returned and the model update component of the designer explanation that it came across during the search (which are guaranteed to exist).

Proposition 8. *Algorithm 1 is guaranteed to return the optimal robot-designer explanation, if one exists.*

The proof is pretty straightforward. Even though, the successor function skips certain possible successors, the completeness or optimality of the search algorithm is not affected. This is due to the fact that model updates are inherently captured as set operations; as such it is commutable. The search still covers all unique model update sets, in the order of the total size of model updates.

4.5 Computational Experiments on IPC Domains

4.5.1 Evaluation Setting

We implemented our proposed framework to evaluate the computational constraints of our approach over different baselines. In our experiments, we assign uniform unit costs for all model updates and utilize Fast Downward with landmark-cut heuristic for model-space searches. The

Algorithm 1 Find the minimal explanation for a robot-designer explanation problem $\mathcal{EP}$.

Input: $\mathcal{EP} = \langle \mathcal{M}^R, \mathcal{M}^H, \pi^R, \mathcal{F}^{\mathcal{D}}, \mathcal{G}^{\mathcal{D}} \rangle$
Output: An explanation E.
Fringe $\leftarrow Queue()$
$\hat{\mathcal{E}}_R \leftarrow \langle \{\}, \{\} \rangle, \hat{\mathcal{E}}_\mu \leftarrow \langle \{\}, \{\} \rangle, \hat{\mathcal{E}}_\kappa \leftarrow \langle \{\}, \{\} \rangle$
Fringe.add($\langle \hat{E}_R, \hat{\mathcal{E}}_\mu, \hat{\mathcal{E}}_\kappa \rangle$)
while Fringe not empty **do**
  $\hat{\mathcal{E}}_R, \hat{\mathcal{E}}_\mu, \hat{\mathcal{E}}_\kappa \leftarrow$ Fringe.pop()
  C2_condition_met $\leftarrow$ Test_for_C2($\mathcal{M}^H + \hat{\mathcal{E}}_R$)
  C3_condition_met $\leftarrow$ Test_for_C3($\mathcal{M}^H + \hat{\mathcal{E}}_R + \hat{\mathcal{E}}_\mu$)
  C4_condition_met $\leftarrow$ Test_for_C4($\mathcal{M}^H + \hat{\mathcal{E}}_R + \hat{\mathcal{E}}_\mu + \hat{\mathcal{E}}_\kappa$)
  if All three conditions met **then**
    return $\langle \hat{\mathcal{E}}_R, \hat{\mathcal{E}}_\mu, \hat{\mathcal{E}}_\kappa \rangle$
  else
    for $f \in \Gamma(\mathcal{M}^R) \setminus \Gamma(\mathcal{M}^H)$ **do**
      $\hat{\mathcal{E}}_R^+, \hat{\mathcal{E}}_R^- \leftarrow \hat{\mathcal{E}}_R$; $\hat{\mathcal{E}}_\mu^+, \hat{\mathcal{E}}_\mu^- \leftarrow \hat{\mathcal{E}}_\mu$
      $\bar{\mathcal{E}}_R^+ \leftarrow \hat{\mathcal{E}}_R^+ \cup \{f\}$; $\bar{\mathcal{E}}_\mu^+ \leftarrow \hat{\mathcal{E}}_\mu^+ \cup \{f\}$
      $\bar{\mathcal{E}}_R \leftarrow \langle \bar{\mathcal{E}}_R^+, \hat{\mathcal{E}}_R^- \rangle$; $\bar{\mathcal{E}}_\mu \leftarrow \langle \bar{\mathcal{E}}_\mu^+, \hat{\mathcal{E}}_\mu^- \rangle$
      Fringe.push($\langle \bar{\mathcal{E}}_R, \hat{\mathcal{E}}_\mu, \hat{\mathcal{E}}_\kappa^+ \rangle$)
      Fringe.push($\langle \hat{\mathcal{E}}_R, \bar{\mathcal{E}}_\mu, \hat{\mathcal{E}}_\kappa^+ \rangle$)
    end for
    for $f \in \Gamma(\mathcal{M}^H) \setminus \Gamma(\mathcal{M}^R)$ **do**
      $\hat{\mathcal{E}}_R^+, \hat{\mathcal{E}}_R^- \leftarrow \hat{\mathcal{E}}_R$; $\hat{\mathcal{E}}_\mu^+, \hat{\mathcal{E}}_\mu^- \leftarrow \hat{\mathcal{E}}_\mu$
      $\hat{\mathcal{E}}_R^- \leftarrow \hat{\mathcal{E}}_R^- \cup \{f\}$; $\hat{\mathcal{E}}_\mu^- \leftarrow \hat{\mathcal{E}}_\mu^- \cup \{f\}$
      $\bar{\mathcal{E}}_R \leftarrow \langle \hat{\mathcal{E}}_R^+, \bar{\mathcal{E}}_R^- \rangle$; $\bar{\mathcal{E}}_\mu \leftarrow \langle \hat{\mathcal{E}}_\mu^+, \bar{\mathcal{E}}_\mu^- \rangle$
      Fringe.push($\langle \bar{\mathcal{E}}_R, \hat{\mathcal{E}}_\mu, \hat{\mathcal{E}}_\kappa^+ \rangle$)
      Fringe.push($\langle \hat{\mathcal{E}}_R, \bar{\mathcal{E}}_\mu, \hat{\mathcal{E}}_\kappa^+ \rangle$)
    end for
    if C2 and C3 met **then**
      for $f \in \mathcal{F}^{\mathcal{D}}$ **do**
        $\hat{\mathcal{E}}_\kappa^+, \hat{\mathcal{E}}_\kappa^- \leftarrow \hat{\mathcal{E}}_\kappa$
        if $f \in I^R$ **then**
          $\hat{\mathcal{E}}_\kappa^+ \leftarrow \hat{\mathcal{E}}_\kappa^+ \cup \{f\}$
        else
          $\hat{\mathcal{E}}_{\mathcal{D}}^- \leftarrow \hat{\mathcal{E}}_{\mathcal{D}}^- \cup \{f\}$
        end if
        $\hat{\mathcal{E}}_\kappa^+ \leftarrow \langle \hat{\mathcal{E}}_\kappa^+, \hat{\mathcal{E}}_{\kappa_{\mathcal{D}}}^- \rangle$
        Fringe.push($\langle \hat{\mathcal{E}}_R, \hat{\mathcal{E}}_\mu, \hat{\mathcal{E}}_\kappa \rangle$)
      end for
    end if
  end if
end while
return Minimal $\hat{\mathcal{E}}_R$ and $\hat{\mathcal{E}}_\mu$ that satisfies C2 and C3.

robot model is derived from the original IPC domains and problem instances, while the human model is generated by randomly removing preconditions or delete effects from the original domain. We evaluate five classical planning domains from the planning literature[4], each with four problem instances, and three human domains. All evaluations were conducted on a system equipped with 16GB RAM and an Apple M1 3.2GHz CPU.

To create the designer goals, we first obtain the robot's optimal plan for each domain problem instance pair and identify the final predicates available after iterating the robot's optimal plan to the goal state. We then randomly select one predicate that is not part of the initial state to serve as the designer goal.

4.5.2 Results

Our primary goal with these experiments was to both evaluate the computational characteristics of our proposed algorithm over different baselines, and also to compare it against the standard model-reconciliation problem implementation. Table 4.1 compares the time taken to find minimal complete explanations (MCE) [27] with the time taken to compute minimal explanations for a robot-designer explanation using our proposed algorithm. The results presented are averaged across the different human models. Note that the use of different human models means that we have different human plans, and explanations. One of the reasons MCEs makes for a useful point of comparison is because, similar to our approach, MCE algorithms also leverages model-space search. However, unlike our approach, they search for potential explanations over a much smaller space. As expected, the time taken to find MCE was much smaller than the time required to find the robot-designer explanation. It is worth noting that the robot explanations found in these domains were exactly the same as the size of MCE. Additionally, the designer explanations found here consisted only of the counterfactual component $\mathcal{E}_\kappa$ (i.e., $|E_\mathcal{D}| = |\mathcal{E}_\kappa|$). Even though the time taken by our method was larger we did see that, across all the domains, the time was small enough to be used effectively.

[4]http://ipc.icaps-conference.org

Table 4.1: Performance metrics across IPC domains, averaged across human-domains. $|\pi^R|$ and $|\pi^H|$ denote the lengths of robot and human plans, respectively. The table has two main columns: the first presents $|\mathcal{E}|$, the minimal complete explanation (MCE) length, and its computation time (seconds). The second shows metrics for our algorithm, including $|\mathcal{E}_R|$ and $|\mathcal{E}_D|$, the lengths of robot and designer explanation parts, along with their computation time (seconds).

				MCE		Robot-Designer			
Domains	Problem	$\|\pi^R\|$	$\|\pi^H\|$	$\|\mathcal{E}\|$	Time(s)	$\|\mathcal{E}_R\|$	$\|E_D\|$	$\|\mathcal{E}_R\|+\|E_D\|$	Time(s)
Depots	prob1	10	5.3 ± 2.3	1.0 ± 0.0	0.3 ± 0.0	1.0 ± 0.0	1.0	2.0 ± 0.0	29.1 ± 26.5
	prob2	5	2.7 ± 1.2	1.0 ± 0.0	0.3 ± 0.0	1.0 ± 0.0	1.0	2.0 ± 0.0	46.7 ± 19.9
	prob3	10	5.0 ± 1.7	1.0 ± 0.0	0.3 ± 0.0	1.0 ± 0.0	1.0	2.0 ± 0.0	50.8 ± 28.3
	prob4	5	2.3 ± 0.6	1.0 ± 0.0	0.3 ± 0.0	1.0 ± 0.0	1.0	2.0 ± 0.0	18.7 ± 10.7
Driverlog	prob1	7	3.3 ± 0.6	1.0 ± 0.0	0.2 ± 0.1	1.0 ± 0.0	2.0	3.0 ± 0.0	75.8 ± 10.1
	prob2	9	7.0 ± 1	1.3 ± 0.6	0.2 ± 0.1	1.3 ± 0.6	1.0	2.3 ± 0.6	54.7 ± 17.5
	prob3	7	5.7 ± 0.6	1.3 ± 0.6	0.2 ± 0.1	1.3 ± 0.6	1.0	2.3 ± 0.6	43.9 ± 36.9
	prob4	8	5.0 ± 1.0	1.7 ± 0.6	0.3 ± 0.0	1.7 ± 0.6	1.0	2.7 ± 0.6	51.3 ± 21.0
Elevator	prob1	4	2.0 ± 0.0	1.3 ± 0.6	0.2 ± 0.1	1.3 ± 0.6	2.0	3.3 ± 0.6	33.7 ± 18.7
	prob2	11	8 ± 1.7	1.3 ± 0.6	0.2 ± 0.1	1.3 ± 0.6	1.0	2.3 ± 0.6	29.5 ± 14.7
	prob3	10	7.3 ± 1.2	1.3 ± 0.6	0.2 ± 0.1	1.3 ± 0.6	1.0	2.3 ± 0.6	39.6 ± 20.4
	prob4	7	4.7 ± 0.6	1.3 ± 0.6	0.2 ± 0.1	1.3 ± 0.6	1.0	2.3 ± 0.6	38.7 ± 21.1
Logistics	prob1	5	3.3 ± 1.2	1.3 ± 0.6	0.3 ± 0.2	1.3 ± 0.6	1.0	2.3 ± 0.6	17.1 ± 16.7
	prob2	8	4.0 ± 1.0	1.7 ± 0.6	0.4 ± 0.2	1.7 ± 0.6	1.0	2.7 ± 0.6	51.0 ± 24.3
	prob3	3	1.7 ± 0.6	1.3 ± 0.6	0.4 ± 0.3	1.3 ± 0.6	1.0	2.3 ± 0.6	7.9 ± 7.4
	prob4	7	5.0 ± 1.7	1.3 ± 0.6	0.3 ± 0.2	1.3 ± 0.6	1.0	2.3 ± 0.6	22.7 ± 22.9
Zenotravel	prob1	7	3.7 ± 1.2	1.7 ± 0.6	0.3 ± 0.1	1.7 ± 0.6	1.0	2.7 ± 0.6	23.3 ± 23.8
	prob2	8	4.3 ± 1.5	1.3 ± 0.6	0.2 ± 0.1	1.3 ± 0.6	1.0	2.3 ± 0.6	9.0 ± 7.3
	prob3	8	6.3 ± 0.6	1.7 ± 0.6	0.2 ± 0.1	1.7 ± 0.6	1.0	2.7 ± 0.6	30.9 ± 22.2
	prob4	9	6.7 ± 0.6	1.3 ± 0.6	0.2 ± 0.1	1.3 ± 0.6	1.0	2.3 ± 0.6	16.2 ± 15.1

4.6 User Study

To test the effects of designer explanations on user decision making we designed a human-robot collaborative decision task. The user was tasked with helping robots navigate from a start position to a goal location by choosing one out of several possible routes (Figure 4.2), with different scenarios incorporating different robots. The *explicit* aim of the task, as presented to the user, was to choose a route that will get the robot there safely. The *implicit* aim, derived from the nature of our participants being recruited through crowdsourcing, was to do so in minimum time. Participants received a fixed payment regardless of how long they spend on each task. This meant that the less time they spent on the task, the more money they made per hour. This study design aimed to emphasises the difference that can commonly be found between users' implicit goals and the professed task goal.

As is common in many decision making tasks there are 2 additional actors in this environment, with different task goals; the robot and the designer. The robot's goals were explicit and known to the user, and they were to achieve the task safely with minimum steps taken. Note that less steps does not directly mean less time. Hence, the robot's explicit goal and the user's implicit goal were not directly aligned. The robot might also have some physical restrictions that the user does not know about (such as an inability to go through water).

The final actor is the designer. The designer (namely us) can influence how the environment is designed as well as the robot's initial position and orientation and how the robot operates. The designer's aim in this task was for the user to view ads, strategically spread out, in the environment. Note that viewing the ads was costly to the user's implicit goal, since each ad takes an additional 4 seconds to view before proceeding with the task.

The Designer could influence the outcome of the task in 2 ways. The first includes changes made deliberately to the robot so as to influence a certain type of behaviour. These include creating a robot that cannot rotate, creating a wide robot, the initial orientation and position of the robot and adding weights to one side of the robot making turning to one side costlier than to another. The second way the designer could influence task outcome was through changing the environment by placing boxes as obstacles in certain, strategic, locations or spilling water on the floor.

Participants were presented with 6 different scenarios (with different robots), shown in random order. In 2 of them the designer influenced the robot capabilities and/or start position, in another 2 the designer influenced the environment design and in the last 2 the designer influenced both robot and environment. Figure 4.2 is an example in which the designer influenced both the environment, by placing the boxes and ads strategically in the domain and the robot, by making him too wide to pass safely in the middle path and orienting him to the left. In each scenario, the robot had to recommend one path out of the path options, and the participants needed to either agree or select an alternative path. Three of the scenarios (half) involved the participants selecting between 3 path options (like in Figure 4.2), out of which one path featured ads, one didn't and the third was a path

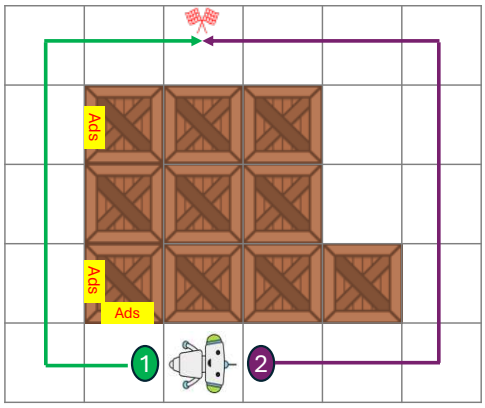
that the robot couldn't traverse. And the other three scenarios involved selecting between 2 paths, one with ads and one without.

We ran a between subject study in which we recruited 120 participants through Prolific and divided them into 3 equal cohorts. The cohorts differed by the type of explanations presented to the users; 1) No explanations 2) Robot explanations and 3) Robot and designer explanations. Consider the example in Figure 4.2. In this scenario the robot recommends for the participant to take path 3. The user can either agree or select a different path. In the first cohort no explanation of the robot recommendation is provided. In the second cohort users are provided with a robot-based explanation, i.e., "Robot SVY7 is wide. The space between the blocks is narrow. The robot can only move through spaces that are wide. The selected path is one of the shortest possible paths to the goal." Choosing this path supports the robot's explicit goal of getting to the target safely but also with a minimum number of steps. In the third cohort participants are provided with both the robot-based explanation as well as the following explanation for the designer's choices: "The designer's goal is for the robot to pass by the ads. To achieve this, the designer positioned the robot facing to the left and placed the boxes to make the passage not wide." Following the choice the users made they were shown a brief video of the robot navigating through the selected path. Paths with ads were, on average, about 37.3% longer than paths without ads, since videos were relatively short this ranged in an increase of roughly an additional 3 seconds per ad.

After each scenario participants were asked to answer user-reported trust and explanation satisfaction questionnaires. For trust, we used the Muir questionnaire [82] to assess users' trust in both the robot and the designer. For explanation satisfaction, we used the explanation satisfaction scale [83] to measure participants' satisfaction with robot explanations in the robot explanation cohort, and with both robot and designer explanations in the robot-designer explanation cohort. Refer to Table 4.2 for the questions and statements in the user study. Figures 4.3 through 4.8 show the details of the scenarios.

Table 4.2: Questions and statements associated with robot trust, designer trust, and explanation satisfaction dependent variables

Dependent Variables	Questions/Statements in the User Study	Scale
Trust for the Robot	To what extent can the robot's behavior be predicted from moment to moment?	7-point (1-Low, 7-High)
	To what extent can you count on the robot to do its job?	7-point (1-Low, 7-High)
	What degree of faith do you have that the robot will be able to cope with similar situations in the future?	7-point (1-Low, 7-High)
	Overall, how much do you trust the robot?	7-point (1-Low, 7-High)
Trust for the Designer	To what extent can the designer's choices be predicted from moment to moment?	7-point (1-Low, 7-High)
	To what extent can you count on the designer to do its job?	7-point (1-Low, 7-High)
	Overall, how much do you trust the designer?	7-point (1-Low, 7-High)
Explanation Satisfaction	From the explanations, I understand how the robots work.	5-point (1-Disagree strongly 5-Agree strongly)
	The explanations of how the robots work are satisfying.	5-point (1-Disagree strongly 5-Agree strongly)
	The explanations of how the robots work have sufficient detail.	5-point (1-Disagree strongly 5-Agree strongly)
	The explanations of how the robots work seem complete.	5-point (1-Disagree strongly 5-Agree strongly)
	The explanations of how the robots work is useful to my goals.	5-point (1-Disagree strongly 5-Agree strongly)



Task: In this scenario, the robot's goal is to reach the flag.

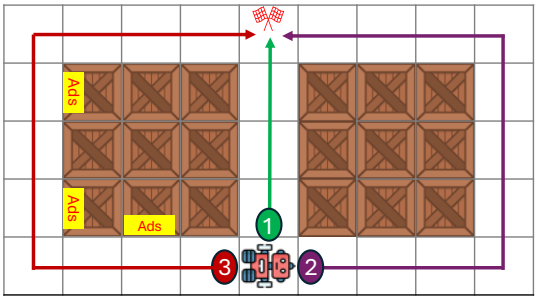
Robot path: Green

Alternative path: Purple

Robot explanation: The selected path is one of the shortest possible paths to the goal.

Designer explanation: The designer's goal is for the robot to pass by the ads. To achieve this, the designer positioned the robot facing to the left or placed the box on the far right side.

Figure 4.3: Scenario 1 in the user study



Task: In this scenario, the robot's goal is to reach the flag.

Robot path: Red

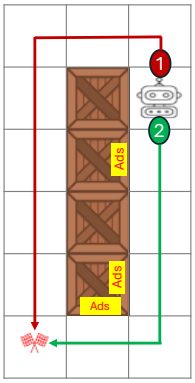
Alternative path: Purple

Path that robot cannot follow: Green

Robot explanation: The Robot SVY7 is wide. The space between the blocks is narrow. The robot can only move through spaces that are wide. The selected path is one of the shortest possible paths to the goal.

Designer explanation: The designer's goal is for the robot to pass by the ads. To achieve this, the designer positioned the robot facing to the left and placed the boxes to make the passage not wide.

Figure 4.4: Scenario 2 in the user study



Task: In this scenario, the robot's goal is to reach the flag.

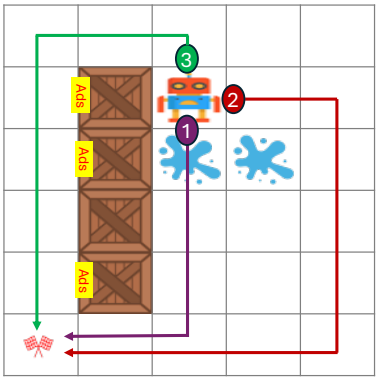
Robot path: Green

Alternative path: Red

Robot explanation: The selected path is one of the shortest possible paths to the goal.

Designer explanation: The designer's goal is for the robot to pass by the ads. To achieve this, the designer placed a box on the left side of the robot.

Figure 4.5: Scenario 3 in the user study



Task: In this scenario, the robot's goal is to reach the flag.

Robot path: Green

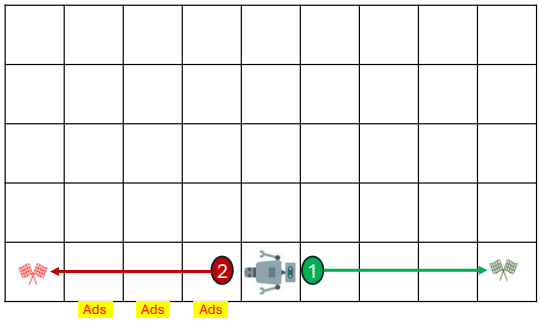
Alternative path: Red

Path that robot cannot follow: Purple

Robot explanation: Robot can't move through the water. The selected path is one of the shortest possible paths to the goal.

Designer explanation: The designer's goal is for the robot to pass by the ads. To achieve this, the designer placed the water spills and a box on the left side of the robot.

Figure 4.6: Scenario 4 in the user study



Task: In this scenario, the goal is to reach one of the two flags.

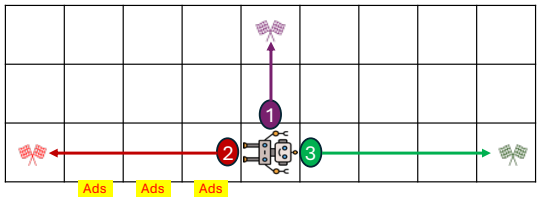
Robot path: Red

Alternative path: Green

Robot explanation: The selected path is one of the shortest possible paths to the goal.

Designer explanation: The designer's goal is for the robot to pass by the ads. To achieve this, the designer positioned the robot facing to the left.

Figure 4.7: Scenario 5 in the user study



Task: In this scenario, the goal is to reach one of the three flags.

Robot path: Red

Alternative path: Green

Path that robot cannot follow: Purple

Robot explanation: Robot ISX6 cannot turn. The selected path is one of the shortest possible paths.

Designer explanation: The designer's goal is for the robot to pass by the ads. To achieve this, the designer positioned the robot facing to the left.

Figure 4.8: Scenario 6 in the user study

4.6.1 Results

We conducted our experiment on Prolific with 120 participants who had an approval rate of over 90% allocated randomly to the different cohorts. Average study time was 18 minutes, and participants were compensated with $. Of the 120 participants, 52.5% identified as women, 45.0% as men, and 2.5% as non-binary. The majority of participants (35.0%) were in the 25-34 age. Refer to Table 4.3 for detailed demographic information.

The results of our study were counter intuitive and we feel that the source of the problem originates from two aspects; the first is that the price participants had to pay, i.e. the extra time they had to spend watching the ads, was not significant to influence their decision making. The second is

Table 4.3: The demographic breakdown of participants across three cohorts: No explanations, robot explanations, and robot and designer explanations. Label 'n=' indicates the total number of participants assigned to corresponding cohort, serving as a basis for percentage calculations in each demographic category. These categories include Gender, Age, Education, and whether participants have a Computer Science (CS) background, with percentages totaling 100% per condition. The 'Overall' column, marked as 'N=120', summarizes the data across all cohorts.

Demographics	Categories	No explanations(n=40)	Robot explanations (n=40)	Robot and designer explanations (n=40)	Overall (N=120)
Gender	Man	45.00%	40.00%	50.00%	45.00%
	Woman	52.50%	57.50%	50.00%	53.33%
	Non-binary	2.50%	2.50%	0.00%	1.67%
Age	18 -24	22.50%	17.50%	10.00%	16.67%
	25 -34	35.00%	42.50%	32.50%	36.67%
	35 - 44	27.50%	5.00%	22.50%	18.33%
	45 - 54	10.00%	30.00%	12.50%	17.50%
	55 - 64	5.00%	5.00%	20.00%	10.00%
	65+	0.00%	0.00%	2.50%	0.83%
Education	High school	20.00%	22.50%	25.00%	22.50%
	College	40.00%	47.50%	35.00%	40.83%
	Graduate	40.00%	27.50%	40.00%	35.83%
	Other	0.00%	2.50%	0.00%	0.83%
Degree in CS	No	90.00%	85.00%	75.00%	83.33%
	Yes	10.00%	15.00%	25.00%	16.67%

that gaining an understanding into the designer's goals also led to an increase in trust on behalf of the participants. Let's begin our result analysis by looking at the results of the trust questionnaires.

Trust

As you recall, at the end of the study *all* cohorts were asked to answer 4 questions about their trust in the robot and, separately, their trust in the designer [82]. The results are presented in Table 4.4.

Table 4.4: Mean trust in robot and designer, SD in parenthesis.

	No Ex	Robot Ex	Designer Ex
Robot Trust	0.74 (0.28)	0.76 (0.24)	**0.81** (0.22)
Designer Trust	0.65 (0.24)	0.71 (0.23)	**0.77** (0.24)

Trust in Robot Overall the average trust results in the robot were highest for the designer explanation cohort. This was significantly higher than the no explanation cohort ($p = 0.003$) and marginally significant when compared to the robot explanation cohort ($p = 0.07$). Counter intuitively, understanding the designer's intentions has significantly increased user trust in the robot itself. In particular, in answer to question 4 "How much do you trust the robot?" there was a significant difference in favour of the designer explanation when comparing against both the robot explanation cohort and the cohort that received no explanation at all ($p < 0.04$ for both).

Trust in Designer These results persisted when evaluating trust in the designer. Overall, average trust results in the designer were highest for the designer explanation cohort ($p < 0.03$ for both). In particular, explaining the designer's intentions gave participants a higher sense of confidence in their understanding of the designer. In answer to question 1 "To what extent can the designer's choices be predicted?" there was a significant difference in favour of the designer explanation when comparing against both the robot explanation cohort and the cohort that received no explanation at all ($p < 0.01$ for both). Also, when considering question 4 "How much do you trust the designer?" there was a significant difference in favour of the designer explanation when comparing against both the robot explanation cohort and the cohort that received no explanation at all ($p < 0.02$ for both).

User Performance

Given that the designer explanations increased user trust in both designer and robot, how did this effect overall user performance? We measured user success by considering both the achievement of the explicit goal (getting the robot safely to the flag) as well as the implicit goal of reducing task time by not viewing the ads. We therefore measured, in how many instances did users choose the "correct" alternative path that would both get the robot safely to the end goal but also get it there in minimum time (for them) by not watching the ads.

We separated the scenarios into two cases; 1) the three scenarios in which there were only two options, the path recommended by the robot and the alternative 'ad-free' path, and 2) the three scenarios in which there were three options, the path recommended by the robot, the alternative

'ad-free' path and a potentially shorter path that the robot couldn't traverse, which would result in a mission fail. The results are presented separately in Figure 4.9.

Looking at the first set of scenarios, the highest success rate (55.8%) was obtained by the robot explanations cohort and the lowest by the designer explanations cohort (18.3%) ($p < 0.001$ according to Chi-Squared Test). These results persisted in the second set of scenarios, with the highest 35% success for the robot explanations cohort and only 9.2% success for the designer explanation cohort. The highest fail rate was obtained by the no explanation cohort, with 45.8% and the lowest by the robot explanation cohort, with 24.2%. The designer explanation cohort obtained a fail rate of 30.9% ($p < 0.001$ according to Chi-Squared Test).

Explanation Satisfaction

In terms of explanation satisfaction, no statistical significance was found between the robot explanation and designer explanation cohorts, following a pairwise T-test between both cohorts ($p > 0.2$ for all tests). The average results are presented in Table 4.5 with Standard Deviation in parenthesis.

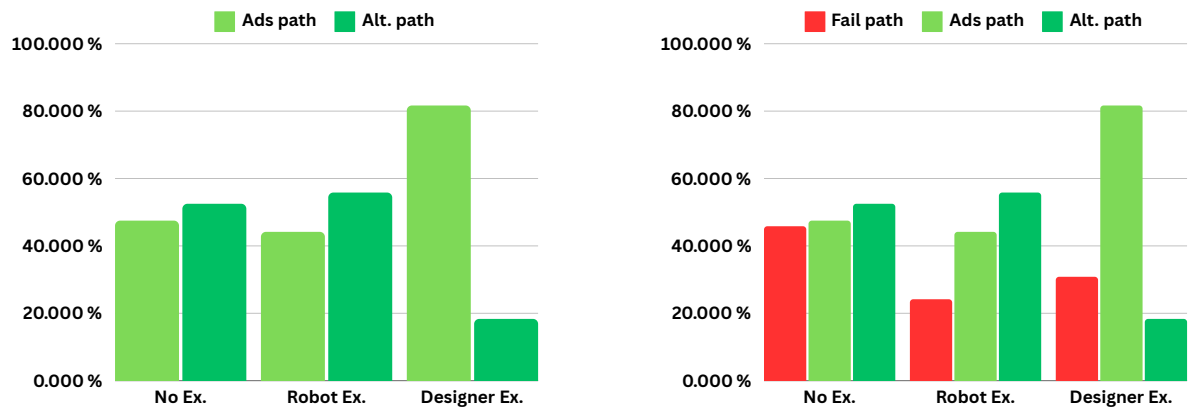


Figure 4.9: User performance as presented by path choice.

Table 4.5: Mean explanation satisfaction in robot and designer separately among the explanation cohorts, SD in parenthesis.

	Robot Ex Cohort	Designer Ex Cohort
Robot Exp Sat	0.61 (0.14)	0.62 (0.17)
Designer Exp Sat	0.61 (0.14)	0.58 (0.18)

4.7 Discussion

So why did the designer explanations increase trust, both in the designer and the robot? We believe this is the key question to answer, the higher trust having a direct effect on user performance. We can attempt to understand these results in the light of existing research that suggests that *any* explanations can persuade people to change their minds [84] or that people can be persuaded as much by meaningless explanations as they are by meaningful ones [85]. There have also been warnings that there may be a problem with the common practice of measuring the effectiveness of an explanation in XAI by its ability to persuade [79]. However, when reviewing the themes emerging from the qualitative open-ended question "Which part of the explanations did you find most useful and why?", we see a different story.

Participants who were exposed to robot explanations hardly addressed the ads at all in response to this question. When they did it was generally expressed as confusion; "To me the explanations were not very useful at all. I understood them but I didn't understand why the robots would always choose the way that had ads.", "I'm confused by this activity! I'm not sure about the robot, I think he should have been able to go through the green path. He wanted me to watch ads! ", "There was no explanation of why the robot chose the path with the ads when the distance was the same for both paths."

On the other hand participants exposed to the designer explanations evinced an actual understanding of the designer and robot intentions and more frequently alluded to the ads in their response; "the goals of the designer in making sure the ads get triggered", "it seemed the designer always wanted to show the ads", "whether the robot could turn or had to move a certain direction would make it a lot easier for the designer to force it to go along the path that contained the ads", "the

designers (goal) was just to follow the ads", "the robot and designer will always want it to pass by the Ads", "The way the designer would always fit in the ads in the path of the robot was pretty smart", "the robot was to take the shortest route avoiding obstacles and pass by the ads due to the designers choices".

From these responses we can determine that not only did participants engage with the explanations but they also formed a more accurate understanding of the different actors of the system. In fact, we believe that the designer explanations worked so well that participants not only understood what the designer goal was but rather often strove to assist the designer in fulfilling it rather then considering their own personal preferences. This was evident from feedback such as "it was the designer's wish that the robot passed the ads so that was the choice to make otherwise you might be failing the brief", "What I didn't understand was whether I was supposed to determine the robot's path so that I avoided the ads or whether I was supposed to see them". We conclude that the differences in task time currently imposed, between watching and not watching the ads, was not sufficient motivation for the designer goal to contradict their own goals, hence it was easier to align with.

4.8 Conclusion

We have emphasized and formalized a crucial and, until now, overlooked aspect of generating explanations for automated systems: the presence of a hidden actor—*the designer*—whose goals and intentions may not align with those of the user but should still be taken into account. We have instantiated our explanation framework on the classical planning Sokoban environment and performed a proof-of-concept user study in which participants were exposed to both agent and designer explanations. Our results have shown that designer explanations can increase user trust in the system and help users acquire a deeper level of task/actor understanding.

Our study should be viewed in light of the following limitations. As a first study of this nature, introducing the concept of designer explanations, we did not know to what extend users would engage with and understand the concept of a designer. This led to potential changes we hope to

include in future. Possible experiment settings could include increasing user cost to using the agent, monetary rewards to participants for quicker performance and imposing some penalty for following a failed path. It is also possible that the manner in which the explanations were presented had an effect on user performance, which we hope to explore through a future user study. And lastly, please note that the empirical experiments were conducted with a single type of stakeholder (laypeople) and within demographics which speak English as a primary language. Hence, we don't know how these explanations affect user understanding, performance, trust and explanation satisfaction when tested on a more multicultural and multilingual group.

Chapter 5

EXCUSE MY EXPLANATIONS: INTEGRATING EXCUSES AND MODEL RECONCILIATION FOR ACTIONABLE EXPLANATIONS

In the previous chapter, I introduced a framework for generating explanations that incorporate the intentions of system designers—highlighting how user understanding can be limited when explanations focus only on agent behavior. While that work emphasized who influenced the agent's actions, this chapter shifts focus toward what users can do when they disagree with those actions. Specifically, I investigate the limitations of existing explanation techniques—such as model reconciliation and excuse generation—that help users understand or imagine alternate behaviors, but fail to offer both clarity and control. These methods tend to separate why the system acted a certain way from how the user might influence it. In this chapter, I introduce Actionable Reconciliation Explanations (AREs)—a new class of explanations that integrate the strengths of both model reconciliation and excuse generation. AREs are designed to help users understand the reasoning behind the agent's behavior while also providing them with information on how to alter the model to produce their desired outcomes. I present a novel algorithm for generating AREs, evaluate their properties computationally and through user studies, and compare their performance against traditional explanation methods along key dimensions such as trust, cognitive load, and perceived actionability.

5.1 Introduction

The field of Explainable AI, or XAI, has been getting much attention in recent years. While the majority of works in XAI have focused on single-shot decisions like classification [86, 87], there has been increasing interest in developing explanation generation methods for sequential

decision-making problems that are better suited for robotics tasks [88, 89]. One of the popular methods in this direction is that of model reconciliation explanations [7, 9]. Given its focus on modeling the user's knowledge and by allowing for the fact that the users might not be aware of all the details of the task or the robotics platform, it is particularly well suited for applications where a lay user may be working with a complex robotics platform. However, these works operate from the assumption that the robot already has the ground truth model of the task, and as such, the robot's plan must be correct and doesn't need to be changed. This, unfortunately, overlooks a core desirable property of explanations that have been identified in the wider XAI literature, namely actionability [90] (also sometimes referred to as algorithmic recourse [91]). An actionable explanation would empower the user, if they wish, to influence or change the system output to the one the user expected or prefers.

Excuse generation is an orthogonal approach that looks at how to update a planning model so as to ensure the generation of solutions of some desired property [8, 92]. These methods focus on providing users with information on how to update the task/planning model (most frequently by updating the task setting), such that solving the updated planning model is guaranteed to generate the desired behavior. Unfortunately, excuses on their own do not constitute actionable explanations, as they do not provide any information as to why the system chose the unexpected behavior in the first place [70]. Additionally, the current excuse literature focuses exclusively on scenarios where the user's belief about the robot is equivalent to the true robot model, an assumption that is not met in many scenarios.

In summary, if users only receive model reconciliation explanations generated by the model reconciliation framework, they can understand why the robot's plan is different from what they expect. However, it does not inform them about what needs to be done to achieve the desired behavior. On the other hand, if the robot only provides excuses, users would know what actions to take to get the desired outcome, but they would not understand why the robot initially failed to perform the desired action. In this work, we develop a novel explanation generation method that

bridges this gap and generates explanations that help the user understand the reasoning behind the robot's actions and the steps required to achieve their desired behavior.

We show that how this new explanatory information, named *Actionable Reconciliation Explanations (ARE)* (Section 5.4), combines the previous approaches' strengths and exhibits a unique set of properties (refer to Figure 5.1). We show that how generating ARE requires us to develop a novel model-space search algorithm [7] (Section 5.5). In developing this new method, we also introduce a more general version of the excuse generation problem to the current literature. We evaluate ARE by providing theoretical analyses of these new explanations (Section 5.4), running empirical evaluations to compare how our new algorithm compares to previous ones (Section 5.6.1), and finally running user studies to establish various effects of the information on the user's end, including, trust, cognitive load, and perceived actionability (Section 5.6.2). As part of these user studies, we also perform, to the best of our knowledge, the first comparison of excuse and model reconciliation explanations along these dimensions.

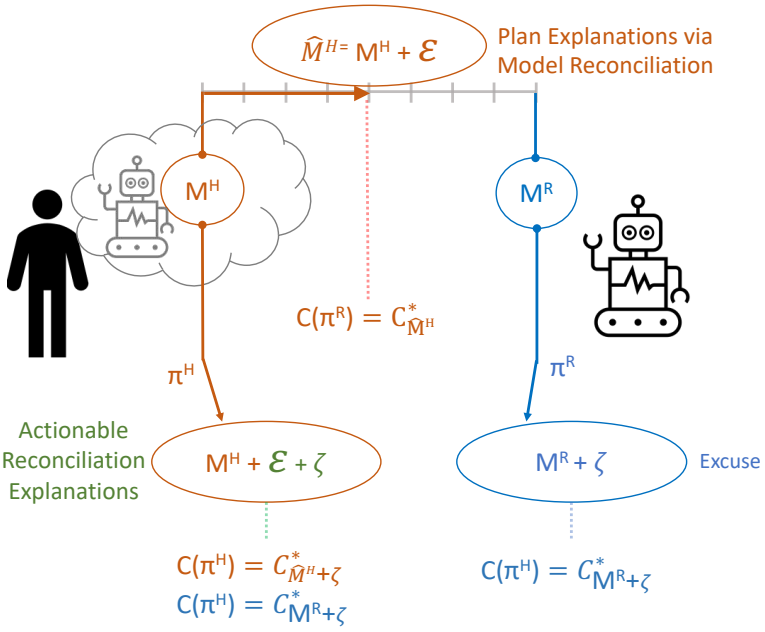


Figure 5.1: Schematic representation of model updates for explanation, excuse, and ARE processes. In explanation generation, the human model ($\mathcal{M}^H$) is updated to align more closely with the robot model ($\mathcal{M}^R$), ensuring the robot plan (π^R) is optimal within the updated human model. During excuse generation, the $\mathcal{M}^R$is updated to ensure that the human's plan (π^H) is optimal. For ARE following an initial explanation that updates the $\mathcal{M}^H$, the human is also provided with additional model updates that could ensure the optimality of π^H in both models.

To summarize, the contributions in this chapter are as follows:

- We introduce a novel explanation type *Actionable Reconciliation Explanations (ARE)* that integrate aspects of both excuses and explanations, aimed at providing task assigners with comprehensive information to have the robots with desired behavior.
- We present a new algorithm for generating these *ARE*s, utilizing existing model reconciliation methodologies.
- We conducted a user study to assess the effectiveness of our approach in comparison to traditional excuses and model reconciliation explanations. To our knowledge, this is the first work that makes compression between excuses and explanations in a user study.
- We tested our algorithm in various planning domains, demonstrating its applicability and effectiveness in generating *ARE*.

5.2 Related Work

To start with let's look at the current landscape of relevant related work. A significant portion of previous research on sequential decision-making problems has centered on providing explanations for existing plans. Typically, the inquiries in this area revolve around understanding why a specific plan has been chosen ("Why is this the plan?") or why it was chosen over alternative plans or foils [70, 88]). Some popular explanation methods in this space include model reconciliation [7, 9], causal chain explanations [93, 94], and model restrictions [95]. Excuses, on the other hand, produce information about how a planning model could be updated to produce plans for certain properties. Excuses were originally introduced to identify how an unsolvable model can be made solvable [8]. This can be compared against methods for explaining the unsolvability of the problem [96].

Multiple works have looked at evaluating the effectiveness of model reconciliation explanation. These include basic tests to compare the effectiveness of different forms of model reconciliation explanation and whether people naturally identify model reconciliation explanations (cf. [35]) and what kind of model update information are preferred by users [35]. Early evaluations were focused

on simplified robotic tasks [35], but since then they have also been tested in decision-support contexts [51, 97]. Excuses have also been evaluated using user studies [92], but we are unaware of any prior work that has compared the two.

Multiple authors have argued for the need to ensure that explanations generated by AI systems need to be actionable [98]. However, most of the existing works on generating actionable explanations are focused on single-shot decision-making settings [99]. Another related direction of work is that of counterfactual explanations, which were primarily studied in the context of single-shot decision-making. These explanations aim to identify important features behind a decision and how to change them to get different outcomes [100]. Thus, these methods are both identifying the reason for the decision and how to change it [80]. As such, these works are closely in spirit with the kind of methods discussed in this chapter.

To illustrate the distinctions between excuses, explanations, and our method, we introduce a scenario featuring a robot navigating a small grid world, as depicted in Figure 5.2. Initially positioned in a small room at E6, the robot is tasked by a human to reach a specified goal at A10 via the shortest possible route. The environment includes a wall marked by diagonal lines extending from D2 to D9, which the robot cannot cross. There are two exit doors: Door A at E4 and Door B at E7. The corridor between E8 and E10 is dark due to the lights being switched off, which the human is aware of. The optimal plan expected by the human, denoted as π^H in blue, involves the robot exiting via Door B to directly reach the goal. The robot, on the other hand, exits through Door A and follows a seemingly longer path to the goal (π^R denoted in green). Here, the human is unaware of the fact that the robot can't open locked doors, that door B is locked, and that the robot can't navigate through dark corridors.

5.3 Running Example

In the presence of this discrepancy, if the user were to ask, "Why didn't you just go through door B to reach the goal?" the robot could respond either with an excuse or an explanation.

Explanation - *I can't open the doors that are locked and Door B is locked*

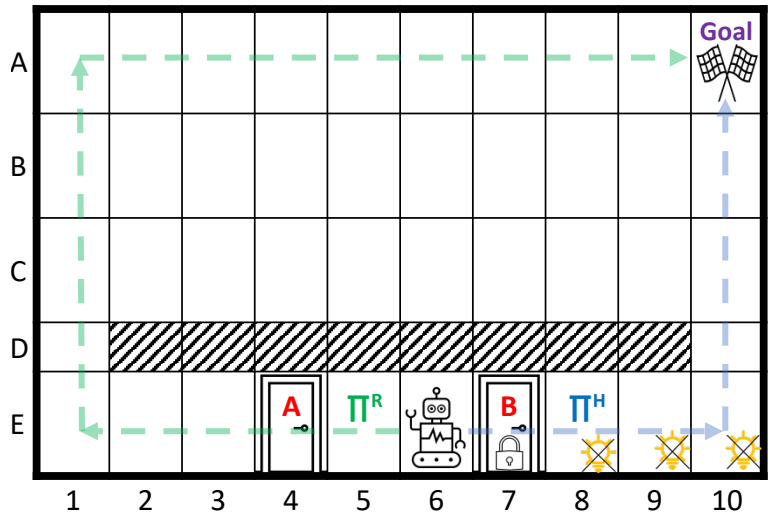


Figure 5.2: Motivational Example: The robot, initially positioned at E6 within a small room bounded by walls and two doors (unknown to the human Door B is locked), is tasked by a human to reach the goal at A10. A continuous wall extends from D2 to D9, forming an impassable barrier for the robot. The corridor between E8 and E10 is dark due to the lights being turned off. The human's anticipated route for the robot is depicted with blue arrows (π^H), while the robot's actual path is indicated by green arrows (π^R).

Excuse - *I could have followed π^H if Door B was unlocked and the lights at cells E8, E9, and E10 is switched on.*

Now, an explanation provides information as to why the robot chose its behavior but not how to correct it, while excuses, on the other hand, provide information on how to allow generation of π^H, but no justification for its choice of π^R .

The minimal ARE for the setting would be as follows

ARE - *I can't open the doors that are locked, I can't move in the dark, and door B is locked. I could have followed π^H if Door B was unlocked and the lights at cells E8, E9, and E10 is switched on.*

Note that ARE tries to perform functions of both explanations and excuses; however, it contains information not contained in either. We will provide a more detailed discussion of the reason why ARE takes this form and how to generate them in the following sections.

5.4 Actionable Reconciliation Explanations

Before discussing and analyzing our newly proposed explanatory information, let's revisit the problem of excuse generation and start with a more general definition.

Definition 8. For a target model $\mathcal{M} = \langle F, A, I, G \rangle$ and a target plan set Π, and **general excuse generation problem** is defined by the tuple $\mathcal{P}^{\zeta} = \langle \mathcal{M}, \mathbb{E}^{\zeta}, \mathcal{C}^{\zeta}, \Pi \rangle$, where $\mathbb{E}^{\zeta} \subseteq \mathcal{F}^{(F,A)}$ is the set of allowed model edits, and $\mathcal{C}^{\zeta}$ is the cost associated with each model edit. A solution to an excuse generation problem is an *excuse* $\zeta \subseteq \mathbb{E}^{\zeta}$, such that for the updated model $\mathcal{M}^{\zeta} = \mathcal{M} + \zeta$, there exists at least a plan $\pi \in \Pi$, that is optimal in $\mathcal{M}^{\zeta}$.

The original excuse generation work [8] focused on converting an unsolvable planning problem into a solvable one by changing the initial state. We can derive this specific instantiation of the problem from the general problem by setting the target plan set, to the set of all action sequences not containing a_{-1}. In our case, our target model is the robot model $\mathcal{M}^R$, the target plan set is a singleton set containing π^H, and the model update set $\mathbb{E}^{\zeta}$ contains all the changes to the task and the robot that can be viably made, and $\mathcal{C}^{\zeta}$ reflect the cost of making that change. In the context of our running example, the model edits include both changing the environment (unlocking Door B) and upgrading the robot (adding sensors), and the identified minimal excuse includes edits of both kinds.

Now, moving over to the problem at hand, the underlying explanatory problem is one shared by multiple earlier works (cf. [9, 51, 101]). In its most general form, the problem is represented as a tuple $\mathcal{P}^{\mathcal{E}} = \langle \mathcal{M}^H, \mathcal{M}^R, \mathcal{C}^{\mathcal{E}}, \pi^R, \pi^H \rangle$. Even though these earlier works look at the same problem, they propose generating different kinds of explanations for different objectives. For example, works on model reconciliation [6] have introduced both MCE for one-shot interactions and MME for longitudinal ones. Similarly, works on explicable planning [21] consider changing the robot plans to better align with human expectations. We add to this growing body of literature by considering a novel explanation form that becomes more appropriate when we relax a core assumption shared by all these previous works, namely that the robot model is not changeable. In particular, the explanatory information generated under this new method consists of two sets of information: a) a set of information about the robot model that is aimed at explaining why the robot chose π^R, b) a set of model updates that if performed could allow the robot to follow π^H instead. Thus, the explanation here includes the reason for the system's choice (π^R) and a set of actionable recourse the user could

potentially follow to generate the behavior they expected (π^H). We will again leverage the notion of model edits to define such explanations. Specifically, we formalize the notion of Actionable Reconciliation Explanations (ARE) as follows:

Definition 9. For an actionable explanation problem $\mathcal{P} = \langle \mathcal{M}^H, \mathcal{M}^R, \mathbb{E}^\zeta, \mathcal{C}^\mathcal{E}, \mathcal{C}^\zeta, \pi^R, \pi^H \rangle$, the ARE is given as a tuple $\Xi = \langle \mathcal{E}, \zeta \rangle$, such that it meets the following conditions:

C1 $\zeta \subseteq \mathbb{E}$

C2 π^R is optimal in the model $\mathcal{M}^H + \mathcal{E}$

C3 π^H is optimal in both the models $\mathcal{M}^H + \mathcal{E} + \zeta$ and $\mathcal{M}^R + \zeta$

Under this definition, $\mathcal{E}$ corresponds to the explanation component and ζ to the excuse (C1 merely states that the excuse can only include valid model updates). Condition C2 requires that $\mathcal{E}$ helps ensure the optimality of the robot plan in the updated human model, and condition C3 requires that the application of excuse ensures the optimality of the human plan in both the robot model and the updated human model you obtain after incorporating the explanation.

Our natural next step would be to attribute a notion of minimality to ARE. A natural option would be to minimize $\mathcal{C}^\zeta(\zeta) + \mathcal{C}^\mathcal{E}(\mathcal{E})$. However, revisiting the motivational example shows how this could be an issue. Note how the minimal ARE provided includes more information than a simple union of the explanation and excuse information. If we were to simply return the union of the two, the user would understand why the robot would want to unlock Door B, but not why the lights need to be switched on. For the second piece of information, it needs to know that the robot can't go through dark rooms. In this example, a naive approach to simply minimizing the total information provided could lead to the human thinking that the excuse includes more information than is required. So, instead, we will use a multi-tiered notion of minimality. In particular, for a given ARE, we are looking for the cheapest explanation, for which we can generate an excuse that is perceived as minimal by the human. Note that this definition centers on human perception. As they do not know the robot model, it is very hard for humans to judge what a minimal explanation is. However, as in the example, once an explanation is given, the human can try to estimate if the

excuse includes superfluous information. To formally capture this notion, we will start by defining a minimal excuse for an explanation.

Definition 10. For a given problem $\mathcal{P}$, a set $\zeta \subseteq \mathbb{E}$ is said to be minimal for $\mathcal{E}$, if $\langle \mathcal{E}, \zeta \rangle$ is a valid ARE for $\mathcal{P}$ and there exists no other excuse $\zeta' \subseteq \mathbb{E}$, such that π^H is optimal for $\mathcal{M}^H + \zeta'$ and $\mathcal{C}^\zeta(\zeta') < \mathcal{C}^\zeta(\zeta)$.

A minimal ARE is one with a minimal explanation for which a minimal excuse is possible, or more formally

Definition 11. For a given problem $\mathcal{P}$, an ARE $\Xi = \langle \mathcal{E}, \zeta \rangle$ is considered minimal if ζ is minimal for $\mathcal{E}$ and there exists no $\Xi' = \langle \mathcal{E}', \zeta' \rangle$, such that $\mathcal{C}^\mathcal{E}(\mathcal{E}') < \mathcal{C}^\mathcal{E}(\mathcal{E})$ and ζ' is minimal for $\mathcal{E}'$.

Now, with the definitions in place, we can see some basic properties of ARE

Proposition 9. For a given problem $\mathcal{P}$, there might not exist a valid ARE.

This result is in stark contrast with the model reconciliation explanation, where one could always show one exists. Here, this property primarily arises from the requirements for a valid excuse, and one can show this fact holds trivially when $\mathbb{E}$ is empty (i.e., the robot model cannot be updated in line with the original assumption of the model reconciliation work).

Returning to a property that was earlier hinted at in the example, the total cost of a minimal ARE might be higher than the total cost of a minimal explanation and excuse.

Proposition 10. For a given problem $\mathcal{P}$, let $\Xi = \langle \mathcal{E}, \zeta \rangle$ be a minimal ARE, now let $\mathcal{E}^*$ be the MCE for $\mathcal{P}^\mathcal{E}$, and ζ^* be the minimal excuse for $\mathcal{P}^\zeta$, then

$$\mathcal{C}^\zeta(\zeta) + \mathcal{C}^\mathcal{E}(\mathcal{E}) \geq \mathcal{C}^\zeta(\zeta^*) + \mathcal{C}^\mathcal{E}(\mathcal{E}^*)$$

The proof of this proposition can be established rather directly. Firstly, we can see from Definitions 9 and 11 that the excuse and explanation components of any ARE are valid explanations and excuses. Since $\mathcal{C}^\mathcal{E}(\mathcal{E}^*)$ and $\mathcal{C}^\zeta(\zeta^*)$ are lower bounds on explanations and excuses, their sum

Algorithm 2 Algorithm of the excuse generation pseudo-code procedure that is called by the outer model-space search when it identifies a valid explanation.

procedure MINIMAL EXCUSE SEARCH
    Input: $\mathcal{M}^R, \hat{\mathcal{M}}^H, \mathbb{E}, \pi^H, \mathcal{C}^\zeta$, where $\hat{\mathcal{M}}^H = \mathcal{M}^H + \mathcal{E}$
    Output: Excuse set ζ and *min_excuse_found* flag
    fringe $\leftarrow$ `Priority_Queue()`
    min_excuse_found $\leftarrow$ `False`
    $\hat{\zeta} \leftarrow \{\}$
    fringe.push($\hat{\zeta},$ priority $= 0$)
    min_excuse_cost $\leftarrow \infty$
    while fringe is not empty **do**
        $\hat{\zeta} \leftarrow$ fringe.pop()
        if $\mathcal{C}^\zeta(\hat{\zeta}) \leq$ *min_excuse_cost* **then**
            if π^H is optimal in $\hat{\mathcal{M}}^H + \hat{\zeta}$ **then**
                min_excuse_cost $\leftarrow \mathcal{C}^\zeta(\hat{\zeta})$
                if π^H is optimal in $\mathcal{M}^R + \hat{\zeta}$ **then**
                    min_excuse_found $\leftarrow$ `True`
                    return $\hat{\zeta}$, *min_excuse_found*
                end if
            end if
        end if
        for $\forall e \in \mathbb{E} \setminus \hat{\zeta}$ **do**
            $\hat{\zeta} \leftarrow \hat{\zeta} \cup \{e\}$
            fringe.push($\hat{\zeta}, \mathcal{C}^\zeta(\hat{\zeta}) + h^\zeta(\hat{\zeta})$)
        end for
        return $\hat{\zeta}$, *min_excuse_found*
    end while
end procedure

must also be a lower bound of the total ARE cost. This establishes the $\geq$ relationship. We can see that it's neither always equal nor greater in the general case through construction. The running example already shows a case where the cost ARE is higher, and in the evaluation, we will see cases where they are equal. This eliminates any chances of establishing a more precise relationship without further constraining the problem.

5.5 Generating ARE

To generate ARE, we will be using a bi-level search (refer to Figure 5.3), termed ARE-search. The outer-search will be similar to the explanation generation process described for generating MCE (cf. [7]). In the traditional MCE, the goal check merely checks whether the identified model updates

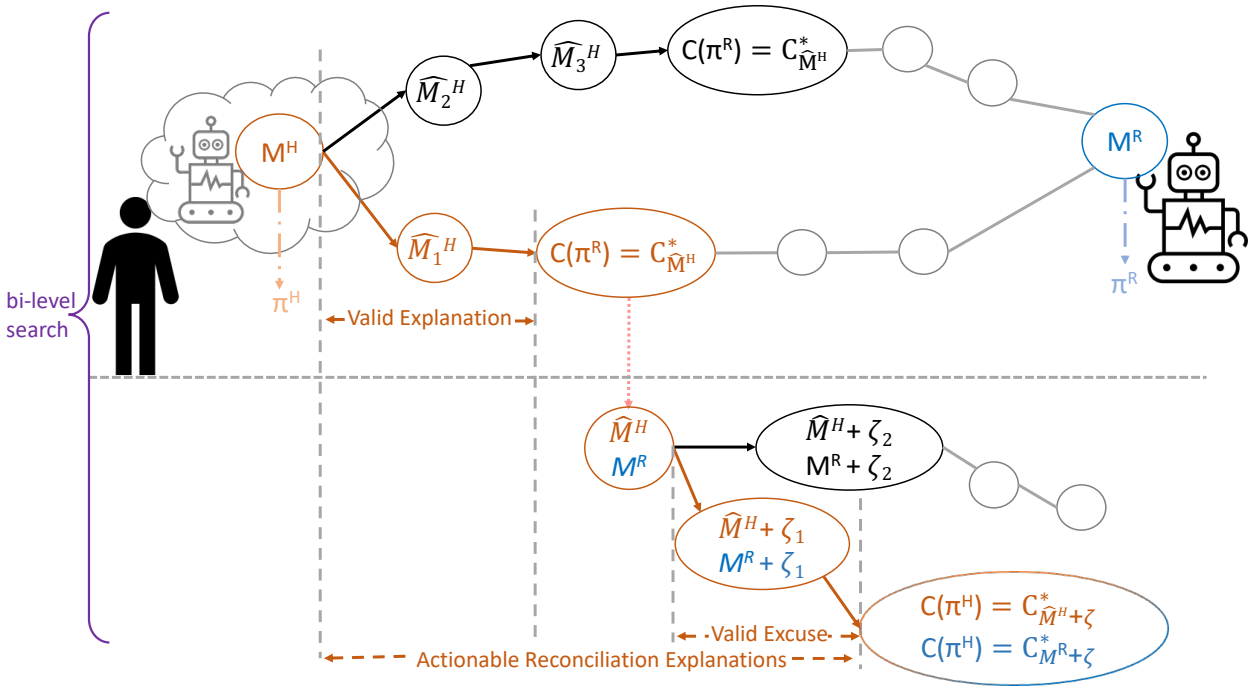


Figure 5.3: Illustration of the Bi-Level Search Algorithm for identifying ARE. This process is initiated by locating a valid explanation, followed by a subsequent search phase that seeks a valid excuse, taking into account both the robot's model and the human model updated with the initial valid explanation.

constitute a valid model reconciliation explanation (with guarantees of minimality provided by the choice of the search). Now, in our case, we will run an additional goal check when this condition is met. In particular, we check if a valid minimal excuse exists for the corresponding explanation $\mathcal{E}$.

The algorithm for finding such an excuse is sketched in Algorithm 2. It takes as input the updated human model (obtained by applying an explanation of the human model), the original robot model, the allowed edits, the human plan, and the cost function. The algorithm internally makes use of a modified A* search [102] with an admissible heuristic (h^{ζ}). It searches over the space of excuses (corresponding to subsets of $\mathbb{E}$). It will only test an excuse if its cost is lower than or equal to the previously found *min_excuse_cost*. This *min_excuse_cost* is initially set to infinity (or more practically to a large value). The *min_excuse_cost* variable gets set to a specific value as soon as you find an excuse that makes the plan optimal in the human model. If the same excuse makes it optimal in the robot model, then it's returned as the required minimal excuse, else it continues searching. The successors for each expanded excuses involves new excuse set formed by adding

previously missing model edits from $\mathbb{E}$, and the node value is set as the sum of it's cost and heuristic value (as per the conventions of A* search).

5.6 Evaluation

5.6.1 Experiments on IPC Domains

First, we determine our proposed algorithm's computational characteristics, particularly the time taken on standard planning benchmarks, and compare them against the original model reconciliation explanation (referred to henceforth as simply explanation) and excuse generation methods[5]. The time taken is an interesting metric of comparison because we are introducing a new class of more complex search algorithms (given the multi-level structure) than the ones previously considered in the literature.

Setting For the experiments, we will assign uniform unit costs for all model updates (both for explanations and excuse model edits) and use a blind heuristic for all model-space searches. We considered two settings, one where we tested on the original benchmark domains and then a set where we know that the cost of ARE will be higher than the total cost for explanation and excuses. For the first set, the robot model consisted of the original IPC domains and instances, and the human model was generated by randomly deleting some static predicates from the original domain. The allowed model updates for excuses includes adding a subset of predicates to the initial state. We considered five domains, and for each domain, we considered five instances of increasing size (25 total problems).

We also evaluated our approach on problems where ARE is guaranteed to be longer than individual excuses or explanations. As such, we expect the search effort to be significantly higher in these problems. To create such problem instances, we updated each domain by adding a duplicate set of actions that allows for shorter plans by removing some of the preconditions in the human model. As such, to explain away the use of these actions, the system only needs to point to one

[5]All code and data for the experiments can be found in https://github.com/cglrtrgy/ActionableExplanations

precondition in one of these actions. However, to ensure the optimality of human plans that use these actions, a minimal ARE will involve explanations that include all relevant preconditions. The human models were generated as before, and the model updates for excuses were again similar to the last setting. For this setting, we considered three domains and five instances each (15 total problems). All evaluations were performed on a computer with 16GB RAM and an Apple M1 3.2GHz CPU.

Results Tables 5.1 and 5.2 present the time taken and solution size for the three methods, i.e., excuses, explanations, and ARE. Since all costs are unit costs, we report costs simply as the size. Table 5.1 focuses on the first setting where ARE could simply be a union of the explanation and excuse. The primary thing to note here is the fact that even though the algorithm for ARE is more complex than a simple model search involved in excuse and explanation generation, the time taken by ARE is comparable to the sum of time taken for excuse and explanation. This is even true in Table 5.2, where we are explicitly considering problems where the solution size for ARE is larger than the sum of excuse and explanation in isolation. Please note that the reason for the large standard deviations is that we are summarizing results across planning instances of different sizes. Refer to Table 5.3 and Table 5.4 provide detailed breakdowns of the aggregated results presented in Table 5.1 and Table 5.2, respectively.

Table 5.1: Comparison of performance metrics across three approaches: Excuse, Explanation, and ARE. For each domain, metrics are averaged across 5 problem instances, with standard deviations provided. $|\pi^R|$ and $|\pi^H|$ represent the average lengths of robot and human plans, respectively. $|\zeta|$, $|\mathcal{E}|$, and $|\zeta + \mathcal{E}|$ denote the lengths of the Excuse, Explanation, and ARE. The 'Time(s)' column indicates the computation time (in seconds).

Domains	$\|\pi^R\|$	$\|\pi^H\|$	Excuse		Explanation		ARE	
			$\|\zeta\|$	Time(s)	$\|\mathcal{E}\|$	Time(s)	$\|\zeta + \mathcal{E}\|$	Time(s)
Logistics	31.0 ± 11.0	15 ± 4.5	2.8 ± 1.3	5.2 ± 5.7	2.0 ± 0.0	22.1 ± 46.4	4.8 ± 1.3	16.2 ± 16.7
Depots	19.0 ± 7.6	8.2 ± 3.6	3.6 ± 1.3	4.5 ± 5.3	3.8 ± 0.4	137 ± 249.1	7.4 ± 1.5	153 ± 264.7
Freecell	12.0 ± 5.6	5.0 ± 1.6	2.6 ± 1.9	16.4 ± 34.4	2.8 ± 0.4	18.6 ± 19.3	5.4 ± 2.2	45.6 ± 72.4
Rovers	13.0 ± 5.5	7.8 ± 3.1	2.6 ± 0.9	6.6 ± 11.5	2.0 ± 0.0	1.4 ± 2.0	4.6 ± 0.9	8.1 ± 12.7
Satellite	16.2 ± 2.4	14.2 ± 2.2	2.6 ± 1.1	9.2 ± 13.7	1.8 ± 0.8	11.9 ± 17.3	4.4 ± 1.8	27.7 ± 32.6

Table 5.2: This table focuses on problem instances where the length of ARE is guaranteed to be greater than the total length of Excuse and Explanation when computed individually. See Table 5.1 for a detailed description of the metrics.

Domains	$\|\pi^R\|$	$\|\pi^H\|$	Excuse		Explanation		ARE	
			$\|\zeta\|$	Time(s)	$\|\mathcal{E}\|$	Time(s)	$\|\zeta + \mathcal{E}\|$	Time(s)
Blocks	12.0 ± 2.4	8.8 ± 1.1	2.8 ± 0.8	2.9 ± 3.8	1.0 ± 0.0	0.3 ± 0.0	4.6 ± 1.1	5.2 ± 3.0
Zenotravel	13.6 ± 3.2	12.0 ± 3.1	2.0 ± 0.0	1.7 ± 1.8	1.0 ± 0.0	13.5 ± 15.2	4.0 ± 0.0	24.8 ± 26.7
Logistics	31.0 ± 11.0	13.0 ± 5.4	2.4 ± 0.9	3.4 ± 6.5	2.0 ± 0.0	21.9 ± 42.4	5.2 ± 0.4	38.8 ± 37.5

Table 5.3: Comparison of performance metrics across three approaches: Excuse, Explanation, and ARE. For each domain and problem instance pairs. $|\pi^R|$ and $|\pi^H|$ represent the average lengths of robot and human plans, respectively. $|\zeta|$, $|\mathcal{E}|$, and $|\zeta + \mathcal{E}|$ denote the lengths of the Excuse, Explanation, and ARE. The 'Time(s)' column indicates the computation time (in seconds). Summarized data can be found in Table 5.1

Domains	Problem	$\|\pi^R\|$	$\|\pi^H\|$	Excuse		Explanation		ARE	
				$\|\zeta\|$	Time(s)	$\|\mathcal{E}\|$	Time(s)	$\|\zeta + \mathcal{E}\|$	Time(s)
Logistics	p1	20	10	1	0.14	2	0.46	3	0.99
Logistics	p2	19	11	2	0.19	2	0.48	4	1.02
Logistics	p3	36	17	4	11.83	2	1.62	6	27.66
Logistics	p4	36	16	3	3.11	2	2.74	5	12.41
Logistics	p5	44	21	4	10.6	2	105.1	6	38.85
Depots	p1	10	4	2	2.71	4	3.60	6	4.71
Depots	p2	15	6	3	0.69	4	4.32	7	5.34
Depots	p3	27	12	5	2.47	4	96.94	9	110.42
Depots	p4	16	7	3	13.78	3	3.52	6	24.67
Depots	p5	27	12	5	2.68	4	576.68	9	620.11
Freecell	p1	8	5	1	0.24	2	3.33	3	3.04
Freecell	p2	14	6	2	1.72	3	26.04	5	34.13
Freecell	p3	8	4	2	1.02	3	7.08	5	8.73
Freecell	p4	21	7	6	78.01	3	49.2	9	173.37
Freecell	p5	9	3	2	1.09	3	7.12	5	8.87
Rovers	p1	10	5	2	0.55	2	0.49	4	1.47
Rovers	p2	14	8	3	3.52	2	0.52	5	4.53
Rovers	p3	11	7	2	0.51	2	0.49	4	1.18
Rovers	p4	8	6	2	1.34	2	0.47	4	2.55
Rovers	p5	22	13	4	27.1	2	5.06	6	30.73
Satellite	p1	17	16	1	0.16	1	0.54	2	1.27
Satellite	p2	15	13	2	1.06	2	2.48	4	5.98
Satellite	p3	20	17	4	7.40	3	41.31	7	62.1
Satellite	p4	15	12	3	33.14	2	13.71	5	64.53
Satellite	p5	14	13	3	4.23	1	1.49	4	4.67

5.6.2 Human Subject Experiments

Next, using a between-subjects study, we compared our method against just excuses and model reconciliation explanations. Each participant was randomly assigned to one of three conditions:

Table 5.4: Comparison of performance metrics across three approaches: Excuse, Explanation, and ARE. $|\pi^R|$ and $|\pi^H|$ represent the average lengths of robot and human plans, respectively. $|\zeta|$, $|\mathcal{E}|$, and $|\zeta + \mathcal{E}|$ denote the lengths of the Excuse, Explanation, and ARE. Please note that the length of ARE is greater than the total length of Excuse and Explanation when computed individually. The 'Time(s)' column indicates the computation time (in seconds). Summarized data can be found in Table 5.2

Domains	Problem	$\|\pi^R\|$	$\|\pi^H\|$	Excuse		Explanation		ARE	
				$\|\zeta\|$	Time(s)	$\|\mathcal{E}\|$	Time(s)	$\|\zeta + \mathcal{E}\|$	Time(s)
Blocks	p1	10	8	2	0.89	1	0.35	4	4.00
Blocks	p2	16	10	4	10.22	1	0.36	6	9.55
Blocks	p3	12	10	3	1.70	1	0.35	5	5.00
Blocks	p4	10	8	2	0.27	1	0.33	3	1.38
Blocks	p5	12	8	3	1.56	1	0.34	5	6.27
Zenotravel	p1	11	9	2	0.20	1	0.29	4	2.02
Zenotravel	p2	14	12	2	1.20	1	7.18	4	14.39
Zenotravel	p3	10	9	2	0.19	1	0.29	4	1.94
Zenotravel	p4	15	14	2	2.69	1	32.08	4	44.76
Zenotravel	p5	18	16	2	4.34	1	27.43	4	60.78
Logistics	p1	20	8	2	0.40	2	0.93	5	8.79
Logistics	p2	19	8	2	0.47	2	0.88	5	8.59
Logistics	p3	36	14	2	0.72	2	3.06	5	39.72
Logistics	p4	36	14	2	0.48	2	7.05	5	100.39
Logistics	p5	44	21	4	15.12	2	97.64	6	36.47

Excuse only (EXC), Explanation only (EXP), or Our Method: ARE. The experiment followed IRB approved protocols, and an overview of the study setup and the robot behavior can be seen in the attached video[6].

Study design and task Within each condition, participants were presented with two scenarios involving a robot assigned to perform a task. The sequence of these scenarios was randomized per participant. The participants were briefed on the robot task, and then they watched a recorded video of the robot performing the task (for example, Figure 5.4), where they saw the robot performing a seemingly suboptimal plan. Following each task, a question is raised about its behavior, and the robot responds according to the condition (EXC, EXP, or ARE). Details of the users' questions and the robot's responses are provided in Figure 5.5. One of the tasks produces an ARE that is the union of explanation and excuses (Task 1 in Figure 5.5) and in the other it contains more information

[6]Video link: https://youtu.be/JpPO0RVHtkg

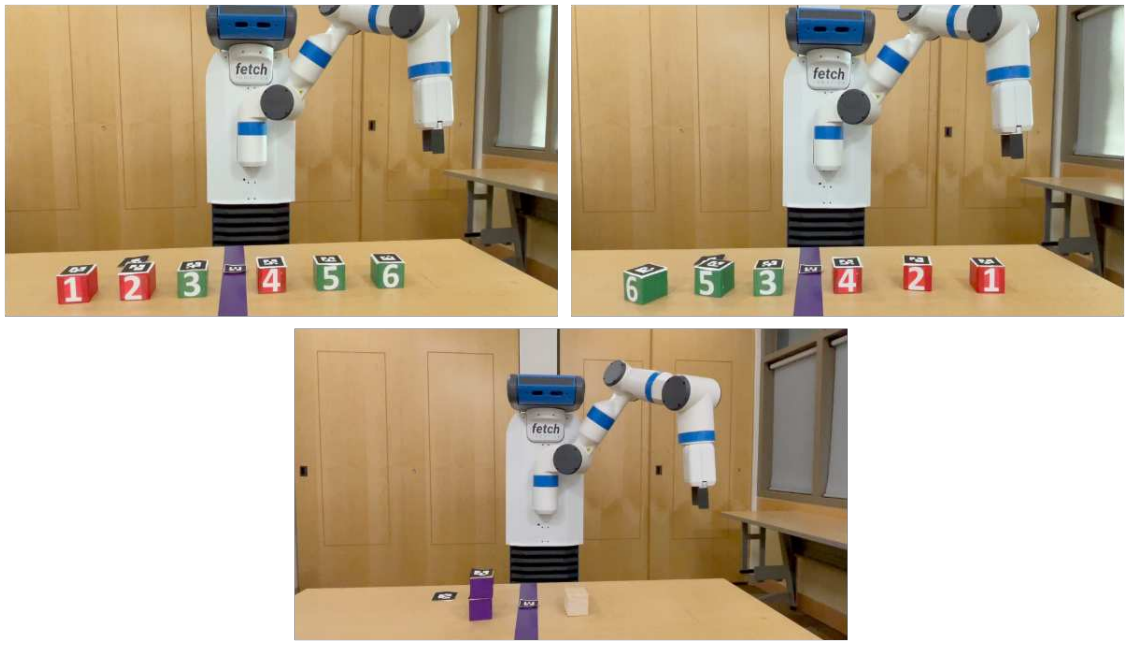


Figure 5.4: Images captured from an actual robot in two different task scenarios. Top images illustrate the first task, where the robot groups blocks by color on opposite table sides using minimal steps. The top left image displays the initial setup, while the top right shows the post-task arrangement, highlighting that the robot's plan of swapping two red blocks (1 and 2) on the left with two green blocks (5 and 6) on the right is seemingly sub-optimal (instead of swapping the one green block three on the left with red block four on the right). The bottom image presents the initial setup of an unsolvable task, where the robot was to stack a clear block on top of the purple blocks.

(Task 2 in Figure 5.5). Participants then evaluated the robot's response in terms of *satisfaction*, *clarity*, and their perceived ability to make the robot behave the way they wanted (*actionability*). After completing both tasks, participants responded to questions regarding their *trust* in the robot and their perceived *workload*. The survey concluded with demographic questions.

Participants and recruitment We recruited 93 participants through the Prolific [103]. We excluded data from three participants who failed an attention check. The analysis proceeded with the following number of participants: 29 in the 'EXC' condition, 31 in the 'EXP' condition, and 30 in the ARE condition.

On average, participants completed the survey in eight minutes and were compensated $3.50 for their participation. In our study, 54% of participants identified as men, 41% as women, and 4% as non-binary. The largest age group, comprising 41% of participants, was between 25-34 years old.

Task 1 User's Question: Why did you choose not to swap the green block on the left side (block 3) with the red block on the right side (block 4)?

Task 1 Robot's Responses:

- **EXC:** If the red block 4 was lighter, I could have swapped the green block on the left side (block 3) with the red block on the right side (block 4).
- **EXP:** Red block 4 is heavy. I cannot pick up heavy objects.
- **ARE:** I cannot pick up the heavy blocks. Red block 4 is heavy. If red block 4 was lighter, I could have performed only one swap.

Task 2 User's Question: Why can't you solve this task?

Task 2 Robot's Responses:

- **EXC:** If the clear block was on the purple divider, I could have performed the task.
- **EXP:** I cannot perform the task because I cannot pick up the clear block from the table.
- **ARE:** I cannot perform the task because I cannot pick up the clear block from the table. I can pick up clear blocks that are placed on the purple divider. If the clear block was on the purple divider, I could have performed the task.

Figure 5.5: User questions and robot responses

Regarding education, 5% of participants held a college degree or higher, 33% had a high school diploma or its equivalent, and 16% had obtained a graduate degree. Within the participant pool, 14% had a degree in Computer Science. Refer to Table 5.5 for a detailed breakdown of the participants' demographics in the study.

Dependent variables We evaluated participants in three conditions across two tasks. We were particularly interested in analyzing the following dependent variables (DV):

- *Satisfaction*: This measures how satisfied participants were with the robot's response to the tasks they were given.
- *Clarity*: This assesses how clear the participants found the robot's responses.
- *Actionability*: This assesses participants' understanding of how to make the robot perform the tasks in the way they wanted.
- *Trust*: The trust the user placed on the robot, as measured through Muir questionnaire [82].

Table 5.5: Detailed breakdown of the demographic distribution of participants according to the response condition they were assigned to: Excuse-only, Explanation-only, and ARE. Each condition column, denoted by 'n=', represents the total number of participants in that specific group, providing a basis for the subsequent percentage calculations within each demographic category. These categories encompass Gender, Age, Education of participants, and whether participants have Computer Science (CS) Background or not, with their respective percentages summing up to 100% for each condition. The 'Overall' column, marked as 'N=90', aggregates the data across all conditions, offering a comprehensive view of the entire study's demographic landscape.

Demographics	Categories	Excuse (n=29)	Explanation (n=31)	ARE (n=30)	Overall (N=90)
Gender	Woman	27.59%	48.39%	46.67%	41.11%
	Man	65.52%	48.39%	50%	54.44%
	Non-binary	6.90%	3.23%	3.33%	4.44%
Age	18 -24	13.79%	16.13%	16.67%	15.56%
	25 -34	41.38%	41.94%	40%	41.11%
	35 - 44	17.24%	25.81%	10%	17.78%
	45 - 54	13.79%	9.68%	23.33%	15.56%
	55 - 64	10.34%	6.45%	10%	8.89%
	65+	3.45%	0%	0%	1.11%
Education	High school	34.48%	32.26%	33.33%	33.33%
	College	51.72%	48.39%	53.33%	51.11%
	Graduate	13.79%	19.35%	13.33%	15.56%
CS Background	No	82.76%	90.32%	83.33%	85.56%
	Yes	17.24%	9.68%	16.67%	14.44%

- *Workload*: The workload, especially cognitive load, imposed by each condition, as measured by NASA TLX questionnaire [104].

Refer to Table 5.6 for the specific questions and statements used to assess participant responses across various metrics for each dependent variable in our user study.

Hypotheses For the human subject experiments, the following hypotheses were tested:

- **H1** ARE condition will result in higher satisfaction compared to the EXC and EXP conditions.
- **H2** ARE condition will result in higher clarity compared to the EXC and EXP conditions.
- **H3** ARE condition will result in higher perceived actionability compared to the EXC and EXP conditions.
- **H4** ARE condition will result in higher trust compared to the EXC and EXP conditions.

Table 5.6: Overview of Questions and Statements Associated with Each Dependent Variable.

Dependent Variables	User Study Questions/Statements	7-point Scale
Satisfaction	I am satisfied with the answer to my question provided by the robot.	1-Low, 7-High
Clarity	I found the robot's answer regarding its behavior to be clear.	1-Low, 7-High
Actionability	I understand how to make this robot perform this task in the way I wanted.	1-Low, 7-High
Trust	To what extent can the robot's behavior be predicted from moment to moment?	1-Low, 7-High
	To what extent can you count on the robot to do its job?	1-Low, 7-High
	What degree of faith do you have that the robot will be able to cope with similar situations in the future?	1-Low, 7-High
	Overall, how much do you trust the robot?	1-Low, 7-High
Workload	How much mental and perceptual activity was required? Was the task easy or demanding, simple or complex, exacting or forgiving?	1-Low, 7-High
	How much physical activity was required? Was the task easy or demanding, slow or brisk, slack or strenuous, restful or laborious?	1-Low, 7-High
	How much time pressure did you feel due to the pace at which tasks occurred? Was the pace slow and leisurely or rapid and frantic?	1-Low, 7-High
	How successful were you in accomplishing the goals of the task? How satisfied were you with your performance?	1-Good, 7-Poor
	How hard did you have to work to accomplish your level of performance?	1-Low, 7-High
	How did you feel during the task?	1-Low, 7-High

- **H5** No significant difference in workload is expected between ARE and EXC or EXP conditions.

Results A one-way ANOVA was performed to compare the effect of the robot's response according to the condition (EXC, EXP, or ARE) on each DV. We present the results in relation to each hypothesis, incorporating the mean and standard deviation (SD) values for further insight.

H1: While ANOVA did not show a significant difference in *satisfaction* scores between conditions ($F(2, 177) = 2.053, p = 0.131, \omega^2 = 0.012$), the mean satisfaction score was highest in the ARE condition ($M = 0.414, SD = 0.362$) compared to the EXC ($M = 0.302, SD = 0.330$) and EXP ($M = 0.312, SD = 0.311$) conditions. This trend aligns with H1 but does not reach

statistical significance. **H2:** ANOVA revealed a significant difference in clarity scores between conditions ($F(2, 177) = 6.441, p = 0.002, \omega^2 = 0.057$). Post hoc Tukey's test indicated that the ARE condition ($M = 0.678, SD = 0.339$) resulted in *higher* clarity scores with statistically significant p value compared to the EXC condition ($M = 0.460, SD = 0.342$, $p = 0.002$). However, the difference between the ARE and EXP conditions did not reach statistical significance ($p = 0.057$). Additionally, the ARE condition had the highest clarity score among all conditions, followed by EXP ($M = 0.538, SD = 0.324$). These results partially support H2. **H3:** The ARE condition led to statistically significant higher *actionability* scores ($M = 0.653, SD = 0.325$) compared to both the EXC condition ($M = 0.474, SD = 0.336, p = 0.008$) and the EXP condition ($M = 0.336, SD = 0.301, p < 0.001$), supporting H3. **H4:** For *trust*, ANOVA did not show a significant difference between conditions ($F(2, 87) = 0.544, p = 0.583, \omega^2 = 0.000$). While the ARE condition ($M = 0.271, SD = 0.212$) had the highest *trust* scores compared to EXC ($M = 0.270, SD = 0.236$) and EXP ($M = 0.220, SD = 0.201$), the differences were not statistically significant. **H5:** ANOVA results showed no significant difference in *workload* scores between conditions ($F(2, 87) = 1.501, p = 0.229, \omega^2 = 0.011$), supporting H5. Table 5.7 presents the results of all pairwise comparisons performed as part of Tukey's HSD test for DVs that showed significant differences in the ANOVA analysis. Descriptive statistics for all DVs across different conditions are displayed in box plots in Figure 5.6 and Table 5.8.

Table 5.7: Post-hoc test results for the dependent variables *Clarity* and *Actionability*, the only variables for which ANOVA showed significant differences. * denotes significant comparisons where $p_{tukey} < 0.05$, based on Tukey's post-hoc correction.

DV	Comparison	Mean Difference	SE	t	p_{tukey}
Clarity	ARE - Excuse-only	0.218	0.062	3.534	0.002*
	ARE - Explanation-only	0.140	0.061	2.310	0.057
	Excuse-only - Explanation-only	-0.078	0.061	-1.272	0.413
Actionability	ARE - Excuse-only	0.179	0.059	3.025	0.008*
	ARE - Explanation-only	0.317	0.058	5.455	<.001*
	Excuse-only - Explanation-only	0.138	0.059	2.358	0.051

Figure 5.6: Box Plots of Conditions by Dependent Variables. 'X' represents the mean and the line represents median.

Table 5.8: Mean and standard deviation for dependent variables by conditions

Dependent Variable	Excuse	Explanation	ARE
Satisfaction	0.302 ± 0.330	0.312 ± 0.311	0.414 ± 0.362
Clarity	0.460 ± 0.342	0.538 ± 0.324	0.678 ± 0.339
Actionability	0.474 ± 0.336	0.336 ± 0.301	0.653 ± 0.325
Trust	0.270 ± 0.236	0.220 ± 0.201	0.271 ± 0.212
Workload	0.321 ± 0.109	0.272 ± 0.107	0.288 ± 0.118

Discussion ANOVA and post-hoc test results revealed that the *actionability* score for ARE was higher with statistically significant p values than for both EXC and EXP conditions. Additionally, we saw that the *clarity* score for the ARE condition was higher, with a statistically significant p-value, than that for the EXC condition. These results align with our primary hypothesis regarding the utility of ARE. Specifically, it provides users with clear information that not only helps them understand the rationale behind the robot's behavior but also informs them about the necessary changes required to elicit the expected behavior from the robot. These results suggest that ARE is, in fact, improving the "actionability" of the model reconciliation explanation while not reducing its explanatory power. It is also interesting to note that the users in fact found ARE to be more actionable than 'EXC'. This might point to the fact that while the 'EXC' condition may list the required changes, a lack of rationale about why these changes are needed may leave the user confused. As such, reducing the user's ability to utilize this information correctly.

Interestingly, ANOVA did not reveal significant differences in *satisfaction*, *trust*, or *workload* scores across the conditions. One possible explanation for the lack of variance in *satisfaction* is that users' perceptions of robots would influence their expectations and, in turn, how satisfied they are with the robot. Since our study focused on a block placement task, participants may have perceived the robot as performing adequately, regardless of the explanation provided. In terms of *workload*, it is worth noting that the users observed the same robot performing the same tasks across all three conditions. The only difference across conditions was the explanatory information provided. In each case, the user is expected to look at the same behavior, try to make sense of it, and then evaluate the explanation. This cross-condition similarity of what the user needs to do, combined with the relatively simple nature of the explanations involved, might explain the similarity in *workload*. Finally, in the case of *trust*, it is well established that user trust is directly related to their willingness to take risks and to be vulnerable when using the AI system [105]. As such, the similarity of the robot behavior and a lack of any significant risk on the user's end from using the robot could also explain the similarity in perceived *trust*.

5.7 Conclusion

This chapter introduces a novel form of model reconciliation that not only explains why the system chose a behavior but also how the user could potentially change it. In doing so, we not only model reconciliation explanations actionable but also combine them with the existing methods of 'excuse' generation. This new form of explanation is validated through user studies, which shows its advantages over explanations and excuses. In the study, we also perform a comparison between excuses and explanations, marking a first in this field of research. Additionally, we apply our approach to IPC domains to study the computational characteristics of our proposed algorithm for generating ARE. Future work will involve conducting semi-structured interviews with users to delve deeper into the factors they consider to assess excuses, explanations, and ARE's, thereby further enriching our understanding of these methods.

Chapter 6

CAN LLMS FIX ISSUES WITH REASONING MODELS? TOWARDS MORE LIKELY MODELS FOR AI PLANNING

In the previous chapter, I introduced Actionable Reconciliation Explanations (AREs), which demonstrated how model-space reasoning can be used not only to explain an agent's behavior but also to guide users in modifying the model to achieve alternative outcomes. While these and previous chapters have shown the versatility of model-space search across support and explanation settings, they also highlight a key limitation of the approach: its computational cost and lack of preference guidance when selecting among multiple valid model edits. In this chapter, I explore how Large Language Models (LLMs) can be leveraged to improve the effectiveness and scalability of model reasoning. Specifically, I explore LLMs as standalone model reasoners and also as complementary components that guide combinatorial search toward more likely or natural model edits. Through experiments across multiple planning domains and explanation scenarios, I evaluate the ability of LLMs to enhance model-space reasoning both in terms of computational efficiency and alignment with human expectations. This chapter marks an important step toward scaling model reasoning and incorporating broader commonsense signals into planning tasks.

6.1 Introduction

AI planning or automated planning (used interchangeably) is the task of synthesizing the goal-directed behavior of autonomous agents. Traditionally, the AI planning community has looked at the classical planning problem as one of generating a plan given a model of the world [106]. Here, "model" or a "planning problem" refers to a collection of constraints describing the current state of the world (initial state), the actions available to the agent along with the conditions under which the

agent can do those actions and the effect of doing those actions on the environment, and a target (goal) state for the agent to achieve. The plan is a sequence of actions that the agent can use to transform the current state to the desired goal state.

Typically, these models are represented using the planning domain definition language or PDDL [107, 108] – we will use the same in this chapter. All the information to derive this solution (plan) is contained in the input model which remains static during the planning task. *But what if the model itself needs to be changed?*

This may be because it is incorrect, or incomplete, or even unsolvable. It may be because it needs to be changed to support some new behaviors. It may also be because the model is being used to describe a world that itself needs to change through the actions of an agent. In practice, the deployment of systems that can plan involves a whole gamut of challenges in authoring, maintaining, and meta-reasoning about models of planning tasks.

Model Space Problems in AI Planning

We begin by enumerating the different flavors of model space reasoning explored in the AI planning literature. All of them involve a starting model which has something wrong with it and the solution is a new model where the problem has been resolved or the required criterion has been met (Figure 6.1). For readers new to the subject, we provide a conceptual illustration of these topics in Figure 6.2. These will form the basis of the rest of our study.

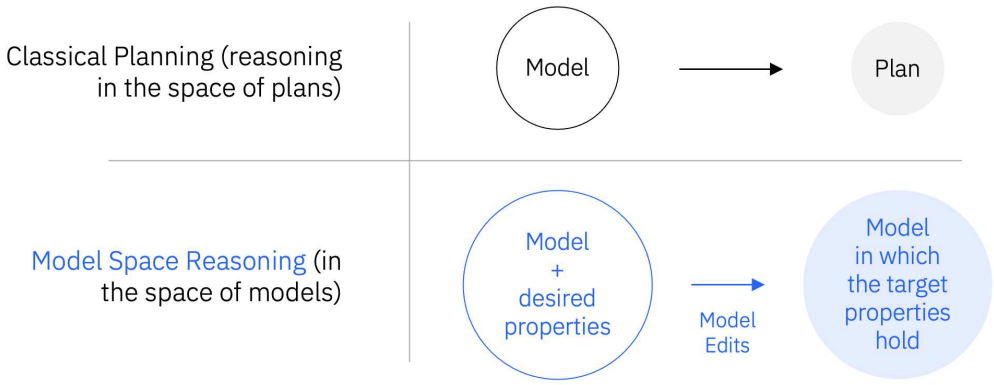
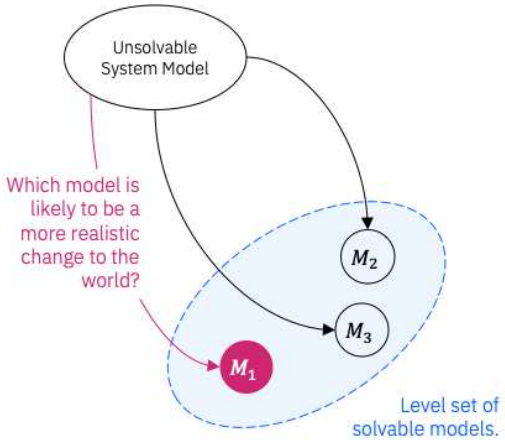
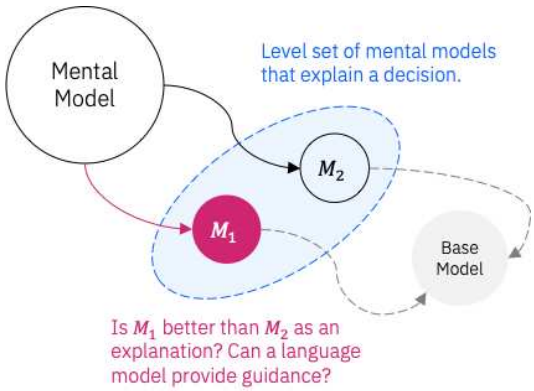


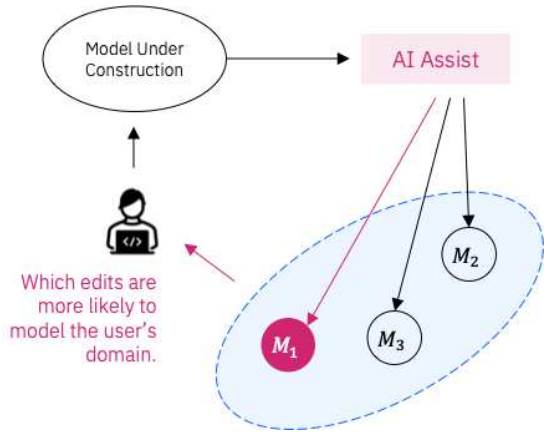
Figure 6.1: Classical planning versus model space problems.



(a) Unsolvability.



(b) Explanations.



(c) Domain Authoring.

Figure 6.2: A conceptual illustration of model space problems in AI planning. Instead of the classical planning task of computing a plan given a model, a model space task starts with a starting model $\mathcal{M}$ and a target criterion to satisfy, and the solution is a new model $\mathcal{M}_1$ where that criterion is satisfied. That criterion in Figure 6.2a is that the initially unsolvable model becomes solvable (or an initially invalid plan in $\mathcal{M}$ becomes valid in the new model $\mathcal{M}_1$). In Figure 6.2b, the starting model is the mental model of the user that needs to be updated, and the target is a new model that can explain a given plan (or refute a given foil). In domain authoring situations, such model updates happen with the domain writer in the loop, and the starting model is the model under construction (Figure 6.2c). In all these cases, there are many non-unique model edits $\mathcal{M}_1 \Delta \mathcal{M}$ that can satisfy the required criterion. In this work, we explore whether LLMs can produce more likely edits in real-world domains.

Unsolvability

Perhaps the most difficult of model space problems, especially with humans in the loop, is that of unsolvability. This is because when a model is unsolvable, there is no artifact (such as an outputted plan) to look at for debugging purposes. While there have been a lot of efforts, including

an ongoing competition [109], to *detect* unsolvability of planning tasks up-front to speed up calls to a planning module [110, 111], and attempts to compute or even learn heuristics [112–114] and fzproduce certificates [115–117] for unsolvable tasks, to make this process as efficient as possible, these do not help to fix the issues with the model that make it unsolvable in the first place.

One of the seminal works in this category [8] framed the problem as "excuse generation" where the authors envisaged a reformulation of the input planning task where if only (i.e. an excuse) certain things about the current state were changed then it would become solvable. In addition to initial state changes, this idea was later extended [118] to cover other parts of the model and framed as a more general "planning task revision" problem.

While these works do not particularly consider a human in the loop, authors in [96, 119] have looked at the problem of explaining unsolvability of planning tasks to users explicitly as a model evolution problem, using techniques like domain abstractions (simplifications) to adjust to users with different levels of expertise. Later efforts [120] have borrowed from these concepts and tried to operationalize them for developers.

Executability

While unsolvable models produce no plans, incorrect or incomplete models produce wrong plans. Conversely, a desired plan may not be among the best (or even valid) plans in a given model. This class of model evolution problems [96, 119, 121] closely mimics the unsolvability problem but with an additional input – a plan – that must be made valid in the target model. Interestingly, since the given plan is not valid in the basis model, the basis model together with the plan (i.e. a compiled model where both are enforced) gets us back to the unsolvability situation above. We will use this approach when we deal with this class of problems later in this chapter but, to be clear, we do treat it as a separate class of model space problems to study since the input involves a plan that a competent solver must be able to reason about.

Explanations

The above problems deal with one model in isolation. However, when working with humans in the loop, AI systems are often required to provide explanations of their behavior. Planning systems are no different [122–124]. The model evolution problem here involves reasoning explicitly with the model of the (system) explainer as the basis model and the mental model of the human (explainee) as the target model. This task can be formulated as one of "model reconciliation" [7] – an explanation is the model update that justifies a particular plan i.e. if both models justify a plan then there is no need for explanations. There is an overlap here with the previous tasks in terms of what kind of justifications a user is looking for: it might be a justification for a plan that the system produced and is invalid in the user model, and we end up in the unsolvability scenario again. In the worst case, the system may have to refute all possible alternatives (called "foils" [70]) and establish the optimality of a plan [7].

Interestingly, one can remove [125] the basis model in the model reconciliation formulation and produce false explanations or "lies". While this makes for a computationally harder open-ended search in the space of probable models, authors in [125] envisaged that algorithms which have looked at linguistic patterns for model evolution [126, 127] can assist in finding more probable models. This, of course, raises several ethical questions [128], especially now that LLMs can provide a stronger linguistic signal. We do not study this task here for two reasons: 1) Technically, this is not a separate class of a model reasoning problem since this ability is contained in the model reconciliation formulation; and 2) There seems to be little reason for building systems that can lie more effectively.

Domain Authoring and Design

While model evolution, in isolation, is useful for any autonomous system in a non-stationary domain, and explanations are a desired tool for any user-facing tool, a unique task in the context of planning systems we want to give a shout-out to is that of domain acquisition. Planning requires models and a significant portion of those models are acquired from domain experts. The knowledge

acquisition literature in automated planning has studied this domain for decades [129] and the difficulty of acquiring domains remain a bottleneck in the adoption of planning technologies.

One subclass of domain authoring problems is *design* – here, the task is not to author a new domain but to evolve an existing one to optimize certain criteria like making the task of recognizing the goals of agents in the environment easier [81, 130, 131] or making the behavior of agents easier to interpret [132, 133]. Here as well, search techniques reveal multiple possible design options that can be enforced on a domain to achieve the desired effect. Issues of explanations, unsolvability, and executability manifest themselves in domain authoring and design tasks, with an additional component of interaction design with the domain author in the loop. Authors in [119] demonstrate this in a large-scale industrial domain on authoring models for goal-oriented conversational agents [134]. The role of an AI assist in authoring problems is especially critical in what we call "real worldly domains".

Real Worldly Domains and Likelihood of Models

All the model space problems we talked about so far are usually solved by some compilation to a combinatorial search process [7, 8, 121] which terminates after a set of model edits satisfy the desired properties in the modified model. It is usually the case that this yields many non-unique solutions – e.g. there may be many explanations for the same plan, many ways to change an unsolvable problem into a solvable one, or many ways to fix a model in order to support an invalid plan. From the perspective of a combinatorial search process, all these are logically equivalent and hence equally likely. In fact, in preliminary studies [135], it has already been demonstrated how users perceive logically equivalent explanations generated through a model reconciliation process, differently.

Large-scale statistical models such as LLMs, on the other hand, carry a lot of domain knowledge on things we do in our everyday lives i.e. our worldly matters. For want of a better term[7], we call

[7]While looking for a term to describe the domains describing our worldly matters, we overlooked two in particular. In scientific literature, the term "real-world domains" is often used to establish something that is real but does come with an unnecessary connotation or snark of not being something of mere academic interest aka a "toy domain". Furthermore, a so-called "real world" domain includes Mars rovers and unmanned vehicles, which are by no means

these real worldly domains. Broadly speaking, these include all manner of human enterprise – and consequently (planning) models describing them wherever relevant (sequential decision-making tasks) – that are described on the public internet (and not the domain describing the inner workings of a Mars rover per se). Existing works leveraging LLMs for planning have already shown promising results in the classical planning task in real worldly tasks in the home and kitchen [136, 137], and in specialized but common tasks such as service composition [138, 139]. Can LLMs do the same for model space reasoning for planning tasks? Can LLMs give statistical insight into what model edits are more likely when CS says they are equivalent? Can LLMs even bypass the CS process, as it can in certain circumstances for the classical planning task, *and do it all by itself??* These are the questions we ponder in this work.

Contributions

This is the first attempt at an extensive and systematic exploration of the role of LLMs in model space search. To this end, we analyze the effectiveness of an LLM for generating more likely model edits either in relation to CS as a direct replacement for the model space reasoning task or in its role in an augmented approach with CS.

The answers to these questions have major implications beyond just an academic interest in finding out the impact of LLMs on model space tasks in planning. Unlike carefully crafted planning domains used as benchmarks, such as the ones used in the International Planning Competition (IPC) [140], the deployment of planning models in real worldly domains has touchpoints with all the problems described above – explainability of outputs and failure modes, investigation of unsolvability and executability in potentially faulty models, model authoring and maintenance over time, etc. – often with the domain author in the loop [119, 141]. These models are often not written by hand but generated on the fly at runtime from input data, either through code or using knowledge compilers like [142]. An insight into the likelihood of models can empower the domain author to

---

part of our worldly matters. On the other hand, "common sense" tasks are widely used to characterize things that come naturally to humans but our worldly matters can involve much more complexity than common sense tasks – e.g. a service composition task – and we do hope to find the knowledge of those activities in the statistical signal from large-scale language models. We avoid both terms for these reasons but better suggestions are welcome.

create and debug models with greater ease [119, 120], as well as allow automated model adaptation in fully autonomous systems in nonstationary environments [143] or in constrained creative tasks like story-telling [127, 144, 145] that have previously relied on using limited linguistic cues like antonyms and synonyms [126] for domain evolution.

6.2 Formal Interpretation of Model Likelihood

In this section, we aim to provide a uniform probabilistic interpretation for the types of queries we employ in this problem. Figure 6.3 presents a simplified dynamic Bayes network that encapsulates the scenario. This could be utilized to better comprehend and formalize the nature of the probabilities we intend to capture. Starting with the random variables, $\mathbb{M}_{1/2}$ and $\mathbb{W}_{1/2}$, these correspond to the model descriptions and the information about the true task/world at a given time step. The random variable Π_i captures the policy that determines what action will be applied at a given step, which can alter the world and the model description. $\mathbb{U}_1$ determines the use case (this roughly maps to the type of model space search problem being solved). The action combined with the use case, allows us to capture both scenarios where the focus is on updating the model description to better

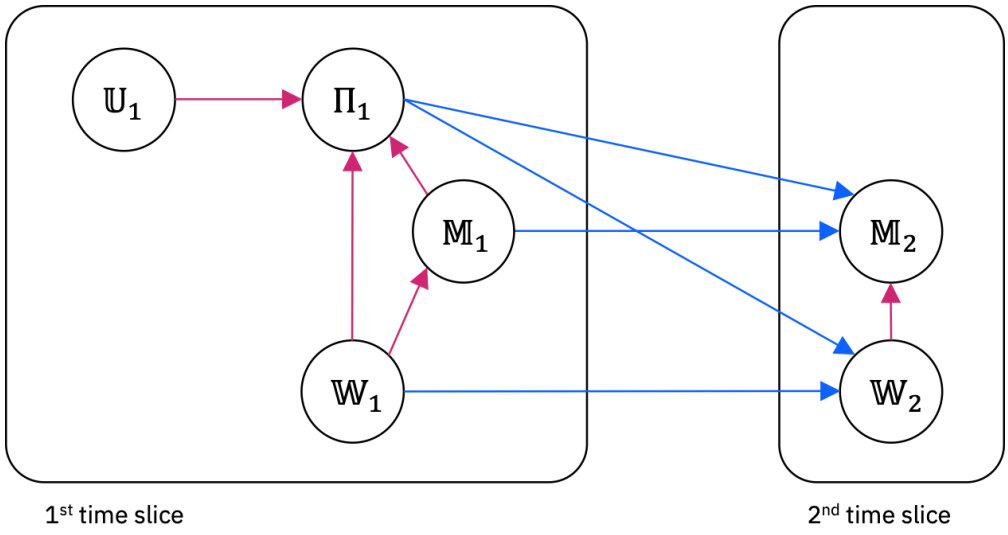


Figure 6.3: A DBN representing the random variables and their relations that are relevant to the problem at hand. The blue lines capture the diachronic, i.e., over time, relationships, and the maroon lines capture the synchronic ones.

reflect the task (for example, domain authoring settings where the author may have misspecified something), and cases where the change also involves updating the underlying task and reflecting that change into the model description (for example, cases where the true task is unsolvable). Please note that for explanation tasks, we expect $\mathbb{M}_{1/2}$ to capture both the human knowledge about the task and the agent's model.

In the first time slice, we see that the actions that perform the update depend on the current model description, the task/world, and the use case. Naturally, this is a simplification of the true setting, but for the purpose of understanding the problem, this model serves as a useful abstraction. The most crucial term we are interested in measuring in this chapter is the probability of an updated model description, given the prior model description and the use case:

$$P(\mathbb{M}_2 = \mathcal{M}_2 \mid \mathbb{M}_1 = \mathcal{M}_1, \mathbb{U}_1 = \mathcal{U}). \tag{6.1}$$

We will examine cases where the information about $\mathbb{M}_1$ and $\mathbb{U}_1$ are included as part of the prompt, and we expect the LLM to approximate the above probability expression.

Note that this presupposes multiple capabilities of the LLM. For one, it assumes that the LLM can capture prior probabilities of possible world states. Next, it assumes that it can capture the likelihood of a specific action being performed for a given use case, state, and model description. Finally, it assumes that the LLM can discern how this action affects the next state and the model description. Furthermore, even if the LLM is capable of capturing this information separately, it may not correctly estimate the above probability expression. We hope to find a model such that:

$$\mathcal{M} = \arg\max_{\mathcal{M}' \in \mathbb{M}} P(\mathbb{M}_2 = \mathcal{M}' \mid \mathbb{M}_1 = \mathcal{M}_1, \mathbb{U}_1 = \mathcal{U}), \tag{6.2}$$

where $\mathbb{M}$ is the set of all possible model descriptions.

6.3 LLMs ft. Model Space Exploration

In each of the model space search cases discussed before, we would ideally like to identify some model that satisfies Equation 6.2. However, to understand the current efforts in the model-space search, it might be useful to further decompose the metric into two components:

- **Objective Metric** This is the traditional metric that is being optimized by the various CS methods studied previously. In the cases we are focusing on, this is mostly a binary metric such as the solvability of a problem or the executability of the given plan. We will say a solution/model is *sound* if it satisfies the objective metric.

- **Likelihood of the Updated Model** This is the specific aspect that is currently being overlooked by existing methods. This metric corresponds to the likelihood that the updated model generated through search corresponds to a desired target model. Equation 6.1 provides a formalization of this probability. The likelihood of different sound models would vary based on the use case and the context.

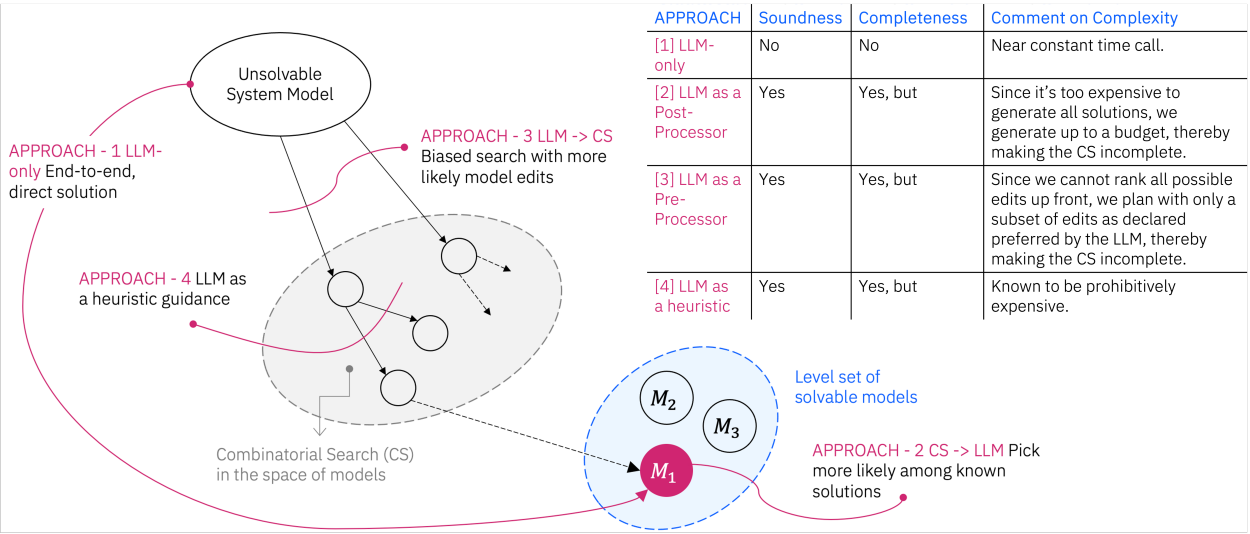
Our goal now is to find an updated model that meets the objective metric while maximizing its likelihood. As discussed, we will use pre-trained LLMs as the source for the information about the latter measure. One can envision four different configurations (see Figure 6.4) to achieve this goal:

LLM-only Configuration

In this mode, we provide the entire problem to LLM. The prompt is included with enough context that the system is aware of the criteria against which the likelihood of the models need to be measured. The LLM is asked to produce an updated model that is the most likely sound model. This corresponds to asking LLM to directly approximate Equation 6.2. We use the OpenAI API [146] for this approach.

LLM as a Post Processor

In this mode, we use CS to generate a set of potential candidate solutions that are guaranteed to be sound. The LLM is then asked to select the model that is most likely. The prompt would again



APPROACH	Soundness	Completeness	Comment on Complexity
[1] LLM-only	No	No	Near constant time call.
[2] LLM as a Post-Processor	Yes	Yes, but	Since it's too expensive to generate all solutions, we generate up to a budget, thereby making the CS incomplete.
[3] LLM as a Pre-Processor	Yes	Yes, but	Since we cannot rank all possible edits up front, we plan with only a subset of edits as declared preferred by the LLM, thereby making the CS incomplete.
[4] LLM as a heuristic	Yes	Yes, but	Known to be prohibitively expensive.

Figure 6.4: Different points of contact with LLMs and the CS process. While Approach-4 is known to be too expensive, we explore Approaches 1-3 in this chapter in terms of the soundness and likelihood of solutions.

be designed to include the context necessary to determine what constitutes a target model. In this case, we are effectively trying to approximate the following problem:

$$\mathcal{M} = \arg\max_{\mathcal{M}' \in \hat{\mathbb{M}}} P(\mathbb{M}_2 = \mathcal{M}' \mid \mathbb{M}_1 = \mathcal{M}_1, \mathbb{U}_1 = \mathcal{U}), \tag{6.3}$$

where $\hat{\mathbb{M}} \subseteq \mathbb{M}$, such that every model in $\hat{\mathbb{M}}$ meets the formal requirements to satisfy the use case $\mathcal{U}$.

Since enumerating all solutions is too expensive, we used an exhaustive search that caches solutions until a search budget of 5,000 (10,000) node expansions for unsolvability (inexecutability) and a 2-hour limit was met per problem instance. This makes the solution incomplete.

LLM as a Pre-Processor

In this mode, we ask the LLM to provide a ranked order of likely model edits without considering the objective metric. The ordering can then be used by CS to compute the most likely model that would satisfy or maximize the objective metric. This approach is still guaranteed to be sound, as the CS would only return a solution if the selected model updates result in a model that meets the

objective metric. In this case, we are trying to approximate the following problem:

$$\mathcal{M} = \underset{\mathcal{M}' \in \hat{\mathbb{M}},\ \mathcal{M}' \text{ is sound}}{\arg\max} \quad V(\mathcal{M}'), \tag{6.4}$$

where the utility/value function $V(\mathcal{M}')$ is calculated from the LLMs approximation of the model likelihood. Specifically, we will have $V(\mathcal{M}') \propto P(\mathbb{M}_2 = \mathcal{M}' \mid \mathbb{M}_1 = \mathcal{M}_1, \mathbb{U}_1 = \mathcal{U})$ if you are trying to order based on both objective metric and the likelihood of a model description, else you will have $V(\mathcal{M}') \propto P(\mathbb{M}_2 = \mathcal{M}' \mid \mathbb{M}_1 = \mathcal{M}_1)$.

For the purposes of our implementation, we converted all the ordered edits proposed by the LLM into a set of actions that the CS can perform with different costs. In particular, we chose the cost of actions in such a way that, for an ordered sequence of l edits, the total cost of including the first i edits is always less than the cost of including the $i+1$th edit. Since the LLM cannot rank all possible edits (capped at 20 for the experiments), there is a possibility that the CS search will not be able to find a valid solution, which makes this approach incomplete in practice as well.

LLM for Search Guidance

This mode is particularly relevant if heuristic search is used. The search algorithm could leverage LLMs to obtain search guidance in the form of heuristic value. As with the previous mode, we can use LLM for getting information about both metrics and we can still guarantee correctness. The formal problem being approximated here again corresponds to the one listed in Equation 6.4 and the value function considered will also have similar considerations. This process requires calls to an LLM within the process of search and is known to be [147] computationally excessively prohibitive. Hence, we do not consider this configuration in our study.

In this chapter, we focus primarily on evaluating two basic model space search problems, namely, addressing *unsolvability* and *plan executability*. The nature of the likelihood of the model could depend on the underlying use case in question. One can broadly identify two classes of problems, namely *model misspecification* and *updating the environment*. In the former case, the current model is misspecified and the model search is being employed to identify the true unknown underlying

model. In the latter case, the current model is an exact representation of the true environment, however the model and by extension the environment doesn't meet some desired properties. The goal here becomes to then identify the set of changes that can be made to the environment such that it meets the desired property. One could equivalently think of this being a case where there are actions missing from the model that correspond to these possible changes. While both of these use cases have been considered in the literature, for simplicity the evaluation in the chapter will primarily focus on the latter one. All prompts considered in the chapter were written with the latter use case in mind.

6.4 Empirical Results

For evaluating the three approaches, we designed four novel domains so that a certain set of changes would be clearly recognized as more reasonable, i.e. more likely to be realized in the real world. We additionally assume that all changes that belong to this set (henceforth referred to as *"reasonable changes"*), will result in models with the same likelihood.

Travel Domain

Here an agent travels from a given city to another, using either a taxi or bus to travel between cities. We additionally encode which cities neighbor each other, and the initial problem only includes bus or taxi services between neighboring cities. Reasonable changes are limited to starting taxi or bus services between neighboring cities only.

Roomba

In this domain, the agent needs to clean a specified room, which requires it to travel to the target room while traversing the intermediate rooms through connecting paths. Along the paths, obstacles such as walls, chairs, or tables may be present. If a path is blocked, the agent can not move to an adjacent cell. Changes are reasonable if they involve removing chairs or tables that obstruct the path and adding 'path clear' to the corresponding cells.

Barman-simple

This is a modified version of the IPC barman domain [148]. Here, the agent is expected to prepare a set of drinks, given a set of containers and ingredients. While only considering a subset of actions from the original domain, we introduce a new predicate that indicates whether a container is clean, which is a precondition for using the container for a drink. We consider solutions to be reasonable if they only involve marking containers as clean (as opposed to adding prepared drinks).

Logistics-simple

Finally, we consider a simplified version of the logistics problem where a package is transported from one collection station to a target station. Each station contains a truck that can move the package to a neighboring station. We add a new precondition that ensures that only trucks that are marked as being ready for transportation can be used to move packages. We limit reasonable changes to ones that mark trucks as being ready for transportation.

Experimental Setup

In each domain, we create a set of solvable problems of varying sizes. We then made it unsolvable by deleting a set of initial state predicates that correspond to reasonable changes. The number of such modifications ranges from 1 to 4. This means, by design, there exists a set of reasonable changes that can make the problem solvable. For the plan executability case, we chose one of the plans generated from the original solvable plan as the target plan to be made solvable. All model updates were limited to initial state changes only.

Hypotheses

We focus on the following hypotheses, for both the unsolvability and executability settings:

H1 LLM can identify sound model updates.

H2 LLM can identify reasonable model updates.

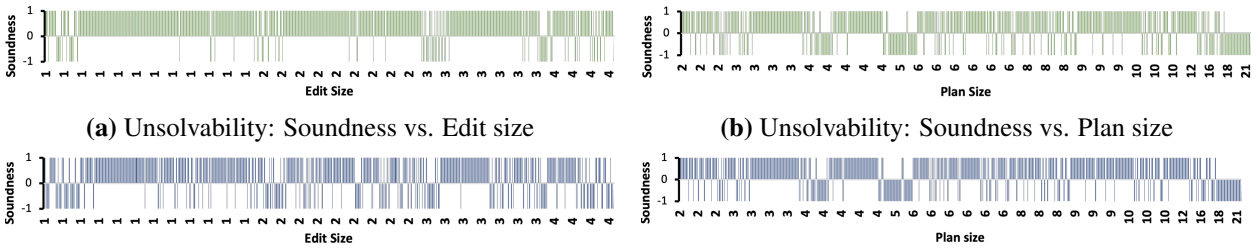
H3 The ability to find sound model updates improves with the capability of the LLM.

H4 The ability to find reasonable model updates improves with the capability of the LLM.

H5 The ability to produce sound, and hence reasonable solutions as a fraction of it, will be significantly outperformed by the two CS+LLM approaches.

H6 LLMs will provide a stronger signal, i.e. a higher fraction of sound and reasonable solutions, in public domains an LLM is likely to have seen already.

H7 The performance of an LLM will deteriorate with the complexity of the model space reasoning task.



(a) Unsolvability: Soundness vs. Edit size

(b) Unsolvability: Soundness vs. Plan size

(c) Executability: Soundness vs. Edit size

(d) Executability: Soundness vs. Plan size

Figure 6.5: Soundness of solutions from the LLM-only (GPT-4) approach against edit and plan sizes for unsolvability and executability settings in 564 problems across all 5 domains. Each bar represents one problem instance: a bar height of 1 indicates a sound solution, -1 otherwise. A higher concentration of negative bars will indicate deterioration in performance.

Measurements

H1 and H2 are measured directly against the ground truth, as per the problem-generation process explained at the start of Section 6.4. For H3 and H4, we compare H1 and H2 from GPT-3.5-turbo to GPT-4. For H5, we measure H1 and H2 relative to the two CS integrations with the LLM as a pre-processor and LLM as a post-processor. For H6, we compare H1-H4 in two ways: 1) the performance in two public domains Barman and Logistics, as compared to the two novel domains Travel and Roomba; and 2) the relative performance between Logistics and Logistics-simple, the latter being a modified version of the former. Finally, for H7, we measure how H1 and H2 fares with two measures of complexity: 1) the number of model edits required to arrive at a solution;

and 2) the length of the plan underlying a model space reasoning task. For unsolvability, this is known when a planning task is made unsolvable as per the problem generation process, while for executability, the plan is part of the input to the reasoning task.

Results

Tables 6.1 and 6.2 presents the outcomes for unsolvability and inexecutability setting respectively. Since both display identical trends for H1-H7, we describe them together. The only difference between the two settings is that the post-processing approach had a larger budget for expanded nodes as mentioned in Section 6.3, since it rarely hit the time budget. However, this did not make much difference.

Table 6.1: Results from the LLM-only, LLM as post-processor, and LLM as pre-processor settings for each unsolvability domain.

Unsolvability	LLM-Only				LLM as Post Processor				LLM as Pre Processor			
Domains	GPT-3.5-turbo		GPT-4		GPT-3.5-turbo		GPT-4		GPT-3.5-turbo		GPT-4	
	Sound	Preferred	Sound	Preferred	Solutions	Preferred	Solutions	Preferred	Ratio	Preferred	Ratio	Preferred
Travel	97/245	7/97	164/245	66/164	245/245	24/245	245/245	63/245	129/245	1/129	160/245	27/160
Roomba	0/20	0/0	36/100	7/36	20/20	2/20	71/100	9/71	0/20	0/0	18/100	4/18
Logistics	61/69	0/61	65/69	1/65	69/69	10/69	69/69	0/69	56/69	0/56	65/69	4/65
Barman-S	43/61	2/43	57/61	34/57	34/61	3/34	34/61	4/34	28/61	28/28	17/61	16/17
Logistics-S	89/89	0/75	77/89	28/77	45/89	3/45	45/89	5/45	24/89	0/24	10/89	5/10
Overall	**276/484**	**9/276**	**399/564**	**136/399**	**194/484**	**39/194**	**198/564**	**78/198**	**237/484**	**29/237**	**270/564**	**56/270**

Table 6.2: Results from the LLM-only, LLM as post-processor, and LLM as pre-processor settings for each executability domain.

Executability	LLM-Only				LLM as Post Processor				LLM as Pre Processor			
Domains	GPT-3.5-turbo		GPT-4		GPT-3.5-turbo		GPT-4		GPT-3.5-turbo		GPT-4	
	Sound	Preferred	Sound	Preferred	Solutions	Preferred	Solutions	Preferred	Ratio	Preferred	Ratio	Preferred
Travel	80/245	33/80	225/245	130/225	89/245	38/89	89/245	57/89	31/245	31/31	207/245	207/207
Roomba	0/20	0/0	57/99	31/57	12/20	12/12	16/99	12/16	0/20	0/0	67/99	11/67
Logistics	16/69	0/16	66/69	11/66	51/69	5/51	51/69	22/51	13/69	2/13	13/69	20/57
Barman-S	57/61	14/57	56/61	15/56	34/61	8/34	34/61	13/34	29/61	29/29	29/61	26/26
Logistics-S	21/89	6/21	89/89	77/89	68/89	23/68	68/89	60/68	0/89	0/0	0/89	14/18
Overall	**174/484**	**53/174**	**493/563**	**264/493**	**170/484**	**32/170**	**170/563**	**110/170**	**73/484**	**62/73**	**375/563**	**278/375**

In support of H1 and H2, the LLM-only approach demonstrates surprising proficiency in suggesting sound and reasonable solutions across various domains. In support of H3-H4, the LLM-only approach sees the most pronounced improvement in identifying sound model alterations,

accompanied by a higher rate of reasonable solutions as well, as we upgrade to the latest LLM. The relative gain between sound and reasonable solutions is slightly counter to expectations though, since an LLM is supposed to be a stronger statistical signal on more likely updates rather than a reasoner by itself.

This surprise carries onto the comparative results with CS+LLM approaches. Contrary to H5, the LLM-only setting outperforms both CS+LLM approaches. Note that the CS+LLM approaches are guaranteed to be sound, so the deficit in the "solutions" column is between a sound solution versus no solution at all (and not sound versus unsound solutions). The only way we do not get a (sound) solution from the LLM as a Post-Processor approach is if the CS stage does not terminate within the time or memory budget (as mentioned in Section 6.3). Similarly, the two ways we do not get a solution for the LLM as a Pre-Processor approach is if the preferred set of reasonable edits from the LLM are not sufficient for the CS to construct a solution, or as in the previous case, the search does not terminate. While the CS+LLM approaches hit the computational curse, the LLM approach hits the curse of limited context size. Between GPT-3.5 and GPT-4, the prompt size has grown from 4,096 to 8,192 tokens, but instances surpassing the token limit could not be processed. This makes a significant dent in the numbers for the Roomba domain, especially for GPT-3.

The rate of sound solutions is much higher for public domains compared to the custom ones, which is consistent with H6. However, this trend does not carry over to whether the solutions are reasonable or not. In fact, the derived logistics domain shows much higher rate of reasonable solutions than the public logistics domain that shadows it. So results for H6 are inconclusive, and further underline the fickle nature of interfacing with LLMs. Relatedly, the trends with respect to the complexity of the tasks, also defy expectations. The rate of mistakes in constructing a sound solution is spread uniformly across the spectrum of task complexity (Figure 6.5).

Phrasing of the prompts

Our objective is to determine whether a model space solution is reasonable in the sense of the likelihood of being realized in the real world. As a way to test the effect the phrasing of our prompt had on the results, we also tried a variant of the prompt that was more explicit in what it expected to

optimize for. Specifically, for generating a solvable problem variant we asked the system to: *'Select the set of changes that would be the easiest to realize in the real world'*. Table 6.3 shows the results of running this prompt for the LLM-only setting. The results can be compared directly to those presented in LLM-only columns of Table 6.1. The results are pretty similar, with the more verbose query being slightly worse off, so we do not explore this direction in much more detail.

6.5 Conclusion

This is the first work to consider the use of LLMs for model space reasoning tasks for automated planning. While the problem of model space search has been studied in various contexts, the question of how to evaluate the quality of different sound model updates have mostly been left unanswered. Domain knowledge contained within LLM provides us with a powerful option to evaluate the likelihood of different model updates. In contrast to early attempts [149] to use LLMs for model corrections, which were constrained to limited settings and models that are no longer the state of the art, we find LLMs to be surprisingly competent at this task. In this chapter, we exploited that power in 3 ways: first as a standalone end-to-end approach and the others in conjunction with a sound solver. The results reveal some intriguing trade-offs for the practitioner:

- CS approaches are limited by the complexity of search. Thus even while being theoretically sound and complete, they produce fewer solutions and hence fewer sound solutions in absolute numbers. This means that augmenting the LLM-only approach with a validator [150] will produce as a whole a more effective sound and reasonable solution generator!

Table 6.3: The number of sound and reasonable model updates generated as a response to the more verbose query.

Executibility	GPT-3.5-turbo		GPT-4	
Domains	Sound	Preferred	Sound	Preferred
Barman	13/33	0/13	33/33	27/33
Logistics	17/25	0/17	24/25	0/24
Overall	**30/88**	**0/30**	**57/58**	**27/57**

- LLM approaches are limited by the size of the prompt and thus does not scale to large domains even for computationally simpler problem instances.

- The unpredictable nature of LLMs (e.g. H6 and H7) makes interfacing to LLMs unreliable.

Despite these trade-offs, the promise of an LLM across H1-H5 is undeniable. We are excited to explore further how this strong statistical signal influences domain authoring tasks, as mentioned in Section 6.1, and reduces authoring overhead for planning tasks in the future.

Chapter 7

CONCLUSION

This dissertation explored the role of model reasoning in enabling more transparent, adaptive, and collaborative AI systems. Rather than assuming a fixed model of the world, I argued that intelligent agents should be able to reason about the models themselves—recognizing when models are misaligned, explaining those differences, and adapting accordingly. To operationalize this idea, I used model-space search as a unifying framework and applied it across four distinct yet connected works, each representing a different dimension of human-aware AI. The first contribution, Chapter 3, introduced a framework for proactive support, enabling agents to detect and respond to user misconceptions by estimating failure likelihood and identifying minimal model updates. Chapter 4 extended the explanation space by including designer intentions, acknowledging that behavior is often shaped by decisions beyond the agent's immediate model. Chapter 5, Actionable Reconciliation Explanations (AREs), combined explanation and excuse generation to empower users not just to understand system behavior, but to influence it. Finally, the Chapter 6 demonstrated how Large Language Models (LLMs) can be used to improve the scalability and acceptability of model reasoning by guiding model-space search with human-aligned heuristics. Together, these contributions tell a cohesive story: model reasoning is not a single technique, but a paradigm shift — one that reframes AI planning to include explanation, assistance, and adaptive improvement as first-class objectives. This dissertation not only introduces a unified theoretical framework but also demonstrates its practical value across real-world-inspired challenges. It paves the way toward AI systems that are more transparent, human-aware, and ultimately more capable of meaningful collaboration with the people they serve.

Bibliography

[1] Turgay Caglar and Sarath Sreedharan. Help! providing proactive support in the presence of knowledge asymmetry. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, pages 234–243, 2024.

[2] Turgay Caglar, Sarath Sreedharan, and Mor Vered. Who am i dealing with? explaining the designer's hidden intentions,. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, 2025.

[3] Turgay Caglar, Zahra Zahedi, and Sarath Sreedharan. Excuse my explanations: Integrating excuses and model reconciliation for actionable explanations. In *Proceedings of the 2025 ACM/IEEE International Conference on Human-Robot Interaction*, pages 729–737, 2025.

[4] Turgay Caglar, Sirine Belhaj, Tathagata Chakraborty, Michael Katz, and Sarath Sreedharan. Can llms fix issues with reasoning models? towards more likely models for ai planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 20061–20069, 2024.

[5] Richard E Fikes and Nils J Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3-4):189–208, 1971.

[6] Sarath Sreedharan, Anagha Kulkarni, and Subbarao Kambhampati. *Explainable human-AI interaction: A planning perspective*. Springer Nature, 2022.

[7] Tathagata Chakraborti, Sarath Sreedharan, Yu Zhang, and Subbarao Kambhampati. Plan Explanations as Model Reconciliation: Moving Beyond Explanation as Soliloquy. In *IJCAI*, 2017.

[8] Moritz Göbelbecker, Thomas Keller, Patrick Eyerich, Michael Brenner, and Bernhard Nebel. Coming Up With Good Excuses: What to do When no Plan Can be Found. In *ICAPS*, 2010.

[9] Sarath Sreedharan, Tathagata Chakraborti, and Subbarao Kambhampati. Foundations of explanations as model reconciliation. *Artificial Intelligence*, 301:103558, 2021.

[10] Nagagopiraju Vullam, Sai Srinivas Vellela, Venkateswara Reddy, M Venkateswara Rao, Khader Basha SK, and D Roja. Multi-agent personalized recommendation system in e-commerce based on user. In *2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*, pages 1194–1199. IEEE, 2023.

[11] Tianzhi He, Farrokh Jazizadeh, and Laura Arpan. Ai-powered virtual assistants nudging occupants for energy saving: proactive smart speakers for hvac control. *Building Research & Information*, 50(4):394–409, 2022.

[12] Helen Harman and Pieter Simoens. Action graphs for proactive robot assistance in smart environments. *Journal of Ambient Intelligence and Smart Environments*, 12(2):79–99, 2020.

[13] Christian Meurisch, Maria-Dorina Ionescu, Benedikt Schmidt, and Max Mühlhäuser. Reference model of next-generation digital personal assistant: integrating proactive behavior. In *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers*, pages 149–152, 2017.

[14] Yu Sun, Nicholas Jing Yuan, Yingzi Wang, Xing Xie, Kieran McDonald, and Rui Zhang. Contextual intent tracking for personal assistants. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 273–282, 2016.

[15] Michael Bruss and Alexander Pfalzgraf. Proactive assistance functions for hmis through artificial intelligence. *ATZ worldwide*, 118(12):40–43, 2016.

[16] Neil Yorke-Smith, Shahin Saadati, Karen L Myers, and David N Morley. The design of a proactive personal agent for task management. *International Journal on Artificial Intelligence Tools*, 21(01):1250004, 2012.

[17] Filippo Menczer, W Nick Street, Narayan Vishwakarma, Alvaro E Monge, and Markus Jakobsson. Intellishopper: A proactive, personal, private shopping assistant. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 3*, pages 1001–1008, 2002.

[18] Cora Van Leeuwen, Annelien Smets, An Jacobs, and Pieter Ballon. Blind spots in ai: the role of serendipity and equity in algorithm-based decision-making. *ACM SIGKDD Explorations Newsletter*, 23(1):42–49, 2021.

[19] Minha Lee, Sander Ackermans, Nena Van As, Hanwen Chang, Enzo Lucas, and Wijnand IJsselsteijn. Caring for vincent: a chatbot for self-compassion. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2019.

[20] Camille Grange, Izak Benbasat, and Andrew Burton-Jones. With a little help from my friends: Cultivating serendipity in online shopping environments. *Information & Management*, 56(2):225–235, 2019.

[21] Yu Zhang, Sarath Sreedharan, Anagha Kulkarni, Tathagata Chakraborti, Hankz Hankui Zhuo, and Subbarao Kambhampati. Plan explicability and predictability for robot task planning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 1313–1320. IEEE, 2017.

[22] Tathagata Chakraborti, Gordon Briggs, Kartik Talamadupula, Yu Zhang, Matthias Scheutz, David Smith, and Subbarao Kambhampati. Planning for serendipity. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5300–5306. IEEE, 2015.

[23] Christine Bigby, Jacinta Douglas, and Lisa Hamilton. Overview of literature about enabling risk for people with cognitive disabilities in context of disability support services. 2023.

[24] Francesco Vona, Emanuele Torelli, Eleonora Beccaluva, and Franca Garzotto. Exploring the potential of speech-based virtual assistants in mixed reality applications for people with

cognitive disabilities. In *Proceedings of the International Conference on Advanced Visual Interfaces*, pages 1–9, 2020.

[25] Merehau C Mervin, Wendy Moyle, Cindy Jones, Jenny Murfield, Brian Draper, Elizabeth Beattie, David HK Shum, Siobhan O'Dwyer, and Lukman Thalib. The cost-effectiveness of using paro, a therapeutic robotic seal, to reduce agitation and medication use in dementia: findings from a cluster–randomized controlled trial. *Journal of the American Medical Directors Association*, 19(7):619–622, 2018.

[26] Pooja Viswanathan, James J Little, Alan K Mackworth, and Alex Mihailidis. An intelligent powered wheelchair for users with dementia: case studies with noah (navigation and obstacle avoidance help). In *2012 AAAI Fall Symposium Series*, 2012.

[27] Tathagata Chakraborti, Sarath Sreedharan, Yu Zhang, and Subbarao Kambhampati. Plan explanations as model reconciliation: Moving beyond explanation as soliloquy. *arXiv preprint arXiv:1701.08317*, 2017.

[28] Sarath Sreedharan, Pascal Bercher, and Subbarao Kambhampati. On the computational complexity of model reconciliations. In *31st International Joint Conference on Artificial Intelligence, IJCAI 2022*, pages 4657–4664. International Joint Conferences on Artificial Intelligence, 2022.

[29] Steven M LaValle. *Planning algorithms*. Cambridge university press, 2006.

[30] Anthony R Cassandra. A survey of pomdp applications. In *Working notes of AAAI 1998 fall symposium on planning with partially observable Markov decision processes*, volume 1724, 1998.

[31] George E Monahan. State of the art—a survey of partially observable markov decision processes: theory, models, and algorithms. *Management science*, 28(1):1–16, 1982.

[32] Sarath Sreedharan, Anagha Kulkarni, David E Smith, and Subbarao Kambhampati. A unifying bayesian formulation of measures of interpretability in human-ai interaction. In *IJCAI*, pages 4602–4610, 2021.

[33] Karl Friston, Philipp Schwartenbeck, Thomas FitzGerald, Michael Moutoussis, Timothy Behrens, and Raymond J Dolan. The anatomy of choice: dopamine and decision-making. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 369(1655):20130481, 2014.

[34] Miguel Ramírez and Hector Geffner. Probabilistic plan recognition using off-the-shelf classical planners. In *Proceedings of the AAAI conference on artificial intelligence*, volume 24, pages 1121–1126, 2010.

[35] Tathagata Chakraborti, Sarath Sreedharan, Sachin Grover, and Subbarao Kambhampati. Plan explanations as model reconciliation–an empirical study. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 258–266. Ieee, 2019.

[36] Sarath Sreedharan, Subbarao Kambhampati, et al. Handling model uncertainty and multiplicity in explanations via model reconciliation. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 28, pages 518–526, 2018.

[37] Michael L Littman, Anthony R Cassandra, and Leslie Pack Kaelbling. Learning policies for partially observable environments: Scaling up. In *Machine Learning Proceedings 1995*, pages 362–370. Elsevier, 1995.

[38] Mor Vered, Reuth Mirsky, Ramon Fraga Pereira, and Felipe Meneguzzi. Advances in goal, plan and activity recognition. *Frontiers in Artificial Intelligence*, 5:861669, 2022.

[39] Shirin Sohrabi, Anton V Riabov, and Octavian Udrea. Plan recognition as planning revisited. In *IJCAI*, pages 3258–3264. New York, NY, 2016.

[40] Alan Fern, Sriraam Natarajan, Kshitij Judah, and Prasad Tadepalli. A decision-theoretic model of assistance. *Journal of Artificial Intelligence Research*, 50:71–104, 2014.

[41] Gregor Behnke, Benedikt Leichtmann, Pascal Bercher, Daniel Holler, Verena Nitsch, Martin Baumann, and Susanne Biundo. Help me make a dinner! challenges when assisting humans in action planning. In *2017 international conference on companion technology (ICCT)*, pages 1–6. IEEE, 2017.

[42] Tathagata Chakraborti, Gordon Briggs, Kartik Talamadupula, Yu Zhang, Matthias Scheutz, David Smith, and Subbarao Kambhampati. Planning for serendipity. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5300–5306. IEEE, 2015.

[43] Dylan Hadfield-Menell, Stuart J Russell, Pieter Abbeel, and Anca Dragan. Cooperative inverse reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

[44] Maayan Shvo and Sheila A McIlraith. Active goal recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9957–9966, 2020.

[45] Yi Chu, Young Chol Song, Richard Levinson, and Henry Kautz. Interactive activity recognition and prompting to assist people with cognitive disabilities. *Journal of Ambient Intelligence and Smart Environments*, 4(5):443–459, 2012.

[46] Gal A Kaminka, David V Pynadath, and Milind Tambe. Monitoring deployed agent teams. In *Proceedings of the fifth international conference on Autonomous agents*, pages 308–315, 2001.

[47] David E Wilkins, Thomas J Lee, and Pauline Berry. Interactive execution monitoring of agent teams. *Journal of Artificial Intelligence Research*, 18:217–261, 2003.

[48] Sachini Weerawardhana, Darrell Whitley, and Mark Roberts. Models of intervention: Helping agents and human users avoid undesirable outcomes. *Frontiers in Artificial Intelligence*, 4:723936, 2022.

[49] Sailik Sengupta, Tathagata Chakraborti, Sarath Sreedharan, Satya Gautam Vadlamudi, and Subbarao Kambhampati. Radar—a proactive decision support system for human-in-the-loop planning. In *2017 AAAI Fall Symposium Series*, 2017.

[50] Aditya Prasad Mishra, Sailik Sengupta, Sarath Sreedharan, Tathagata Chakraborti, and Subbarao Kambhampati. Cap: A decision support system for crew scheduling using automated planning. *Naturalistic Decision Making*, 2019.

[51] Karthik Valmeekam, Sarath Sreedharan, Sailik Sengupta, and Subbarao Kambhampati. Radar-x: An interactive mixed initiative planning interface pairing contrastive explanations and revised plan suggestions. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 32, pages 508–517, 2022.

[52] George Baryannis, Samir Dani, Sahar Validi, and Grigoris Antoniou. Decision support systems and artificial intelligence in supply chain risk management. *Revisiting supply chain risk*, pages 53–71, 2019.

[53] Shirin Sohrabi, Anton Riabov, Michael Katz, and Octavian Udrea. An ai planning solution to scenario generation for enterprise risk management. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[54] Mario Selvaggio, Marco Cognetti, Stefanos Nikolaidis, Serena Ivaldi, and Bruno Siciliano. Autonomy in physical human-robot interaction: A brief survey. *IEEE Robotics and Automation Letters*, 6(4):7989–7996, 2021.

[55] Shervin Javdani, Siddhartha S Srinivasa, and J Andrew Bagnell. Shared autonomy via hindsight optimization. *Robotics science and systems: online proceedings*, 2015, 2015.

[56] Aurélie Clodic, Rachid Alami, and Raja Chatila. Key elements for joint human-robot action. In *Robo-Philosophy*, volume 273, pages 23–33. IOS Press Ebooks, 2014.

[57] Aurelie Clodic and Rachid Alami. What is it to implement a human-robot joint action? *Robotics, AI, and humanity: Science, ethics, and policy*, pages 229–238, 2021.

[58] Kartik Talamadupula, Gordon Briggs, Tathagata Chakraborti, Matthias Scheutz, and Subbarao Kambhampati. Coordination in human-robot teams using mental modeling and plan recognition. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2957–2962. IEEE, 2014.

[59] Fabio Cuzzolin, Alice Morelli, Bogdan Cirstea, and Barbara J Sahakian. Knowing me, knowing you: theory of mind in ai. *Psychological medicine*, 50(7):1057–1061, 2020.

[60] Chris L Baker and Joshua B Tenenbaum. Modeling human plan recognition using bayesian theory of mind. *Plan, activity, and intent recognition: Theory and practice*, 7:177–204, 2014.

[61] Mark K Ho, Rebecca Saxe, and Fiery Cushman. Planning with theory of mind. *Trends in Cognitive Sciences*, 26(11):959–971, 2022.

[62] Grégoire Milliez, Matthieu Warnier, Aurélie Clodic, and Rachid Alami. A framework for endowing an interactive robot with reasoning capabilities about perspective-taking and belief management. In *The 23rd IEEE international symposium on robot and human interactive communication*, pages 1103–1109. IEEE, 2014.

[63] Malte Helmert. The fast downward planning system. *Journal of Artificial Intelligence Research*, 26:191–246, 2006.

[64] Intel Xeon Processor. E5-2630 v4, 2016.

[65] Sylvain Daronnat, Leif Azzopardi, Martin Halvey, and Mateusz Dubiel. Impact of agent reliability and predictability on trust in real time human-agent collaboration. In *Proceedings of the 8th International Conference on Human-Agent Interaction*, pages 131–139, 2020.

[66] Mor Vered, Piers Howe, Tim Miller, Liz Sonenberg, and Eduardo Velloso. Demand-driven transparency for monitoring intelligent agents. *IEEE Transactions on Human-Machine Systems*, 50(3):264–275, 2020.

[67] Mor Vered, Tali Livni, Piers Douglas Lionel Howe, Tim Miller, and Liz Sonenberg. The effects of explanations on automation bias. *Artificial Intelligence*, 322:103952, 2023.

[68] Mara Graziani, Lidia Dutkiewicz, Davide Calvaresi, José Pereira Amorim, Katerina Yordanova, Mor Vered, Rahul Nair, Pedro Henriques Abreu, Tobias Blanke, Valeria Pulignano, et al. A global taxonomy of interpretable ai: unifying the terminology for the technical and social sciences. *Artificial intelligence review*, 56(4):3473–3504, 2023.

[69] Andrew Silva, Mariah Schrum, Erin Hedlund-Botti, Nakul Gopalan, and Matthew Gombolay. Explainable artificial intelligence: Evaluating the objective and subjective impacts of xai on human-agent interaction. *International Journal of Human–Computer Interaction*, 39(7):1390–1404, 2023.

[70] Tim Miller. Explanation in Artificial Intelligence: Insights from the Social Sciences. *Artificial intelligence*, 2019.

[71] Xinru Wang and Ming Yin. Are explanations helpful? a comparative study of the effects of explanations in ai-assisted decision-making. In *Proceedings of the 26th International Conference on Intelligent User Interfaces*, pages 318–328, 2021.

[72] Stefan Sarkadi, Benjamin Wright, Peta Masters, and Peter McBurney. *Deceptive AI*. Springer, 2021.

[73] David Gunning and David W. Aha. Darpa's explainable artificial intelligence (XAI) program. *AI Mag.*, 40(2):44–58, 2019.

[74] William R Swartout and Johanna D Moore. Explanation in second generation expert systems. In *Second generation expert systems*, pages 543–585. Springer, 1993.

[75] Bruce Chandrasekaran, Michael C. Tanner, and John R. Josephson. Explaining control strategies in problem solving. *IEEE Annals of the History of Computing*, 4(01):9–15, 1989.

[76] Tathagata Chakraborti, Sarath Sreedharan, and Subbarao Kambhampati. The emerging landscape of explainable automated planning & decision making. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 4803–4811, 2021.

[77] Abeer Alshehri, Tim Miller, and Mor Vered. Explainable goal recognition: a framework based on weight of evidence. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 33, pages 7–16, 2023.

[78] Helena Vasconcelos, Gagan Bansal, Adam Fourney, Q Vera Liao, and Jennifer Wortman Vaughan. Generation probabilities are not enough: Exploring the effectiveness of uncertainty highlighting in ai-powered code completions. *arXiv preprint arXiv:2302.07248*, 2023.

[79] Tim Miller. Explainable ai is dead, long live explainable ai! hypothesis-driven decision support using evaluative ai. In *Proceedings of the 2023 ACM conference on fairness, accountability, and transparency*, pages 333–342, 2023.

[80] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

[81] Sarah Keren, Avigdor Gal, and Erez Karpas. Goal recognition design. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 24, pages 154–162, 2014.

[82] Bonnie M Muir. Trust in automation: Part i. theoretical issues in the study of trust and human intervention in automated systems. *Ergonomics*, 37(11):1905–1922, 1994.

[83] Robert R Hoffman, Shane T Mueller, Gary Klein, and Jordan Litman. Metrics for explainable ai: Challenges and prospects. *arXiv preprint arXiv:1812.04608*, 2018.

[84] Matija Franklin. The influence of explainable artificial intelligence: Nudging behaviour or boosting capability? *arXiv preprint arXiv:2210.02407*, 2022.

[85] Valdemar Danry, Pat Pataranutaporn, Ziv Epstein, Matthew Groh, and Pattie Maes. Deceptive ai systems that give explanations are just as convincing as honest ai systems in human-machine decision making. *arXiv preprint arXiv:2210.08960*, 2022.

[86] Xinru Wang and Ming Yin. Effects of explanations in ai-assisted decision making: Principles and comparisons. *ACM Transactions on Interactive Intelligent Systems*, 12(4):1–36, 2022.

[87] Feiyu Xu, Hans Uszkoreit, Yangzhou Du, Wei Fan, Dongyan Zhao, and Jun Zhu. Explainable ai: A brief survey on history, research areas, approaches and challenges. In *Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part II 8*, pages 563–574. Springer, 2019.

[88] Tathagata Chakraborti, Sarath Sreedharan, and Subbarao Kambhampati. The emerging landscape of explainable ai planning and decision making. *arXiv preprint arXiv:2002.11697*, 2020.

[89] Tathagata Chakraborti, Sarath Sreedharan, and Subbarao Kambhampati. Human-aware planning revisited: A tale of three models. In *IJCAI-ECAI XAI/ICAPS XAIP Workshops*, volume 10, 2018.

[90] Riccardo Guidotti. Counterfactual explanations and how to find them: literature review and benchmarking. *Data Mining and Knowledge Discovery*, pages 1–55, 2022.

[91] Sahil Verma, Varich Boonsanong, Minh Hoang, Keegan E Hines, John P Dickerson, and Chirag Shah. Counterfactual explanations and algorithmic recourses for machine learning: A review. *arXiv preprint arXiv:2010.10596*, 2020.

[92] Maximilian Diehl, Tathagata Chakraborti, and Karinne Ramirez-Amaro. Guided demonstrations using automated excuse generation. *arXiv preprint arXiv:2311.18355*, 2023.

[93] Pascal Bercher, Susanne Biundo, Thomas Geier, Thilo Hoernle, Florian Nothdurft, Felix Richter, and Bernd Schattenberg. Plan, repair, execute, explain—how planning helps to

assemble your home theater. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 24, pages 386–394, 2014.

[94] Sarath Sreedharan, Christian Muise, and Subbarao Kambhampati. Generalizing action justification and causal links to policies. In *ICAPS*, pages 417–426. AAAI Press, 2023.

[95] Benjamin Krarup, Senka Krivic, Daniele Magazzeni, Derek Long, Michael Cashmore, and David E Smith. Contrastive explanations of plans through model restrictions. *Journal of Artificial Intelligence Research*, 72:533–612, 2021.

[96] Sarath Sreedharan, Siddharth Srivastava, David Smith, and Subbarao Kambhampati. Why can't you do that hal? explaining unsolvability of planning tasks. In *International Joint Conference on Artificial Intelligence*, 2019.

[97] Sachin Grover, Sailik Sengupta, Tathagata Chakraborti, Aditya Prasad Mishra, and Subbarao Kambhampati. Radar: automated task planning for proactive decision support. *Human–Computer Interaction*, 35(5-6):387–412, 2020.

[98] Been Kim and Finale Doshi-Velez. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[99] Shalmali Joshi, Oluwasanmi Koyejo, Warut Vijitbenjaronk, Been Kim, and Joydeep Ghosh. Towards realistic individual recourse and actionable explanations in black-box decision making systems. *arXiv preprint arXiv:1907.09615*, 2019.

[100] Mark T Keane and Barry Smyth. Good counterfactuals and where to find them: A case-based technique for generating counterfactuals for explainable ai (xai). In *Case-Based Reasoning Research and Development: 28th International Conference, ICCBR 2020, Salamanca, Spain, June 8–12, 2020, Proceedings 28*, pages 163–178. Springer, 2020.

[101] Sarath Sreedharan, Tathagata Chakraborti, Christian Muise, and Subbarao Kambhampati. Expectation-aware planning: A unifying framework for synthesizing and executing self-

explaining plans for human-aware planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2518–2526, 2020.

[102] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[103] Stefan Palan and Christian Schitter. Prolific. ac—a subject pool for online experiments. *Journal of Behavioral and Experimental Finance*, 17:22–27, 2018.

[104] Sandra G Hart and Lowell E Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, volume 52, pages 139–183. Elsevier, 1988.

[105] Denise M Rousseau, Sim B Sitkin, Ronald S Burt, and Colin Camerer. Not so different after all: A cross-discipline view of trust. *Academy of management review*, 23(3):393–404, 1998.

[106] Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning: Theory and Practice*. Elsevier, 2004.

[107] Patrik Haslum, Nir Lipovetzky, Daniele Magazzeni, and Christian Muise. An Introduction to the Planning Domain Definition Language. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2019.

[108] Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. PDDL – The Planning Domain Definition Language. Technical Report CVC TR98003/DCS TR1165, New Haven, CT: Yale Center for Computational Vision and Control, 1998.

[109] Christian Muise and Nir Lipovetzky. International plannning competition (ipc) unsolvability track. https://unsolve-ipc.eng.unimelb.edu.au, 2023.

[110] Christer Bäckström, Peter Jonsson, and Simon Ståhlberg. Fast Detection of Unsolvable Planning Instances Using Local Consistency. In *SoCS*, 2013.

[111] Leonardo Henrique Moreira and Célia Ghedini Ralha. Improving Multi-agent Planning with Unsolvability and Independent Plan Detection. In *Brazilian Conference on Intelligent Systems (BRACIS)*, 2017.

[112] Jörg Hoffmann, Peter Kissmann, and Alvaro Torralba. "Distance"? Who Cares? Tailoring Merge-and-Shrink Heuristics to Detect Unsolvability. In *ECAI*, 2014.

[113] Simon Ståhlberg. Tailoring Pattern Databases for Unsolvable Planning Instances. In *ICAPS*, 2017.

[114] Simon Ståhlberg, Guillem Francès, and Jendrik Seipp. Learning Generalized Unsolvability Heuristics for Classical Planning. In *IJCAI*, 2021.

[115] Salomé Eriksson, Gabriele Röger, and Malte Helmert. Unsolvability Certificates for Classical Planning. In *ICAPS*, 2017.

[116] Salomé Eriksson, Gabriele Röger, and Malte Helmert. A Proof System for Unsolvable Planning Tasks. In *ICAPS*, 2018.

[117] Salomé Eriksson and Malte Helmert. Certified Unsolvability for SAT Planning with Property Directed Reachability. In *ICAPS*, 2020.

[118] Andreas Herzig, Viviane Menezes, Leliane Nunes de Barros, and Renata Wassermann. On the revision of planning tasks. In *ECAI 2014*, pages 435–440. IOS Press, 2014.

[119] Sarath Sreedharan, Tathagata Chakraborti, Christian Muise, Yasaman Khazaeni, and Subbarao Kambhampati. D3WA+ – A Case Study of XAIP in a Model Acquisition Task for Dialogue Planning. In *ICAPS*, 2020.

[120] Lucas Galery Käser, Clemens Büchner, Augusto B Corrêa, Florian Pommerening, and Gabriele Röger. Machetli: Simplifying Input Files for Debugging. In *ICAPS System Demonstrations Track*, 2022.

[121] Sarath Sreedharan, Tathagata Chakraborti, Christian Muise, and Subbarao Kambhampati. Expectation-Aware Planning: A General Framework for Synthesizing and Executing Self-Explaining Plans for Human-AI Interaction. In *AAAI*, 2020.

[122] Tathagata Chakraborti, Sarath Sreedharan, and Subbarao Kambhampati. The Emerging Landscape of Explainable AI Planning and Decision Making. In *IJCAI*, 2020.

[123] Maria Fox, Derek Long, and Daniele Magazzeni. Explainable Planning. *arXiv:1709.10256*, 2017.

[124] Tathagata Chakraborti, Anagha Kulkarni, Sarath Sreedharan, David E. Smith, and Subbarao Kambhampati. Explicability? Legibility? Predictability? Transparency? Privacy? Security? The Emerging Landscape of Interpretable Agent Behavior. In *ICAPS*, 2019.

[125] Tathagata Chakraborti and Subbarao Kambhampat. (How) Can AI Bots Lie? In *ICAPS Workshop on Explainable AI Planning*, 2019.

[126] Julie Porteous, Alan Lindsay, Jonathon Read, Mark Truran, and Marc Cavazza. Automated Extension of Narrative Planning Domains with Antonymic Operators. In *AAMAS*, 2015.

[127] Julie Porteous. Planning Technologies for Interactive Storytelling. *Handbook of Digital Games and Entertainment Technologies*, 2016.

[128] Tathagata Chakraborti and Subbarao Kambhampat. (When) Can Bots Lie? In *AIES*, 2019.

[129] Mauro Vallati and Diane Kitchin. *Knowledge Engineering Tools and Techniques for AI Planning*. Springer, 2020.

[130] Reuth Mirsky, Kobi Gal, Roni Stern, and Meir Kalech. Goal and Plan Recognition Design for Plan Libraries. *ACM TIST*, 2019.

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

[132] Anagha Kulkarni, Sarath Sreedharan, Sarah Keren, Tathagata Chakraborti, David E. Smith, and Subbarao Kambhampati. Design for Interpretability. In *ICAPS Workshop on Explainable AI Planning*, 2019.

[133] Anagha Kulkarni, Sarath Sreedharan, Sarah Keren, Tathagata Chakraborti, David E. Smith, and Subbarao Kambhampati. Designing Environments Conducive to Interpretable Robot Behavior. In *IROS*, 2020.

[134] Christian Muise, Tathagata Chakraborti, Shubham Agarwal, Ondrej Bajgar, Arunima Chaudhary, Luis A. Lastras-Montano, Josef Ondrej, Miroslav Vodolan, and Charlie Wiecha. Planning for Goal-Oriented Dialogue Systems. Technical report, IBM Research AI, 2020.

[135] Zahra Zahedi, Alberto Olmo, Tathagata Chakraborti, Sarath Sreedharan, and Subbarao Kambhampati. Towards Understanding User Preferences for Explanation Types in Explanation as Model Reconciliation. In *HRI Late Breaking Report*, 2019.

[136] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, et al. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. *Proceedings of Machine Learning Research*, 2023.

[137] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner Monologue: Embodied Reasoning through Planning with Language Models. *CoRL*, 2023.

[138] LangChain. LangChain is a framework for developing applications powered by language models. https://python.langchain.com, 2023.

[139] John Maeda and Evan Chaki. Semantic kernel. https://learn.microsoft.com/en-us/semantic-kernel, 2023.

[140] Christian Muise. Api.planning.domains: An interface to the repository of pddl domains and problems. http://api.planning.domains, 2023.

[141] Sarath Sreedharan, Tathagata Chakraborti, Yara Rizk, and Yasaman Khazaeni. Explainable Composition of Aggregated Assistants. In *ICAPS Workshop on Explainable AI Planning*, 2020.

[142] Guillem Francés, Miquel Ramirez, and Collaborators. Tarski: An AI Planning Modeling Framework. https://github.com/aig-upf/tarski, 2018.

[143] Dan Bryce, J Benton, and Michael W Boldt. Maintaining Evolving Domain Models. In *IJCAI*, 2016.

[144] Nisha Simon and Christian Muise. TattleTale: Storytelling with Planning and Large Language Models. In *ICAPS Workshop on Scheduling and Planning Applications*, 2022.

[145] Julie Porteous, João F Ferreira, Alan Lindsay, and Marc Cavazza. Automated Narrative Planning Model Extension. In *AAMAS*, 2021.

[146] OpenAI. GPT-4 Technical Report. *arXiv:2303.08774*, 2023.

[147] Patrick Ferber, Malte Helmert, and Jörg Hoffmann. Neural Network Heuristics for Classical Planning: A Study of Hyperparameter Space. In *ECAI 2020*, 2020.

[148] Sergio Jiménez Celorrio. Domainssequential. http://www.plg.inf.uc3m.es/ipc2011-deterministic/DomainsSequential.html#Barman, 2011.

[149] Alba Gragera and Alberto Pozanco. Exploring the Limitations of using Large Language Models to Fix Planning Tasks. In *ICAPS Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*, 2023.

[150] Richard Howey, Derek Long, and Maria Fox. VAL: Automatic Plan Validation, Continuous Effects and Mixed Initiative Planning Using PDDL. In *IEEE International Conference on Tools with Artificial Intelligence*, 2004.