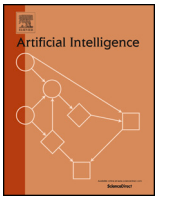
Artificial Intelligence 340 (2025) 104279

Contents lists available at ScienceDirect

Artificial Intelligence

journal homepage: www.elsevier.com/locate/artint



Explain it as simple as possible, but no simpler – Explanation via model simplification for addressing inferential gap

Sarath Sreedharan [a,*], Siddharth Srivastava [b], Subbarao Kambhampati [b]

[a] *Colorado State University, Fort Collins, CO 80524, USA*
[b] *Arizona State University, Tempe, AZ 85281, USA*

ARTICLE INFO

Keywords:
Explanations for plans
Abstractions
Contrastive explanations

ABSTRACT

One of the core challenges of explaining decisions made by modern AI systems is the need to address the potential gap in the inferential capabilities of the system generating the decision and the user trying to make sense of it. This *inferential capability gap* becomes even more critical when it comes to explaining sequential decisions. While there have been some isolated efforts at developing explanation methods suited for complex decision-making settings, most of these current efforts are limited in scope. In this paper, we introduce a general framework for generating explanations in the presence of inferential capability gaps. A framework that is grounded in the generation of simplified representations of the agent model through the application of a sequence of model simplifying transformations. This framework not only allows us to develop an extremely general explanation generation algorithm, but we see that many of the existing works in this direction could be seen as specific instantiations of our more general method. While the ideas presented in this paper are general enough to be applied to any decision-making framework, we will focus on instantiating the framework in the context of stochastic planning problems. As a part of this instantiation, we will also provide an exhaustive characterization of explanatory queries and an analysis of various classes of applicable transformations. We will evaluate the effectiveness of transformation-based explanations through both synthetic experiments and user studies.

1. Introduction

While AI as a field has made rapid progress in recent years, we are still far from realizing the truly transformational potential of AI systems in our society. Chief among the obstacles to achieving this goal is the inability of our current AI systems to work effectively with people from all walks of life. Meeting this challenge requires us to rethink our current paradigms for agent design. We need to look beyond just measuring the effectiveness of agents in terms of their capability to achieve their assigned goals, and consider their ability to communicate and explain their decisions to their end-users intuitively. Additionally, they should allow their users to critique their decisions and, as required, be able to incorporate user suggestions. While the creation of explainable agents capable of supporting such interactions has been receiving significant attention lately [1,2], we are still at the beginning stages of creating systematic approaches to explaining AI agent decisions and facilitating such interactive dialogues.

One of the core challenges of explaining any automated decisions, particularly in sequential decision-making problems, is the complexity of the underlying reasoning problems. Many modern planners and reinforcement learning (RL) agents can generate solu-

* Corresponding author.
E-mail address: ssreedh3@colostate.edu (S. Sreedharan).

https://doi.org/10.1016/j.artint.2024.104279
Received 1 January 2023; Received in revised form 21 December 2024; Accepted 26 December 2024

Available online 7 January 2025

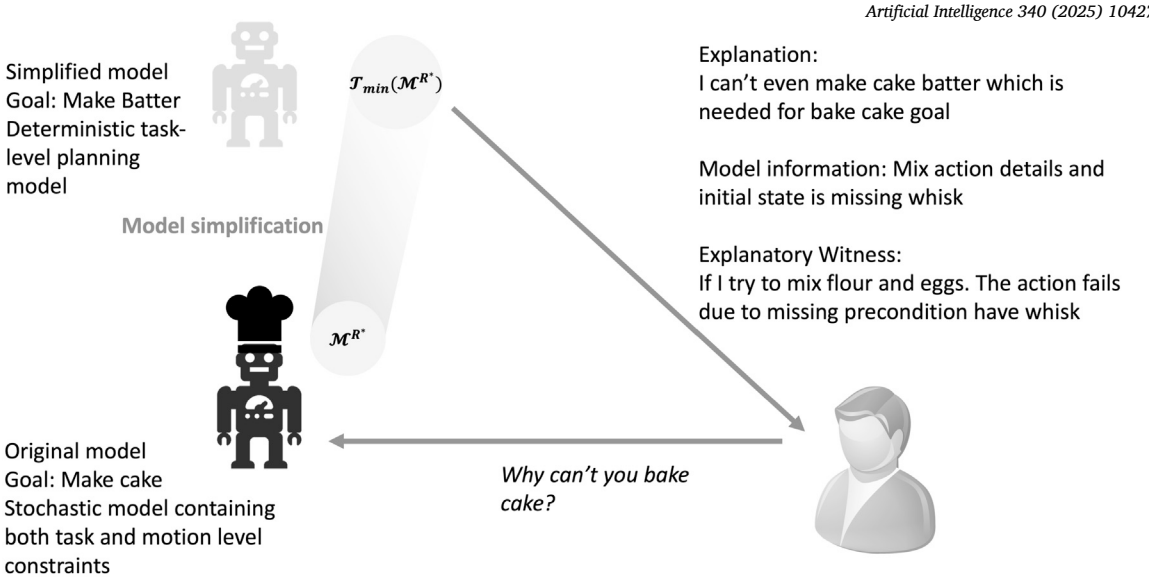


Fig. 1. The model simplification process used to generate the explanations for the motivating example, where a robotic chef needs to explain its inability to bake a cake.

tions to problems that most everyday users would find hard to comprehend completely, let alone solve. In other words, the problem of generating explanations in these scenarios are made more challenging by the presence of an *inferential capability gap* between the user and the decision-making system. Thankfully for most explanatory queries that a user may raise, we would hardly ever need to expose them to the full complexity of the original problem. Instead, we can often get by using just a simplified representation that suffices to answer their specific question. In this paper, we will operationalize this intuition to generate effective explanations by introducing an explanation generation framework that generates simplified representations of the original model that is sufficient to respond to a given user query. To ground this framework, we will look at the use of this explanatory framework in the context of a very general version of sequential decision-making problems, namely, goal-directed stochastic planning problems.

Fig. 1 presents a diagrammatic representation of the overall explanatory process entailed by the methods presented in this paper. Specifically, the figure refers to the example scenario which considers a robotic chef working in a house.

Example 1. In this example the user of the robot starts by asking the robot to bake a cake, to which it replies it can't bake a cake. Now, the user demands an explanation for its failure to prepare a cake. The robot may have come to the conclusion that it cannot prepare a cake from the fact that its stochastic task and motion planner failed to produce a valid plan. Dumping its search tree or the underlying model to the human would hardly suffice as a satisfactory explanation. Suppose the robot was to analyze the planning problem; it could find out that the problem remains unsolvable even if it were to ignore the stochasticity of the domain (related to various slippages the robot may incur in its operation) and all the motion constraints. The robot isn't even able to create cake batter, which is required for the final goal, because it doesn't have a whisk. The robot could ignore all low-level details and just surface a simplified model to the user containing information about the mixing action, which includes the fact that the action requires access to a whisk. Now, given this model, the robot can walk the human through a hypothetical trace where it tries to make the cake better and show that it would fail at the mixing step due to the missing whisk. Now, this hypothetical trace provides a simple example demonstrating the robot's inability to achieve its goals. We will refer to such secondary explanatory information as *explanatory witness*.

Throughout this paper, we will see how our framework can help generate this exact explanation. As for the rest of the paper, we will start the technical discussion by providing an overview of the framework (Section 3.1). We will provide a small summary for each framework component with pointers to relevant sections that provide a detailed discussion of each component. Section 3.2 will provide formal definitions of some of the central concepts we will be using in the paper, including inferential gap, explanatory criteria, and model simplifying transformations, and lay out the basic algorithm to compute explanatory models. With the basic framework in place, we will delve deeper into the grounding of the framework in the context of stochastic planning problems and go over the characterization of each component of the framework in this setting. This would involve providing an exhaustive characterization of contrastive explanatory queries possible in stochastic planning settings (Section 4) and introducing a set of a model transformation (Section 5). Each model transformation class presents a conceptually consistent formalization of a large class of specific transformations that could be applied to a given planning problem to simplify the problem. For each transformation class, we will also point to some previous works in the wider explainable AI (XAI) literature that have applied similar techniques, thereby presenting a previous instance of a limited application of our more general framework. For each transformation class, we introduce one specific technique that only takes polynomial time to perform and analyze the properties of the specific technique for the class of stochastic planning problems we are interested in. Section 6 will introduce a more constrained version of the algorithm introduced in Section 3 specifically

designed for the transformation classes introduced in Section 5 called the stratified search algorithm. Section 7 presents the various experiments we performed to evaluate the explanation generation method discussed here. In particular, Section 7.1 presents a user study on the effectiveness of our overall framework and measures the effectiveness of the individual transformations. In Section 7.2, we evaluate the effectiveness of the generated explanation using computational proxies for the complexity. In section 8, we will further see how most current works aimed at addressing the inferential capability gap can be seen as a specific instance of our more general framework. In particular, we will discuss how each existing work can be seen as using some limited set of transformations or focuses primarily on providing an explanatory witness. All the frameworks and analyses will be grounded in the context of MDPs with the P assumption (i.e., Positive action cost assumption) [3], which generalizes over many commonly studied stochastic planning formalisms.

2. Background

The decision-making model we will use throughout the paper is the undiscounted MDP with absorbing goal states that tries to generate optimal solutions under total expected cost criteria. Such models can be described by a tuple of the form $\mathcal{M} = \langle S, A, P, C, I, G\rangle$, where S is the state space, A the set of possible actions, $P : S \times A \times S \rightarrow [0, 1]$ the transition probabilities, C captures the cost of executing a given action in the state and it resulting in a new state, $I \in S$ the initial state and $G \subseteq S$, is the set of absorbing goal states. General MDP definitions also allow us to limit what actions are applicable in which states, but we will skip explicitly capturing that to simplify the notations. We will specifically limit ourselves to cases where all state-action-state tuples satisfy the condition $C(s, a, s') \geq 0$. MDPs that satisfy this condition are usually referred to as satisfying the P-assumption or the positive action cost assumption (cf. [3]). While most of the results established in the paper also apply to other settings, say when all costs are guaranteed to be less than or equal to zero, we will mostly focus on the former as it is better suited to goal-directed problems. We wanted to preserve goals in the problem formulation as there is plenty of evidence from psychological studies that show that people tend to think in terms of goals when performing planning [4]. MDPs with P-assumptions are still an extremely general formulation and cover several formalisms studied frequently in AI literature, including non-negative cost infinite horizon discounted MDPs [3], non-negative cost SSPs [3], MAXPROB MDPs [5], NEG Mdps [6], non-negative cost GSSP's [5], SSPUDE's [7] etc.

We will use the functions $J : S \rightarrow \mathbb{R}$ and $Q : S \times A \rightarrow \mathbb{R}$ to capture the expected total cost and Q function and use J^* and Q^* to represent the optimal cost and Q function respectively. A policy is said to be optimal if $J^{\pi}(s) = J^*(s)$, where J^{π} is the cost function obtained by following the given policy. In the most general case, a policy takes the form $\pi = \langle \mu_1, \mu_2, ...\rangle$, where each μ_i is a mapping from state to action for a timestep i. A policy is said to be stationary if the mapping does not depend on the time step. Under the P assumption, value iteration converges if the cost function is initialized with a bounded cost function less than the optimal cost function (say zero cost function), provided there exists a fixed point to the function. In addition to the optimal cost, a factor that we will be considering throughout the paper is the probability that the execution of policy from a given state would lead it to a goal state $P^{\pi}(s) = \sum_{g \in G} P(g|s, \pi)$ and we will $P^*(s)$ to denote the highest possible probability of achieving the goal under any policy for the given model (which we will refer to as the MAXPROB policy) and use $P^{\pi}(s)$ to capture probability under a given policy. In most cases, we will focus on the cost and probability of achieving the goal from the initial state and states reachable from the initial state.

Effectively, in this scenario, planning for MDPs becomes a multi-objective optimization problem, where the objectives become the cost of the solution and the reachability of goals. This may come across as surprising to readers who are most familiar with the more restricted classes of MDPs, where any potential goals are usually compiled into the cost function. Thus, one may anticipate that all goal reachability queries will be resolved through cost differences. More general variants of MDPs like iSSPUDE [7] use the probability of getting to the goal as a separate optimization criterion for choosing the policy. So, instead of choosing a plan that directly optimizes the cost, they use the probability of getting to the goal as the primary criterion and the cost as the secondary one. This also means that in every case, there exists a strict ordering between the objectives, and we don't need to rely on additional considerations like establishing Pareto optimality or considering the two objectives simultaneously.

A given MDP could be represented in multiple ways, a particularly popular way one could represent such models is to describe them using problem description languages like PPDDL [8]. Mathematically a model described in PPDDL is given by a tuple of the form $\mathcal{M}^D = \langle F^D, A^D, I^D, G^D, C^D\rangle$, where

- F^D is the set of propositions used to define the state space. Each state in the model will be a specific instantiation of these propositions. That is, each state s_i can be uniquely represented by the set of facts true in that state, i.e., $s_i \subseteq F$. These propositions are also sometimes referred to as state fluents and state factors in the literature (cf. [9]).
- A^D is the set of actions available in the model. Each action $a_i \in A^D$ is further defined as following $a_i = \langle \text{prec}_i, E_i\rangle$
  - $\text{prec}_i \subseteq F$ denotes the preconditions of the action. An action is only applicable in states where its preconditions are true, i.e., action a_i is executable in a state s_k if $\text{prec}_i \subseteq s_k$.
  - E_i is the set of mutually exclusive effects that the execution of the action can cause. Each individual effect $e_i^j \in E_i$, if further represented by the tuple $\langle \text{add}_i^j, \text{del}_i^j, p_i^j, c_i^j\rangle$, where
    - $\text{add}_i^j \subseteq F$, called the add effects, is the set of facts that will be turned true by that effect in the resultant state.
    - $\text{del}_i^j \subseteq F$, called the delete effects, is the set of facts that will be turned false by that effect in the resultant state.
    - p_i^j, of the probability of occurrence of that particular effect (the distribution over the individual effects is expected to form a well formed probability distribution).
    - c_i^j is the cost associated with the transition

- I^D initial state, we expect the system to start in that state.
- $G^D \subseteq F^D$ is the goal specification, where any state s_i such that $G^D \subseteq s_i$ is considered to be an absorbing goal state (or destination state).

Each valid description of the above form is expected to translate into an MDP of the form $\mathcal{M} = \langle S, A, I, P, C, G \rangle$, where

- S is the set of states of the model and corresponds to the state space defined by F (i.e. $|S| = 2^{|F|}$). For each $i \in S$, we will use s_i to denote the symbolic state (described by the set of propositions that are true in the state).
- A is the action/control space of the underlying MDP and is isomorphic to the set A and will use $a^{mdp}_{a_j}$ to represent the corresponding action for the symbolic action a_j, moreover $A(i) = \{a^{mdp}_{a_j} | \text{prec}_j \subseteq s_i\}$ (i.e. the actions available at a state i).
- I is underlying atomic state corresponding to I^D
- P is transition probability and is defined as follows

  $$P(i, a^{mdp}_{a_k}, j) = p^m_k$$

  if there exists an effect $e^m_k \in E_k$ such that $s_j = (s_i \setminus \text{del}^m_k) \cup \text{add}^m_k$, else it is 0. Note that given the assumption the effects are mutually exclusive, there exists at most one effect that can cause this transition, provided there are no redundant effects (i.e., ones that leave the state unchanged because of adding existing facts or trying to remove missing facts).
- G is the set of destination or goal nodes that correspond to the ones specified by G, i.e., for $i \in G$, the corresponding symbolic state s_i, will satisfy the condition $G^D \subseteq s_i$.
- $C : S \times A \times S$ is the cost function. As with the transition probability, the cost function is given by c^j_i.

For a given description $\mathcal{M}^D$, we will use the notation $\mathcal{M}$ to refer to the MDP induced by it, especially if we don't explicitly mention the underlying MDP. Two facts that might be worth keeping in mind is (1) for any MDP with a finite set of actions and states can be captured by a description and (2) for a fixed fluent and action set, the model description for a given MDP is unique.

Such factored model descriptions are usually easier for people to understand and create, not just due to conciseness, but also because it is based on folk psychology principles regarding actions and as such are intuitive descriptions [10]. We assume that models are provided in such descriptions, though one could always start with an atomic or inscrutable representation of the model and derive such models through different learning methods [11,12].

3. Our approach - the model simplification framework for contrastive explanation generation

3.1. Approach overview

In this section, we provide an abstract overview of the framework and by extension for the paper as a whole. We will touch on each component of the framework, discuss how the component comes into play in generating the explanation discussed in Section 1 and provide relevant pointers to parts of the paper that will cover the component in greater details.

Fig. 2 presents an overview of our explanation generation method. A complete list of all the notations used in the paper can be found in Table 5 (Appendix A.1).

3.1.1. Explanatory queries and explanatory criteria

The explanatory process in our case is initiated when the user raises an explanatory query. This paper will exclusively focus on *contrastive queries* [13], where the user is interested in why the system chose the current decision as opposed to an alternate decision they may have expected. The question discussed in Example 1, i.e., "Why can't you bake the cake?", is in fact a contrastive query presented in an implicit form. One can equivalently represent the query as

"Why is the system claiming that no solution is possible as opposed to following some policy π'?"

For each query, we will identify an *explanatory criterion*. An explanatory criterion corresponds to a bound on the possible value that a valid solution can take along one of the optimization objectives (i.e. cost or probability of reaching the goal), but that would need to be violated for the alternate plan to be preferred over the current decision. For example, in the case of the query discussed in Section 1, an explanatory criterion would be the fact that there exist no policy for the model, in which the probability of reaching the goal from initial state is bigger than zero. This explanatory criterion will be surfaced to the user along with the rest of the explanatory information. We will provide a more formal characterization of explanatory criteria in Section 3.2 and Section 4 will include an exhaustive characterization of all types of explanatory criteria possible in a stochastic planning setting.

3.1.2. Inferential capability gap

Our central explanatory challenge is the fact that there may exist an asymmetry between the human and the system with regards to their computational capabilities (referred to as the *Inferential Capability Gap*). In Example 1, human would find it much harder to use the information regarding the complete task and motion planning problem to verify whether in fact there exists any policy which can get to the goal with non-zero probability. We will operationalize such computational effort required by the human by using a

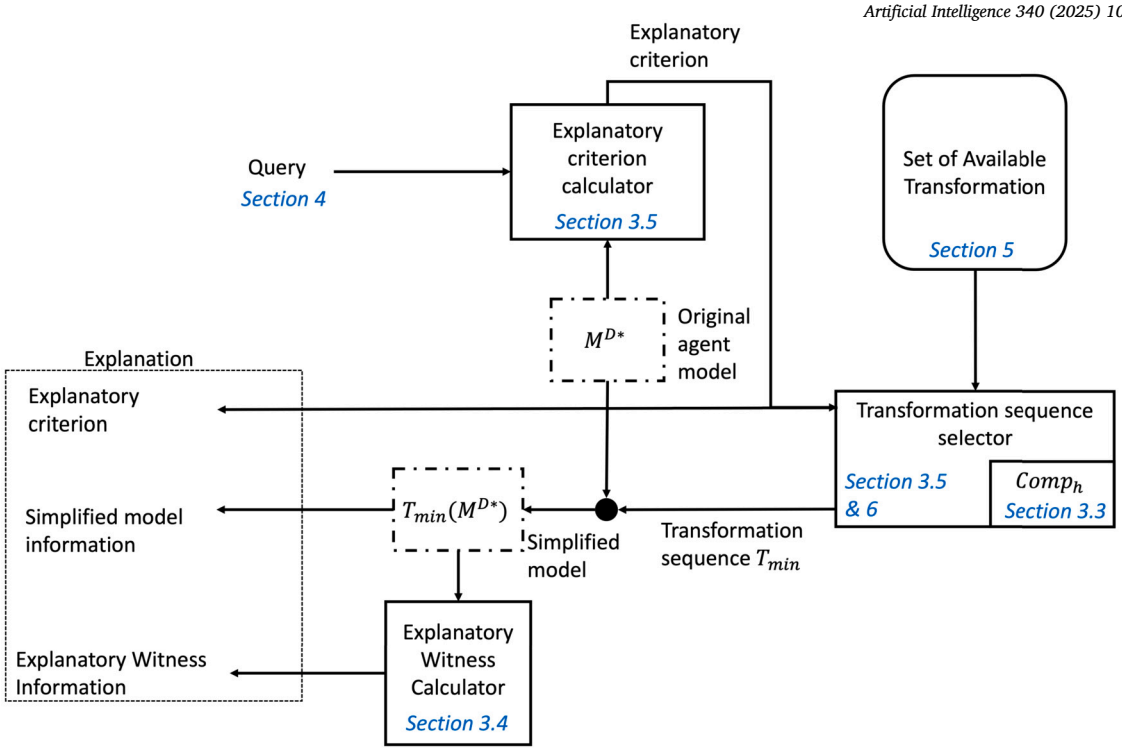


Fig. 2. A diagrammatic overview of the overall explanation generation framework, along with annotations on the paper sections where we will discuss the specific components.

proxy function $Comp_H$. We will use this $Comp_H$ function to identify explanatory information that places minimal computational demands on the user. We will formalize $Comp_H$ in Section 3.2 and delve further into how we can operationalize it in Section 3.3.

3.1.3. Simplifying model transformations

The argument we will make throughout this paper is that even if the complexity of establishing an explanatory criterion (as measured under $Comp_H$) in the original model is high, one can use simplified representations of the original model where one could establish that the same criterion is satisfied with lower effort. In Example 1, rather than asking the user to reason about the unsolvability of the full stochastic task and motion planning problem, the system surfaces a much simpler problem, i.e., it presents its inability to create batter even while ignoring all the motion level constraints and any non-determinism in the task.

We can create such simpler representations, by taking the original model and applying a sequence of model simplification transformations. To ensure that presenting the user with a simplified model doesn't mislead the user about the underlying task, we will additionally enforce that the transformations will not violate any explanatory criteria that were previously true. A requirement we will capture under a concept referred to as the *soundness* of the transformation. The central computational challenge for generating such explanations is to identify the sequence of transformation that will generate the simplest representation that still conserves the explanatory criteria, which previously held. We will formalize model transformations in general in Section 3.2 and discuss specific transformation classes in Section 5. We discuss methods for generating the transformation sequences in Section 3.5 and look at a more specialized algorithm in Section 6.

3.1.4. Explanatory witness

Even after providing the model, the user may not be able to reason about the explanatory criteria on their own and as such may require additional assistance. We will refer to such information as *Explanatory Witness*. These specifically correspond to additional information extracted from the simplified model that demonstrates why the criteria are satisfied by the given model. We will provide a detailed discussion and a classification of the explanatory witness classes in Section 3.4 along with the discussion of how we will generate them for the problems we consider. Also in the related work (Section 8) we will also provide a discussion of the various explanatory witnesses that has been considered previously in the literature.

3.1.5. Final explanatory message

As part of the final explanation the system would surface the following information, (a) an explanatory criterion (though depending on the context this may be implicit), (b) a simplified model where the criterion would be satisfied, and (c) an explanatory witness that demonstrates the fact that the criterion is satisfied.

In Example 1, surfacing the explanatory criterion involves mentioning that there are no policies with non-zero goal reaching probabilities. The model is constructed by skipping motion level constraints, the probability information and considering a more myopic objective (i.e., making batter). Finally, the explanatory witness here consists of the sample trace.

3.2. Framework formalization

Our basic explanatory setting consists of a sound and optimal model-based decision-making system that uses model $\mathcal{M}^{R^*}$ (which may correspond to a description $\mathcal{M}^{D^*}$), to come up with its decisions and it needs to respond to possible explanatory queries a user of the system may raise either about its current decision or about alternative decisions the user may have expected.

The central focus in this paper is to provide *contrastive explanations* [13], where contrastive explanations are said to be explanations that take the form of responses to questions of the form "Why P and not Q?", where 'P' is referred to as the fact being explained and 'Q' is referred to as the foil the fact is being contrasted against. There is a lot of evidence from social science literature that contrastive explanations underpin most of our everyday explanatory dialogues and, as such, have been receiving a lot of attention in recent years [13–15]. In our scenario, we are always interested in answering questions of the form.

"Why policy π and not any policy in the set Π'"

That is the explanation always comes down to establishing the choice of one policy π (possibly the current one being proposed by the system), over the set of alternate policies (possibly expected by the user). More formally,

Definition 1. An **explanatory query** is represented by a tuple of the form $\mathcal{Q} = \langle \pi, \Pi' \rangle$, where π, referred to as the fact policy, corresponds to the policy proposed by the system and Π', referred to as the foil set, corresponds to the alternate policies expected by the user.

To support explaining unsolvable planning problems (i.e., there exists no policy with non-zero probability of reaching the goal), we will use the notation $\emptyset$ to denote the fact policy for an unsolvable planning problem.

In cases where the system is a sound, complete and optimal decision-maker, the system can respond to an explanatory query by establishing the preference of one fact policy over the foil set (or at the very least establishing they are equivalent). In the MDP classes studied in this paper, this always takes the form of establishing that the fact policy either has a higher probability of achieving the goal or lower cost as compared to the policies in the foil set.

In this paper, we will argue that one could achieve a deeper insight into the explanatory process by separating the form of the question from the underlying model constraint that needs to be established as part of resolving the user query. In essence, one could convert every contrastive explanatory query into a problem of establishing that a planning model can only generate solutions that satisfy a particular threshold with respect to one of its optimization objectives.

However before defining an explanatory criterion, we will introduce the notion of a policy satisfying an optimization threshold. Specifically, an optimization threshold specifies a threshold (i.e., an upper bound or a lower bound) over one of the optimization metrics (i.e., cost or goal-reaching probability). In the case of cost, we will limit ourselves to a lower bound and for reachability we will only consider upper bounds. A policy is said to satisfy the threshold for a given model, if the threshold satisfied by the valuation of the policy in the initial state (or more generally a state of interest) in the dimension of the optimization metric specified in the threshold. More formally, we will denote this as,

Definition 2. An **optimization threshold** is denoted by a tuple $\tau = \langle \mathcal{O}, \mathcal{X} \rangle$, where $\mathcal{O}$ is an optimization objective (cost function J or goal-reaching probability P) and $\mathcal{X}$ is an upper bound or a lower bound over $\mathcal{O}$. A policy π is said to satisfy τ in a model $\mathcal{M}$, if $\mathcal{O}^{\pi}(I)$ as evaluated in the model $\mathcal{M}$ satisfies $\mathcal{X}$.

An explanatory criterion is essentially an optimization threshold that is satisfied by the fact policy but not by the foil policies.

Definition 3. For an explanatory query $\mathcal{Q} = \langle \pi, \Pi' \rangle$ and an optimization threshold $\kappa = \langle \mathcal{O}, \mathcal{X} \rangle$, a model $\mathcal{M}$ is said to satisfy κ with respect to $\mathcal{Q}$, if under model $\mathcal{M}$, π satisfies κ but none of the foil policies in Π' does (denoted as $\mathcal{M} \models_{\mathcal{Q}} \kappa$).

Definition 4. For an explanatory query $\mathcal{Q} = \langle \pi, \Pi' \rangle$, an optimization threshold $\kappa = \langle \mathcal{O}, \mathcal{X} \rangle$, is said **to be an explanatory criterion** for a model $\mathcal{M}$, if π is preferred over Π' when $\mathcal{M}$ satisfies the criterion κ with respect to $\mathcal{Q}$ (i.e., $\mathcal{M} \models_{\mathcal{Q}} \kappa$).

The above definition lays out explanatory criteria as applicable in cases where the foil set may be explicitly enumerated. In many cases, the foil set may only be implicitly specified. As we previously saw, the foil set in Example 1, corresponds to set of all policies. In this case, the explanatory criterion must be satisfied by all possible solutions for the given model. To capture this case, we will treat the satisfaction of explanatory criteria as a model-level property, i.e., we will simply use the notation $\mathcal{M} \models \kappa$ or $\mathcal{M} \not\models \kappa$ by dropping the query subscript.

Definition 5. For an optimization threshold $\kappa = \langle \mathcal{O}, \mathcal{X} \rangle$, a model $\mathcal{M}$ is said to satisfy κ, if for model $\mathcal{M}$, every policy π satisfies κ (denoted as $\mathcal{M} \models \kappa$).

Note that this is a more general formulation, as we can always capture explicit foil and the fact policy cases by creating new models that only allow the policies in question.

With respect to the Example 1, one explanatory criterion could be $\kappa = \langle P, = 0 \rangle$*, i.e., if the criterion* κ *is satisfied by the model, then the model is only allowed to support policies whose probability of achieving the goal is equal to* 0*.*

Section 4 presents an exhaustive characterization of all explanatory criteria possible in a stochastic planning setting and how they connect to every possible contrastive query.

While explanatory criteria help establish why the foils may not be preferred, we still have to discuss how the user may realize that the explanatory criterion is satisfied for the given problem. Towards this end, the paper builds on the basic fact that humans are agents capable of reasoning about sequential decision-making problems. Thus, as long as the model is in a human-readable form,[1] the system could communicate the model information and expect the human to try to reason about whether the relevant explanatory criteria satisfy on their own. However, such a naive approach would overlook the fact that the original decision-making problem may be too complex for the user.

To ensure that our explanation method does not place undue cognitive demands on the user, we will try to explicitly take into account how easy it is for a user to verify whether an explanatory criterion is satisfied in a given model. To capture the inferential burden placed on the human or any agent by an explanation, we will introduce the function $Comp(\cdot, \cdot)$ to denote the computational capability of the agent. This function will capture the time taken by the agent to establish whether an explanatory property is satisfied for a given model.

Definition 6. For a given set of models $\mathbb{M}$ and a set of explanatory properties $\mathbb{K}$, the **computational capability function** $Comp : \mathbb{M} \times \mathbb{K} \rightarrow \mathbb{R} \cup \{\infty\}$ function returns the time taken by a reasoning agent to establish whether an explanatory criterion $\kappa \in \mathbb{K}$ is satisfied in a model $\mathcal{M} \in \mathbb{M}$. The function returns ∞ if the agent is incapable of establishing whether the criterion is satisfied.

We will use $Comp_H$ to denote the computational capability function for the human and $Comp_{sys}$ for the one associated with the AI agent. There effectively exists an inferential capability gap between the two if the computational capability function returns different values for the same model and explanatory criterion.

Definition 7. An **inferential capability gap** is said to exist between a human (with function $Comp_H$) and a decision-making system (with function $Comp_{sys}$) for a model $\mathcal{M}$ and an explanatory criterion κ, if

$$Comp_{sys}(\mathcal{M}, \kappa) \neq Comp_H(\mathcal{M}, \kappa)$$

In theory, we could have $Comp_{sys}(\mathcal{M}, \kappa) > Comp_H(\mathcal{M}, \kappa)$, i.e., cases where for the same model the human has an easier time establishing some criterion, but in this paper we are mostly interested in cases where we have $Comp_{sys}(\mathcal{M}, \kappa) < Comp_H(\mathcal{M}, \kappa)$

Now the central argument we will make throughout this paper is that even if the complexity of establishing an explanatory property in the original model is high (measured in terms of $Comp_H(\mathcal{M}, \kappa)$), one can use simplified representations of the original model where one could establish that the criterion is satisfied with lower effort. We can now give the user information about this simplified model and even use this model as a basis for generating other explanatory information. In particular, we will look at the generation of explanatory witnesses that will help the user better understand why the criterion is satisfied in the current model.

Towards formalizing this intuition of explanation generation, we will first start by introducing the notion of model transformation that will become the basis of our remaining framework. In the paper, we will refer to a model transformation function as one that takes a valid MDP model and generates a new MDP specification, i.e.,

Definition 8. A model transformation is a function of the form $\mathcal{F}_. : \mathcal{M}_i \mapsto \mathcal{M}_j$ where $\mathcal{M}_i$ was the original MDP and $\mathcal{M}_j$ is the newly generated MDP, where both $\mathcal{M}_i$ and $\mathcal{M}_j$ are two MDPs that satisfy P-assumptions.

Note that the above definition is an extremely weak notion and not one that in any way helps us build model representations that could help resolve users' queries. To establish that, we need to introduce two new notions about the transformations, namely, soundness of transformation and whether the transformations are building simpler representations.

The soundness of transformation relates to whether any explanatory criterion that is satisfied in the new model has a corresponding criterion that was satisfied in the original model, or more formally:

Definition 9. A model transformation $\mathcal{F}_i$ is said to be a **sound transformation** for a model $\mathcal{M}$ with respect to an explanatory criterion κ_i, if whenever κ_i is satisfied by $\mathcal{F}_i(\mathcal{M})$, it is also satisfied by $\mathcal{M}$.

Note that we are not requiring the transformations to always preserve the explanatory criterion, i.e., κ_i satisfied by $\mathcal{M}$ if and only if κ_i satisfied by $\mathcal{F}_i(\mathcal{M})$. As we will soon see most of the effective transformations we will look at results in optimistic approximations,

[1] We assume our models to be already in a human-readable form, and even if they are not one could always convert them to such forms using methods like those discussed in [11].

where we can't always guarantee that the transformations will preserve the criterion, but we can be sure that the transformations are sound as defined above, i.e., there couldn't exist a criterion κ, such that $\mathcal{M} \nvDash \kappa$ but $\mathcal{F}_i(\mathcal{M}) \vDash \kappa$.

Each transformation we described in the introduction, which we will go into further details in Section 5, are examples of sound transformation for the explanatory criterion $\langle P, = 0 \rangle$

Now we will assert that transformation results in a simpler representation or equivalently is a simplifying transformation if it is easier to establish the criterion in question in the resultant model, i.e.,

Definition 10. A transformation $\mathcal{F}_i$ is said to be a **simplifying transformation** for a model $\mathcal{M}$ and criterion κ, if $Comp_H(\mathcal{M}, \kappa) > Comp_H(\mathcal{F}_i(\mathcal{M}), \kappa)$, additionally $\mathcal{F}(\mathcal{M})$ is referred to as a simpler representation of $\mathcal{M}$.

One can now see that what we hope to build as explanations are models that are built by repeated application of simplifying but sound transformation. However, there is a big question we have yet to answer, namely, how does one stop the method from generating extremely simple models that are, nonetheless, completely disconnected from the original problem at hand? Effectively this would turn the output of the methods into lies rather than satisfying explanations (comparable to discussions provided by papers like [16]), as it would give the user no further insights into the original planning problem. One could try to avoid this by placing restrictions on the types of models generated by the transformation functions. For example, requirements like the need to share action/state-space or the need to preserve some transition function. However, most of these methods are too limiting and insufficient when we consider cases where there might exist a vocabulary mismatch between the user and the decision-maker [11]. In such cases, one might need to translate the current system representation into forms that are easier for the user to follow and it may be quite hard to establish any form of equivalence between model components of the learned models and the original model. Instead, in this paper, we will define the validity of the explanations in terms of their impact on human expectations about the system. In particular, we will require that the transformed model doesn't satisfy ***any*** explanatory criteria not true in the original model (i.e., the model on which the transformation function was applied). Effectively this would make sure that the transformation does not prevent the user from asking any question they might have about the system's capabilities. This will effectively ensure that the user doesn't place unwarranted trust in the system due to the explanation, a requirement for generating effective posthoc explanations [11]. We will refer to such as *universally sound* transformations for a given model:

Definition 11. A transformation function $\mathcal{F}$ is said to be **universally sound** for a model $\mathcal{M}$, if for any explanatory criterion κ, we have $\mathcal{F}(\mathcal{M}) \vDash \kappa$, then $\mathcal{M} \vDash \kappa$. Equivalently we will refer to the model $\mathcal{F}(\mathcal{M})$ a universally sound representation of $\mathcal{M}$.

All the transformations we will discuss in Section 5 are examples of universally sound transformations. Additionally, we will require that every transformed model presented to the human be explicitly noted to be a simplification of the true model, while noting the exact relationship between the true model and transformed model when possible. Usually when we look at transformations that are generated through syntactic transformations of human readable model descriptions (as in the case of transformations discussed in Section 5), such relations are much easier to note.

It is worth keeping in mind that our objective here is to identify a potential model where the current solution can be contrasted against the foil. One of the goals of the explanatory process would be to help induce a user mental model that is equivalent to this model. However, there could be a component of the explanatory message which also contains explicit information contrasting the two, we will look at such information in Section 3.4.

3.3. Modeling user's computational capabilities

The previous section only provides a cursory discussion of the function capturing the computational burden placed on the human to establish a specific criterion, i.e., the function $Comp_H(\cdot, \cdot)$. An exact characterization of $Comp_H(\cdot, \cdot)$ would be hard, since it would require accurately capturing the inferential capabilities of the human. However, one could still use a number of simpler representations of $Comp_H(\cdot, \cdot)$ to calculate useful representations, some of the possible choices here include

- Using computational model with psychological fidelity - This could correspond methods like, the ones that leverage computational implementation of psychological models like prospect theory [17], use of decision-making algorithms that make use of limited memory [18], use of various models of bounded-rationality including ones from behavioral game theory like the finite nested rational model [19], etc. However, works in these models are still in preliminary stages and we are unaware of any existing techniques that could directly be applied to our problem framework.
- Directly learning $Comp_H$ - Another possibility might be to directly learn the $Comp_H$ function from data collected from human subjects. While we are unaware of any method that can currently learn the function of the form we require, some preliminary work in this direction include [20].
- Using Computational Proxies - While we may not have access to $Comp_H$, we could directly measure the hardness of establishing the explanatory property using exact and sound methods and measuring the time. While this isn't expected to be equal to the actual inferential burden faced by the user, we could use this as an approximation of the exact value. In addition to exact time taken by an automated reasoner, one could also use other measures like the size of description, length of the most likely policy trace, etc.

- Using Human-Subject Studies to Establish Effectiveness of Individual Transformations - Another method might be to not directly learn $Comp_H$, but instead identify any preference use might have on various types of transformation that may be applicable for simplifying a given model. Then one could leverage the preference between the transformations to identify solutions that may be preferred by the user.

In this paper, we will mostly focus on the latter two strategies, wherein we will look at generating simplified model that are simpler with respect to an automated reasoner. We also introduce an ordering between the various transformations to be applied (where the more preferred transformation are applied first). This ordering between the transformation are determined through a user subject study.

3.4. Explanatory witness

Even after providing the model, the user may not be able to reason about the explanatory property on their own and, as such, may require additional assistance. We will refer to such information as *Explanatory Witness*. XAI literature includes many examples of such information, and in fact, many works focus on identifying such information while making implicit assumptions about the model information with which the user is supposed to make sense of this information. In general, from the current literature, we can identify three categories of such information;

1. Proof of explanatory properties: A proof of the property being explained, which in our case can be provided in the minimal model. Though unfortunately, it is very hard to create exact interpretable proofs for most query properties. In general, most of these proofs would be incomplete or abstract in the sense of skipping some steps. An example of such abstract explanatory witnesses would be the use of Q-values to contrast the current choice against alternative (as in the case of [21]). Here the explanatory witness doesn't provide information as to why the current action in a state or the alternative has a specific Q-value.
2. Existential Information: This corresponds to presenting instances of potential solutions (plan/policies) or parts of solutions (a specific execution trace from a given policy) that acts as a demonstration of the property being explained. Once such an example is selected, the example could then be contrasted with the original solution and the information presented to the user.
3. Counterfactual Information: In the final category, the user is provided with examples of counterfactual solutions and even planning models where the property being explained isn't satisfied. The assumption is that these examples would help humans get a better sense of when the property might not be satisfied.

While generating explanations for user studies we will use a sampled policy execution trace as the corresponding Explanatory Witness (a type of existential information). For example, when trying to explain why the cost of a policy may be above some threshold, then one could find a specific execution trace, i.e., a sequence of states, actions and resulting state that can be sampled from the current policy and show that the cost of that trace is above the current threshold. This is a particularly appealing for criteria involving cost. For criteria related to goal reachability, possible explanatory witness include the use of qualitative occupancy frequency (similar to [22]) and for unsolvability one could present the infeasibility of some example paths to the goal (as in the case of [23]).
Section 8 includes a discussion of existing types of explanatory witnesses studied in the literature.

3.5. Generating explanations

With these basic definitions in place, we can describe the actual approach to generating the explanations.

The first component, we will need to consider is a way to identify an explanatory criteria. It should be clear that for any given explanatory query, there might be multiple possible explanatory criteria, especially ones that use different optimization metrics. However, given the fact that there exists a strict ordering between the optimization metrics, we will generally prefer explanatory criteria related to the criterion with the higher preference. We will denote the fact that the explanatory criteria κ_i is preferred over a property κ_j, by using the notation $\kappa_i \prec \kappa_i$. In our setting, a reachability related criteria is preferred over those related to cost. This means that one could test whether we can use goal reaching probability of the fact policy to derive an explanatory criteria, if not switch to a cost based one (again the threshold is set by the cost function associated with the fact policy).

Once the most preferred explanatory criterion is identified, our next challenge is to determine the most simplified model where the criterion can still be established.

Definition 12. Given a model $\mathcal{M}^{R^*}$, a set of universally sound transformation functions $\mathbb{F} = \{\mathcal{F}_1, ..., \mathcal{F}_k\}$, the problem of generating a **minimal explanatory model** for a given explanatory criterion κ, corresponds to the problem of finding a sequence of transformation $\mathcal{T}_{min}$ which results in a simplified representation $\mathcal{M}$ that is sound for κ and there exists no other sequence of model transformations that will result in a simpler yet sound representation, i.e.,

$$\nexists \mathcal{T}' \; Comp_H(\mathcal{T}_{min}(\mathcal{M}^{R^*}), \kappa) > Comp_H(\mathcal{T}'(\mathcal{M}^{R^*}), \kappa)$$

The new model $\mathcal{T}_{min}(\mathcal{M}^{R^*})$ will form the basis of the explanation.

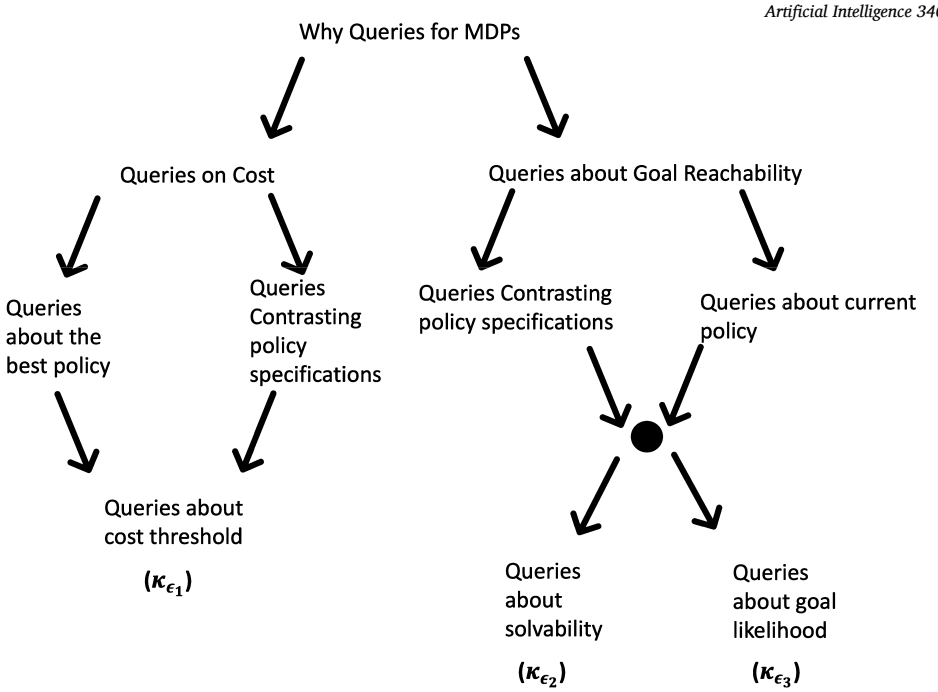


Fig. 3. A hierarchy of *Why* questions that can be asked by a user trying to understand decisions generated by an automated decision-maker using an undiscounted MDP that meets the P Assumption.

The basic algorithm for generating such minimal explanations is given in Algorithm 1, which correspond to an exhaustive search over all possible transformations and then returning the transformation sequence, explanatory criterion pair that meets the requirements provided in Definition 12.

Algorithm 1 Basic Search.

procedure SEARCH
  Input: $\mathcal{M}^{R^*}, \mathbb{F}, \kappa$
  Output: The transformed model $\mathcal{M}$
  Procedure:
  $\mathcal{T}_{\text{best}} = \langle\rangle$
  Curr_Minimal_Effort $\leftarrow \infty$
  for Each subsequence $\mathcal{T}'$ of $\mathbb{F}$ **do**
    Comp_Effort $\leftarrow Comp_H(\mathcal{T}'(\mathcal{M}^{R^*}), \kappa)$
    if Curr_Minimal_Effort $>$ Comp_Effort **then**
      $\mathcal{T}_{\text{best}} = \mathcal{T}'$
  return $\mathcal{T}_{\text{best}}(\mathcal{M}^{R^*})$

Note that the above algorithm is an extremely computationally expensive one. The search space is exponential over the number of transformations possible and each search node evaluation in our case consists of solving a planning problem. However, as we will see in Section 6 by making commitments on the types of transformation and model classes, we can make use of a significantly more efficient search algorithm.

As for generating the explanatory witness, once we have identified the minimal explanatory model, one can use any of the methods outlined in Section 3.4 to generate an appropriate explanatory witness.

4. Queries and explanations

Fig. 3 presents the hierarchy of questions possible in this problem setting organized into two categories. Now for each type, we further list two possible subcategories. The first category corresponds to queries exclusively about the current best solution provided by the system and the second corresponds to queries that contrast the current policy with an alternative the human had in their mind. In the former, the user would want to understand why the solution is worse off than what they were expecting (say in terms of goal reachability or cost), and in the latter, the user would want to know how the current policy compares against the alternative they had in mind. In the end, both categories can be mapped into a question that can be resolved by comparing a policy against a threshold. In the case of the former category, the threshold directly comes from the user's question and the system's policy is compared against

it (for example, the kind of questions that might fall into this category would include instances like *Why does the policy cost more than X?* or *Why is the likelihood of reaching the goal less than Y?*), while in the latter the alternative posed by the user is compared against the cost or goal reachability of the current policy. In the end, the query hierarchy concretizes into three specific query types.

Each query type is characterized by a specific explanatory criteria,

1. κ_{ϵ_1} is satisfied when there exists no policy that has a cost less than a certain threshold for the initial state under the model
2. κ_{ϵ_2} is satisfied when no policy achieves the goal from the initial state with non-zero probability
3. κ_{ϵ_3} is satisfied when there exists no policy whose probability of achieving the goal from the initial state is above a certain threshold.

These criterion classes correspond to a large set of specific explanatory criteria and the exact threshold is determined by the specific query. For a specific model $\mathcal{M}$ and a threshold X, we will denote the fact that an instance of the criterion class κ is satisfied for the threshold as $\mathcal{M} \models \kappa(X)$ (in the case of κ_{ϵ_2} X is limited to 0).

In the categorization, we have split the queries related to goal reachability into two distinct categories, even though, one could specify it as a comparison to a threshold. The reason why we chose not to do that, is two-fold;

1. Not only are queries about unsolvability a natural and commonly studied explanatory query type [24], they are also a lot more accessible and more likely to be used by non-experts who may not be aware of or comfortable with the exact probability associated with the planning problem.
2. One can apply many more explanatory techniques to answer questions related to solvability, which is not necessarily applicable to questions that use direct comparison to probability threshold. For example, as we will see, the use of determinization or even the use of models with purely qualitative non-determinism is possible to answer queries related to solvability but doesn't apply to queries that explicitly use probabilistic thresholds.

Any possible contrastive query, which can be answered by a decision-making system, would correspond to one of these explanatory criteria. Following the conventions generally used in MDP solution strategies, we will establish a strict ordering between the criteria, namely, $\kappa_{\epsilon_1} \prec \kappa_{\epsilon_2} \prec \kappa_{\epsilon_3}$.

5. Model transformation classes

In this section, we will look at some specific transformation classes. The classes were selected based on the fact that some restricted forms of these transformations were already studied in the wider XAI literature. In particular, Section 5.1 presents projective state abstraction ($\mathcal{F}_{F\setminus\Lambda}$); Section 5.2 presents determinization ($\mathcal{F}_{\Delta}$); Section 5.3 covers problem decomposition ($\mathcal{F}_{\mathbb{D}}$) and Section 5.4 local approximation ($\mathcal{F}_{\mathcal{L}}$).

5.1. State abstractions

The first transformation we could make use of is state abstraction that can help reduce the underlying state space of the MDP [25], which has been used for explaining deterministic plans [26] and to summarize policies as in [27]. In particular, we will consider state aggregations that replace a set of states in the underlying MDP $\mathcal{M}^{R^*}$ with a single state [28]. *In Example 1, it will be a state abstraction that will let us ignore most of the state variables, including those related to motion-level constraints.* We will define state abstraction as

Definition 13. For a given model $\mathcal{M} = \langle S, A, I, P, C, G\rangle$, $\widetilde{\mathcal{M}} = \langle \widetilde{S}, \widetilde{A}, \widetilde{I}, \widetilde{P}, \widetilde{C}, \widetilde{G}\rangle$ is said to be an abstraction of $\mathcal{M}$, if there exists a surjective mapping from states and actions in $\mathcal{M}$ to $\widetilde{\mathcal{M}}$ (where $\mathcal{F}$ be the mapping), then $\forall s_1, s_2 \in S$ and for any $a \in A$, if $P(s_1, a, s_2) > 0$, then $P(\mathcal{F}(s_1), \mathcal{F}(a), \mathcal{F}(s_2)) > 0$.

While state abstraction has been a popular topic of investigation in MDP planning/RL topics. We are particularly interested in its use to create sound representation for all the queries. Since we are looking at symbolic representation, it would be useful to have methods that create abstractions that are valid symbolic models. Specifically, we can make use of syntactic projections of the model descriptions, which have previously been used for generating heuristics [29].

5.1.1. State abstractions through syntactic projection

We will define the transformation as

Definition 14. For a given model description $\mathcal{M}^D = \langle F^D, A^D, I^D, G^D\rangle$ and a set of propositions $\Lambda \subseteq F^D$ the syntactic projection (represented as a function $\mathcal{F}_{F\setminus\Lambda}$) results in a new model description $\mathcal{F}_{F\setminus\Lambda}(\mathcal{M}^D) = \langle F^D \setminus \Lambda, A^{\mathcal{F}_{F\setminus\Lambda}}, I^D \setminus \Lambda, G^D \setminus \Lambda\rangle$. Where the new actions $A^{\mathcal{F}_{F\setminus\Lambda}}$ is given as follows: for each $a_i \in A^D$, there existing a corresponding action $\mathcal{F}_{F\setminus\Lambda}(a_i) = \langle \text{prec}_i \setminus \Lambda\rangle$ (To simplify discussion we will be overloading the notation $\mathcal{F}_{F\setminus\Lambda}$ to stand for any mapping from the components of the original model or description to those in the new model)

$$\mathcal{F}_{F\setminus\Lambda}(E_i) = \{\langle \text{add}_i^j \setminus \Lambda, \text{del}_i^j \setminus \Lambda, p', c'\rangle \mid \langle \text{add}_i^j, \text{del}_i^j, p_i^j, c_j^i\rangle \in E_i\} \tag{1}$$

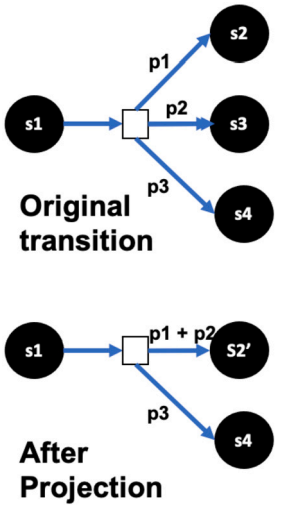


Fig. 4. A diagrammatic representation of the transformation induced by the abstraction function on the transition probabilities. Here the projection operation causes the states $s2$ and $s3$ to collapse into a single state $s2'$.

Where the new probability of the effect p' is given as

$$\begin{aligned} p' = & \sum_{\langle \text{add}_i^k, \text{del}_i^k, p_i^k, c_i^k \rangle \in \text{E}_i \text{ ,where } \text{add}_i^k \setminus \Lambda = \text{add}_i^j \setminus \Lambda \text{ and } \text{del}_i^k \setminus \Lambda = \text{del}_i^j \setminus \Lambda} p_i^k \\ c' = & \, min\{c_i^k | \langle \text{add}_i^k, \text{del}_i^k, p_i^k, c_i^k \rangle \in \text{E}_i \\ & \text{,where } \text{add}_i^k \setminus \Lambda = \text{add}_i^j \setminus \Lambda \text{ and } \text{del}_i^k \setminus \Lambda = \text{del}_i^j \} \} \end{aligned} \tag{2}$$

So effectively the transformation removes every appearance of a fluent being projected out from the description, and if this results in mutually exclusive effects to be duplicated (including being empty), then only one version of the effect is included in the updated model, the probability of this effect becomes the sum of the individual effects and the cost of this effect becomes the minimum of the original effects. Fig. 4, present a visual representation of the transformation.

Proposition 1. *One could create the transformed model description $\mathcal{F}_{F\setminus\Lambda}(\mathcal{M}^D)$, in time polynomial over the size of the original model description and the number of factors in Λ.*

This complexity should be clear from the fact that one could do it by making two passes through the domain model. Once to remove the fluents in Λ and the second to merge duplicate effects. Now we will see that the MDP corresponding to updated model description $\mathcal{F}_{F\setminus\Lambda}(\mathcal{M}^D)$, is in fact a valid state abstraction.

Proposition 2. *The model $\mathcal{M}^{\mathcal{F}_{F\setminus\Lambda}}$ represented by $\mathcal{F}_{F\setminus\Lambda}(\mathcal{M}^D)$ is a state abstraction (per Definition 13) of the model $\mathcal{M}$. Additionally, for any states $i, j \in S$ for MDP $\mathcal{M}$, then $P(i, a_{a_i}^{mdp}, j) \leq P^{\mathcal{F}_{F\setminus\Lambda}}(\mathcal{F}_{F\setminus\Lambda}(i), a_{\mathcal{F}_{F\setminus\Lambda}(a_i)}^{mdp}, \mathcal{F}_{F\setminus\Lambda}(j))$ and $C(i, a, j) \geq C^{\mathcal{F}_{F\setminus\Lambda}}(\mathcal{F}_{F\setminus\Lambda}(i), a_{\mathcal{F}_{F\setminus\Lambda}(a_i)}^{mdp}, \mathcal{F}_{F\setminus\Lambda}(j))$, for states $\mathcal{F}_{F\setminus\Lambda}(i)$ and $\mathcal{F}_{F\setminus\Lambda}(j)$ in $\mathcal{M}^{\mathcal{F}_{F\setminus\Lambda}}$.*

Proof Sketch. First off $\mathcal{F}_{F\setminus\Lambda}(\mathcal{M}^D)$ is a well formed model description, so there exists a unique model $\mathcal{M}^{\mathcal{F}_{F\setminus\Lambda}}$ induced by it. The surjective mapping from state space S of $\mathcal{M}$ to $S^{\mathcal{F}_{F\setminus\Lambda}}$ (Overloading the notation a bit, we will use $\mathcal{F}_{F\setminus\Lambda}$ to also capture this mapping) is given as

$$\mathcal{F}_{F\setminus\Lambda}(i) = \hat{i} \text{ if } \hat{s}_i = s_i \setminus \Lambda$$

It should be easy to verify that this is in fact a surjective function. The relationship between the probability functions and cost functions comes directly as a result of the transformation. □

We can use the abstraction function to also define a relation among the states in the model description, such that $s_i \sim_{\mathcal{F}_{F\setminus\Lambda}} s_j$, if $\mathcal{F}_{F\setminus\Lambda}(s_i) = \mathcal{F}_{F\setminus\Lambda}(s_j)$. Note that this is an equivalent relation and thus helps partition the state space into disjoint sets that map into the same abstract. If $\hat{s}_i$ is an abstract state for $\mathcal{F}_{F\setminus\Lambda}(\mathcal{M})$, then we will use the notation $\mathcal{F}_{F\setminus\Lambda}^{-}1(\hat{s}_i)$ to capture the quotient set for the relation $\sim_{\mathcal{F}_{F\setminus\Lambda}}$ that maps the state from the concrete model into the abstract state $\hat{s}_i$, i.e.,

$$\mathcal{F}^{-1}_{F\setminus\Lambda}(s) = \{s' | \mathcal{F}_{F\setminus\Lambda}(s') = s\}$$

Unless specified otherwise, the cost function for the concrete model are denoted as functions over just state indexes (for example $J(i)$), while that for the abstract model it is denoted with states where an abstraction function has been applied (for example $J(\mathcal{F}_{F\setminus\Lambda}(i))$).

Next we will detail some simple properties of the model and demonstrate the central properties we can use for explanation.

This takes us to the next result

Proposition 3. *Probability of a trace (a sequence of action control state tuples) from a given state to a goal state is either preserved or increases over the transformation, i.e., for a trace* $t = \langle s_1, a_1, ..., s_k \rangle$, $P_{\mathcal{F}_{F\setminus\Lambda}(\mathcal{M})}(t) \geq P_{\mathcal{M}}(t)$.

The result follows from earlier proposition.

Proposition 4. *Probability of an action causing a transition from an abstract state* $\hat{i}$ *to another abstract state* $\hat{j}$*, is equal to the sum of the probability of transitions from one of the states in* $\mathcal{F}^{-1}_{F\setminus\Lambda}(\hat{i})$ *to all the states in* $\mathcal{F}^{-1}_{F\setminus\Lambda}(\hat{j})$ *under that action.*

The result follows from the fact that given the declaration method, all the states that merged must have had similar effects, so the choice of the state $\mathcal{F}^{-1}_{F\setminus\Lambda}(\hat{i})$ doesn't matter. The second part of the proposition follows from the transformation itself.

With this transformation in hand, we can show that the syntactic transformation results in a state aggregation. We can now show the optimal cost function in the new model is lower than that of the original model.

Lemma 1. *For every state* i *in the model* $\mathcal{M}$, $J^*(i) \geq J^*_{\mathcal{F}_{F\setminus\Lambda}}(\mathcal{F}_{F\setminus\Lambda}(i))$, *where* $J^*_{\mathcal{F}_{F\setminus\Lambda}}$ *is the optimal cost function for* $\mathcal{M}^{\mathcal{F}_{F\setminus\Lambda}}$.

Proof Sketch. We can show this by initializing a cost function for the abstract model $J_{\mathcal{F}_{F\setminus\Lambda}}$ with cost from J^*, such that for any abstract state $\hat{i}$ as $J_{\mathcal{F}_{F\setminus\Lambda}}(\hat{i}) = \min_{i\in\mathcal{F}^{-1}_{F\setminus\Lambda}(\hat{i})}(J^*(i))$. If we now apply a bellman operator T, given Propositions 2 and 4, we will have

$$T J_{\mathcal{F}_{F\setminus\Lambda}}(\hat{i}) \leq J_{\mathcal{F}_{F\setminus\Lambda}}(\hat{i})$$

For P condition the bellman operator is still monotonic function [3], and converges to the optimal cost function. Thus the optimal cost function for $\mathcal{F}_{F\setminus\Lambda}(i)$ must be less than or equal to $J^*(i)$. □

Next, in regard to the probability, we can establish that

Lemma 2. *For every state* i *in the model* $\mathcal{M}$, $P^*(i) \leq P^*_{\mathcal{F}_{F\setminus\Lambda}}(\mathcal{F}_{F\setminus\Lambda}(i)))$, *where* $P^*_{\mathcal{F}_{F\setminus\Lambda}}$ *is the maxprob probability for the model* $\mathcal{M}^{\mathcal{F}_{F\setminus\Lambda}}$.

This result directly follows from Proposition 3 (a similar result was also established in [29]). With these two propositions we have established that state abstraction does in fact result in new models that underapproximated cost and overapproximates reachability. Thus establishing that *the syntactic projection described here results in a valid state abstraction per Definition 13 and the resultant transformation is sound with respect to all three explanatory criteria.*

Theorem 1. *The domain description transformation* $\mathcal{F}_{F\setminus\Lambda}$ *corresponds to a universally sound transformation for any model that can be represented by a model description of the form* $\mathcal{M}^D = \langle F^D, A^D, I^D, G^D, C^D \rangle$.

In Example 1, consider the original probabilistic effects of the mixing action, which says after mixing, you might have cake batter with a probability of 0.5, a cake batter with bubbles in it with a probability of 0.25, and with a probability of 0.25, no cake batter at all.

```
(probabilistic 1/2 (and (has-cake-batter))
      1/4 (and (has-cake-batter)
      (cake-batter-has-bubbles)))
```

After projecting out (cake-batter-has-bubbles) you get an effect that says the probability of having cake batter is 0.75

```
(probabilistic 3/4 (and (has-cake-batter)))
```

5.2. Problem determinization

Note that the abstraction operation creates models with probabilistic effects unless the effects merge into a single effect. At least for some of the queries, the system could generate valid responses while using an optimistic determinization of the model that ignores all stochasticity of the model. *For example, in Example 1 even if there were no undesired side-effects due to stochasticity (the 0.25 probability*

of the mixing action not forming a cake batter), the mix action could still not be executed as it has a missing precondition (has-cake-whisk). One possible way to create such optimistic determinization could be to generate a new model where action with multiple stochastic effects is turned into multiple actions with deterministic effects. Determinization belongs to a larger class of transformations (which we will refer to as problem class simplification) that transforms the original problem to simpler decision-making problems for the sake of generating explanations. While there are some instances of problem class simplification for explanations for multi-objective explanations (cf. [30]), we are unaware of any direct use of determinization to simplify explanations.

5.2.1. All outcome determinization

Definition 15. $\mathcal{M}^D = \langle F^D, A^D, I^D, G^D \rangle$ a determinization is presented by an operator $\mathcal{F}_\Delta$ and generates a new model of the form $\mathcal{F}_\Delta(\mathcal{M}^D) = \langle F^D, \mathcal{F}_\Delta(A^D), I^D, G^D \rangle$. Where for every $a_i = \langle prec_i, \{e_i^1, ..., e_i^k\} \rangle$, there exists k actions in $\mathcal{F}_\Delta(A^D)$ such that $a_i^j = \langle prec_i, add_i^j, del_i^j, 1, c_i^j \rangle$, i.e., it generate the j^{th} effect with probability 1.

Such determinization operations are sometimes referred to as all outcome determinization [31]. Which is also closely connected to hindsight optimization techniques that have been studied in multiple fields including control theory [32]. *The transformation could be used for both cost criterion (κ_{ϵ_1}) and solvability criterion (κ_{ϵ_2}).* Also note that for the class of models considered in this paper the transformation of the model description can be carried out effectively.

Proposition 5. *The transformed model description $\mathcal{F}_\Delta(\mathcal{M}^D)$ can be created by performing a single pass through the original description $\mathcal{M}^D$ and the maximum number of actions in $\mathcal{F}_\Delta(\mathcal{M}^D)$ is upper-bounded by $K \times |A|$, where K is the maximum number of mutually exclusive effects that can appear in an action definition in A.*

The fundamental property of this determinization we can use here is the following

Proposition 6. *Let $\tau = \langle I, a_1, ..., a_k, g \rangle$ be the sequence of symbolic states and actions corresponding to a trace from the initial state to a goal state $g \in G$ with non-zero probability for the model defined by the description $\mathcal{M}^D$. Then $A(\tau)$ (the sequence of actions appearing in τ) is a valid plan for the deterministic planning model $\mathcal{F}_\Delta$. That is when the sequence $A(\tau)$ is executed in I it will take you to the state g.*

This property directly comes from the form of the transformation is widely established in the determinization literature.

Lemma 3. *The cost of best possible plan in $\mathcal{F}_\Delta(\mathcal{M}^D)$ is guaranteed to be less than or equal to $J^*(I)$ for the model $\mathcal{M}$ corresponding to $\mathcal{M}^D$*

After all the lowest cost trace that can be sampled from any policy is a plan in the determinized model. As the cost of the policy should be higher than the cost of its lowest cost trace, it should be higher than the cost of the optimal plan in $\mathcal{F}_\Delta(\mathcal{M}^D)$.

Lemma 4. *If the problem $\mathcal{F}_\Delta(\mathcal{M}^D)$ is unsolvable then $P^*(I) = 0$.*

This follows directly from Proposition 6.

With Lemma 3 and 4 in place should be clear that this transformation could be used for both cost criteria (κ_{ϵ_1}) and solvability criteria (κ_{ϵ_2}). One can actually in fact make a stronger claim and show that it is even sound for (κ_{ϵ_3}).

Theorem 2. *The domain description transformation $\mathcal{F}_\Delta$ corresponds to a universally sound transformation for any model that can be represented by a model description of the form $\mathcal{M}^D = \langle F^D, A^D, I^D, G^D, C^D \rangle$*

Proof. As mentioned earlier the soundness of the first two properties directly follow the earlier Lemmas. To see the soundness with respect to κ_{ϵ_3}, note that $\mathcal{F}_\Delta(\mathcal{M}^D)$ is a well formed PPDDL model description, but one that only allows for deterministic transition. So in this model the maxprob probability must be either 1 or 0. This means that the only threshold for which κ_{ϵ_3} can be meaningfully established in the updated model is for threshold 0 (discounting 1 as it as κ_{ϵ_3} for threshold 1 a tautological statement). κ_{ϵ_3} limited to threshold 0 corresponds to κ_{ϵ_2}, which per our earlier discussion the transformation $\mathcal{F}_\Delta$ is already sound for. Thus establishing the fact that $\mathcal{F}_\Delta$ is universally sound. □

With regards to Example 1, such transformation will allow us to ignore the probability of mix action failure when giving the explanation. In particular, we will have two new mixing actions. One with effect (has-cake-batter) and the other with effect (and (has-cake-batter) (cake-batter-has-bubbles)).

5.3. Problem decomposition

Even after applying all the above transformations, the plans generated from the resultant model could be extremely long. One way to address would be to decompose the original task into smaller subtasks. *In Example 1, rather than talking about the problem of*

baking cake it can focus just on explaining the subproblem of making the cake batter. In this section, we introduce problem decomposition with respect to an initial state that focuses on subproblems that reuse states and actions of the original problem and is either cheaper or it is easier to achieve the goal

Definition 16. Given an atomic MDP $\mathcal{M} = \langle S, A, P, I, C, G\rangle$ with an optimal cost J^*, an MDP $\mathcal{M}' = \langle S, A, P, I, C, G'\rangle$ with an optimal cost J', is said to be a subproblem if it is guaranteed that $J'(I) \leq J^*(I)$ (where the decomposition is called cost based decomposition) or $P'(I) \leq P^*(I)$ (where the decomposition is called reachability-based decomposition).

For queries related to κ_{ϵ_1} if we can establish that the cost of the subproblem under cost-based decomposition is greater than the limit, then that automatically establishes that the original problem should be worse. For queries related to reachability, we can look for similar arguments for a subproblem under reachability-based decomposition. Now the question is how can we find such decompositions.

5.3.1. Subgoals

For goal-directed problems with an initial state, a natural idea could be subgoals, where subgoals are representations of intermediate states that the policy may need to achieve before getting to the final goal. A specific subgoal that could be useful here is landmarks, which were recently adapted for SSPs by [33]. These are propositional facts that need to be satisfied by any path from the initial state to goal with non-zero probability. While that paper introduced these facts as a way to summarize policies, we use them to decompose problems and form simpler problems. Such landmarks can be automatically extracted from the model description $\mathcal{M}^{D^*}$. For unsolvable problems, such landmarks could also be generated from abstractions of the problem where the goal is reachable (cf. [24]). In Example 1, has-cake-batter is a landmark for the goal bake-cake. *A problem decomposition using landmarks as the new goal is both a cost and reachability decomposition and is sound for all three explanatory criteria.*

Lemma 5. *If $f \in F^D$ is a fact landmark for the MDP corresponding $\mathcal{M}^D = \langle F^D, A^D, I^D, G^D\rangle$, then the model $\mathcal{F}_{\mathbb{D}}(\mathcal{M}^D) = \langle F^D, A^D, I^D, f\rangle$ is both a cost-based decomposition and probability-based decomposition. In other words, $J^*_{\mathcal{F}_{\mathbb{D}}}(I) \leq J^*(I)$ and $P^*_{\mathcal{F}_{\mathbb{D}}}(I) \leq P^*(I)$, where $J^*_{\mathcal{F}_{\mathbb{D}}}$ and $P^*_{\mathcal{F}_{\mathbb{D}}}$ are the optimal cost function and MAXPROB probabilities for the model corresponding to the description $\mathcal{F}_{\mathbb{D}}(\mathcal{M}^D)$.*

Proof Sketch. Let $\tau = \langle I, a_1, ..., a_k, g\rangle$ be the sequence of symbolic states and actions corresponding to a trace from the initial state to a goal state $g \in G$ with non-zero probability for the model defined by the description $\mathcal{M}^D$. From the definition of landmarks, we know that there must be state s_f in the sequence τ such that $f \in s_f$. Therefore any trace with nonzero probability can be decomposed into a prefix corresponding to the sequence that leads to a landmark state and then the sequence from landmark state to the final goal. Additionally we can see that the probability of this trace prefix must be greater than or equal to the probability of the full trace. Thus for all the traces sampled from a given policy the total probability of getting to all landmarks states must be higher than that of reaching the final goal. We can use a similar reasoning to show that the total expected cost of reaching the landmark states must be less than or equal to the cost of reaching the final goal state. □

The above lemma establishes the fact that problem decomposition through landmarks underapproximates costs and overapproximates reachability, thus its valid for all three explanatory criteria.

Theorem 3. *The domain description transformation $\mathcal{F}_{\mathbb{D}}$ corresponds to a universally sound transformation for any model that can be represented by a model description of the form $\mathcal{M}^D = \langle F^D, A^D, I^D, G^D, C^D\rangle$*

Now coming to the complexity of the transformation, once given a fact landmark the model transformation can be performed quite effectively.

Proposition 7. *For a given fact landmark f and a model description $\mathcal{M}^D$, the domain description transformation $\mathcal{F}_{\mathbb{D}}$ can be performed in constant time.*

This is because you just need to replace the goal description in $\mathcal{M}^D$ to form the transformed description. Now coming to the complexity of generating a landmark, the problem of finding landmark in general is PSPACE-Complete [34]. However, there are classes of landmarks which can be generated more effectively. One class of such landmarks is the causal landmarks which can be generated in time polynomial to the size of the model description [35].

For the example, after this transformation the goal condition of the model description will be changed from (bake-cake) to (has-cake-batter).

5.4. Local approximation

The next transformation we will consider is that of local approximations. Local approximations [36], have been successfully used in the context of explaining machine learning decisions. The basic premise being that the model being used for explanation need not

accurately reflect the entire decision-space, but only the parts that are relevant to the current query. One could translate the same intuition into the sequential decision-making settings, and look at creating simpler representations of the model that focus only on a part of the transition system. *Revisiting Example 1, through local approximation we can establish the facts that the robot doesn't need to talk about any of its skills unrelated to cooking or baking cakes. Based on the fact that it is in a home, it can also figure out that it can skip providing any information to the user about it's ability to use industrial mixers to make cake batter.*

We can describe a local approximation (represented by the transformation $\mathcal{F}_{\mathcal{L}}$) for a set of states $\hat{S}$ and a set of actions $\hat{A}$ as a function that generates a new model that conserves the transition probabilities and cost functions related to states and actions that appear in $\hat{S}$ and $\hat{A}$.

Definition 17. For a given model $\mathcal{M} = \langle S, A, P, C, I, G\rangle$ and a subset of states and actions $\hat{S} \subseteq S$ and $\hat{A} \subseteq A$, a well formed MDP model $\mathcal{M}' = \langle S', A', P', C', I', G'\rangle$, is said to be a **local approximation** (denoted as $\mathcal{F}_{\mathcal{L}}(\mathcal{M}) = \mathcal{M}'$) if there exists a mapping from $\hat{S}$ to S' and from $\hat{A}$ to A' (with a slight abuse of notation we will use $f_{\mathcal{L}}$ for both the state to state and action to action mapping), such that for any given states $i, j \in \hat{S}$ and an action $a \in \hat{A}$, we have $P(i,a,j) = P'(f_{\mathcal{L}}(i), f_{\mathcal{L}}(a)), f_{\mathcal{L}}(j))$, $C(i,a,j) = C'(f_{\mathcal{L}}(i), f_{\mathcal{L}}(a), f_{\mathcal{L}}(j))$ and there exists a mapping from $\hat{S} \cap G$ to G'.

This is a very permissive definition and not all instantiation of the transformation would necessarily lead to transformations that would lead us to ones that may preserve explanatory criteria. So before, we consider an instance of the transformation, let us consider a specific subclass of the transformation, one that focuses on cases where the subset of states and actions consider form a closed transition system, i.e., all states reachable from $\hat{S}$ under the policy is a subset of $\hat{S}$

Definition 18. A set of states $\hat{S}$ and set of actions $\hat{A}$, is said to be **closed for a policy** π, if

1. Execution policy in $\hat{S}$, will never lead to a transition to a state outside the set, i.e., $\forall j \in S \setminus \hat{S} \nexists i \in \hat{S}$ and $a \in \hat{a}$ such that $P(i,a,j) > 0$
2. The set $\hat{A}$ covers all actions assigned to states in $\hat{S}$, under the policy π, i.e., $\forall i \in \hat{S}, \pi(i) \in \hat{A}$

We will now see how any local transformation defined over a closed set, will result in universally sound transformation. More formally, we can state this as;

Proposition 8. *If the sets $\hat{S}$ and $\hat{A}$ are closed for a policy π, then $\forall i \in \hat{S}, J'^*(f_{\mathcal{L}}(i)) \leq J^{\pi}(i)$ and $P'^*(f_{\mathcal{L}}(i)) \geq P^*(i)$, where J'^* and P'^* are the optimal cost function and the maxprob probability for the model $\mathcal{F}_{\mathcal{L}}(\mathcal{M})$.*

Proof sketch. This follows directly from the fact that the transformation introduces no new transitions for states and actions that are part of the set $\hat{A}$ and $\hat{S}$. Moreover the transformation conserves both cost and probabilities for those states and actions. This means there exists multiple policies for $\mathcal{F}_{\mathcal{L}}(\mathcal{M})$ which has the same value and probability for states in $\hat{S}$ as the original policy π. This means the optimal policy and maxprob policy should lead to policy that are more cheaper and with higher likelihood of getting the goal. Note that this doesn't need to be equal as the local approximations allows for the fact that there may be new actions now applicable in the states that are part of $\hat{S}$. □

Proposition 8 ensures that local approximation results in models that underapproximates costs and overapproximates reachability and we can now state the primary theorem related to $\mathcal{F}_{\mathcal{L}}$.

Theorem 4. *The transformation $\mathcal{F}_{\mathcal{L}}$ for the given sets $\hat{S}$ and $\hat{A}$ is a universally sound transformation for a model $\mathcal{M}$, if,*

1. *$\hat{S}$ and $\hat{A}$ are closed with respect to an optimal policy and a MAXPROB policy.*
2. *$I \in \hat{S}$ and $\mathcal{F}_{\mathcal{L}}(I) = I'$, where I' is the new initial state in the model $\mathcal{F}_{\mathcal{L}}(\mathcal{M})$.*

The fact that the initial state is in $\hat{S}$ (and its corresponding image remains the new initial state) and that it satisfies the requirement for Proposition 8 means that all three explanation criteria κ_{ϵ_1}, κ_{ϵ_2}, and κ_{ϵ_3}.

5.4.1. Reachability analysis for local approximation

One way to create a local approximation is to perform a reachability analysis to remove actions and fluents guaranteed to be not reachable from the initial state. To identify the non-reachable fluents, we will consider the delete-relaxation of the all outcome determinization of the original model and try to identify the reachable fluents and actions by building a relaxed planning graph [37]. The fluents and actions not present in the planning graph are removed from the model description. We will refer to the model description formed through this procedure as $\mathcal{F}^{+}_{\mathcal{L}}(\mathcal{M}^D)$, where + denotes the fact that the local approximation relies on a delete relaxation of the model. First thing to note is that $\mathcal{F}^{+}_{\mathcal{L}}$, can be formed rather efficiently. In particular, we have

Proposition 9. *The new model description $\mathcal{F}^{+}_{\mathcal{L}}(\mathcal{M}^D)$ can be created in time polynomial to the size of the original model description $\mathcal{M}^D$.*

Proof. This follows from the fact that an all outcome determinization can be generated in polynomial time. The formation of the relaxed planning graph and subsequent identification of unreachable fluents and actions can again be performed in time polynomial to the size of the determinized model. With the fluents and actions to be removed identified, one can form the model description $\mathcal{F}^+_{\mathcal{L}}(\mathcal{M}^D)$ again in polynomial time. □

Now the fact we need to show is that this model description transformation actually correspond to a local approximation for a certain subset of states and actions and moreover this approximation meets the requirements for Theorem 4. In particularly we will see that the transformation will create a local approximation defined over the original MDP where the subset of states and actions considered correspond to set of states that can be formed by the remaining fluent and remaining actions and it is in fact closed. More formally, we can state

Proposition 10. *Let $\mathcal{M}$ be the model corresponding to the description $\mathcal{M}^D$ and let $\mathcal{F}^+_{\mathcal{L}}(\mathcal{M}^D) = \langle \hat{F}^D, \hat{A}^D, I^D, G^D, C^D \rangle$ be the newly formed transformed model such that, $\hat{F}^D \subseteq F^D$ and $\hat{A}^D \subseteq A^D$. Then the MDP $\mathcal{M}'$ corresponding to the description $\mathcal{F}^+_{\mathcal{L}}(\mathcal{M}^D)$ is a local approximation of the model $\mathcal{M}$, defined over the state and action subset $\hat{S}$ and $\hat{A}$, such that*

1. *$\hat{S}$ corresponds to the state defined by $2^{|\hat{F}^D|}$ and $\hat{A}$ correspond to actions in $\hat{A}^D$ (both of which are subsets of S and A for the model $\mathcal{M}$)*
2. *$\hat{S}$ and $\hat{A}$ meets the requirements specified in Theorem 4*

Proof Sketch. Note that the mapping here is an identity mapping from states and actions in $\hat{S}$ and $\hat{A}$ to those in the model $\mathcal{M}'$. From the construction of the relaxed planning graph, every fluent that is true in initial state will be conserved and thus I must be part of $\hat{S}$. The fact that $\hat{S}$ and $\hat{A}$ are closed comes from the fact that we are considering a delete relaxation of an all outcome determinization. This means if the process removes a state from consideration (i.e. removes a fluent f that is part of the state), then there exists no non-zero probability trace that can reach that state from the initial state. Thus the states are closed under any policy not just optimal or MAXPROB ones. Similarly for actions, the procedure will never remove any action that is applicable for a state that is part of $\hat{S}$ and thus must be closed. □

Proposition 10 thus establishes the fact that $\mathcal{F}^+_{\mathcal{L}}$ also corresponds to a universally sound transformation.

With regards to Example 1, such transformation will allow us to ignore an action mix-with-industrial-mixer if it has a precondition (in-cake-factory), which can never be made true in this case. As such, the delete-relaxed graph will never allow for the application of this action.

6. Using stratified model transformation search to compute explanations

Now let us return back to the question of how to effectively generate the transformation sequences. Even if we limit our transformation classes to the one we discussed in Section 5, a search for the minimal transformation sequence using Algorithm 1 would be an expensive problem. For one the search space could be large and even if one were to introduce a more informed version of the search any search heuristic we employ will have to compute the effect of a transformation on the computational hardness of the problem. A more useful approach may be to set up a hard priority between the transformation classes. If we are able to set a primary transformation class, then we can directly try to find the maximal number of transformations we can apply from that class. The other transformations need only be considered to the degree that they can be applied to the models generated by applying a maximal sequence of primary transformations possible. From a computational point of view, an excellent candidate for primary transformation might be state abstractions. After all, removing each binary state fluent always reduces the state space by half. Among the most abstract models that support the given explanatory query, we can look at applying the other transformations provided they can conserve the criterion being established.

However, one could also additionally exploit the specifics of the transformations to get additional improvement in search. Starting with state abstraction transformation ($\mathcal{F}_{F\setminus\cdot}$), there are two immediate properties we can exploit, namely the fact that the transformation is *commutative* and *compositional*, or more formally

Proposition 11. *For any model $\mathcal{M}^D$ and for any proposition set $\hat{F}_1 \subseteq F$ and $\hat{F}_2 \subseteq F$, we have*

1. *$\mathcal{F}_{F\setminus\hat{F}_1}(\mathcal{F}_{F\setminus\hat{F}_2}(\mathcal{M}^D))$ and $\mathcal{F}_{F\setminus\hat{F}_2}(\mathcal{F}_{F\setminus\hat{F}_1}(\mathcal{M}^D))$, i.e. the state abstraction transformation is commutative*
2. *$\mathcal{F}_{F\setminus\hat{F}_1}(\mathcal{F}_{F\setminus\hat{F}_2}(\mathcal{M}^D)$ and $\mathcal{F}_{F\setminus(\hat{F}_1\cup\hat{F}_2)}(\mathcal{M}^D)$, i.e. the state abstraction transformation is compositional*

Proposition 11 follows directly from the definition and implies that we can apply state abstraction one fluent at a time and can be applied in any order.

Next moving onto the next three transformations, we see another fascinating property. Namely that the order in which the determinization ($\mathcal{F}_{\Delta}$) and subgoal decomposition ($\mathcal{F}_{\mathbb{D}}$) is applied doesn't matter. Similarly, the applying determinization and local approximation via reachability analyses ($\mathcal{F}^+_{\mathcal{L}}$) in any order also results in the same model. However, applying local approximation after subgoal decomposition is always guaranteed to result in removal of more elements, more formally,

Proposition 12. *For any model* $\mathcal{M}^D$

1. $\mathcal{F}_\Delta(\mathcal{F}_\mathbb{D}(\mathcal{M}^D)) = \mathcal{F}_\mathbb{D}(\mathcal{F}_\Delta(\mathcal{M}^D))$ *and* $\mathcal{F}_\Delta(\mathcal{F}^+_\mathcal{L}(\mathcal{M}^D)) = \mathcal{F}^+_\mathcal{L}(\mathcal{F}_\Delta(\mathcal{M}^D))$
2. *If* $M' = \mathcal{F}^+_\mathcal{L}(\mathcal{F}_\mathbb{D}(\mathcal{M}^D))$, $M'' = \mathcal{F}_\mathbb{D}(\mathcal{F}^+_\mathcal{L}(\mathcal{M}^D))$, *then we can guarantee that* $F' \subseteq F''$ *and* $A' \subseteq A''$.

Proof. The first result follows from the facts that (a) determinization will not change the goal description and subgoal decomposition only changes the goal description and thus are independent of each other and (b) the reachability is already calculated on an all outcome determinization of the model. For the second result, remember that the relaxed planning graph is always only built until the goal is achieved, as such using a subgoal that is easier to reach could let us prune out more actions and fluents. □

Algorithm 2 presents the pseudo-code for the stratified search introduced in the main paper. The algorithm starts with the most abstract model (which only includes the goal fluents) and then searches for the minimal number of fluents to be introduced into the model so that the query criterion κ is satisfied. Let $Comp_{sys}$ be the computational proxy we are using. The successor procedure will

Algorithm 2 Stratified Search.

```
procedure SEARCH
    Input:  M^D, κ, 𝔽, C
    Output: Updated model description M̂^D
    Procedure:
    λ = F^D \ G^D
    curr_model = F_{F\λ}(M^D)
    Fringe.push(curr_model) (Where Fringe is a Queue)
    C_min = ∞
    Abs_min = |F^D|
    while Fringe is not empty do
        curr_model = Fringe.pop()
        Criterion_satisfied = False
        if test_criterion(curr_model,κ) then
            Criterion_satisfied = True
            if Abs_min is less than number of fluents in curr_model then
                set Abs_min to number of fluents in curr_model
            C_new = Comp_H(curr_model, κ)
            if C_min > C_new then
                C_min = C_new
                best_model = curr_model
        Fringe.push(successor(curr_model, 𝔽, Abs_min, Criterion_satisfied))
    if test_criterion(F_Δ(best_model)), κ) then
        best_model ← F_Δ(best_model)
    if test_criterion(F_𝔻(best_model)) then
        best_model ← F_𝔻(best_model)
    if test_criterion(F^+_ℒ(best_model)), κ) then
        best_model ← F^+_ℒ(best_model)
    return best_model
```

only consider generating a model generated through non-concretization transformation (i.e. transformation that adds new fluents in the model) if the number of fluents in the model is equal to Abs_min and the criterion is satisfied by the model. Abs_min starts initialized to the total number of fluents in the original model, but as soon as an abstract model is found then Abs_min gets set to the number of fluents in the model. After that, the search will keep concretizing the models to the same level of abstraction (in terms of the number of fluents) and then try to see if further transformations can be applied while maintaining the criterion being explained.

7. Evaluation

7.1. User study

We performed an ablation study to establish the effectiveness of individual transformations to help users understand the explanation.

Study Objective: Our primary objective with the study is twofold: first, to establish the fact that using model simplification via the framework we introduce is a way to generate explanations that are easier to understand. Secondly, we want to help identify the utility provided by each individual of the four transformation classes.

For the primary objective, we will compare how the users deal with an explanation where simplifying model transformations are applied as opposed to one where the entire model is given to the user. We will compare the effectiveness of an explanation defined over a model containing all four transformations against one where one of the transformation operations is missing. For the secondary objective, we will test how the removal of any single transformation affects the overall effectiveness of the explanations. We recruited

Table 1
Summary of user study results. Average time reported is with 95% confidence interval. Last row reports the p-value calculated from a two tailed homoscedastic T-test.

	all	-determin	-locl-apprx	-decomp	-abs	None
# Filtered	27	27	27	24	24	21
% satisfaction	85.18	81.48	81.48	66.66	66.66	90.47
Participants with correct answers	22	23	24	17	20	18
Avg Time Taken	237.59 ∓26	281.45∓48	369.65∓66	249.06 ∓40	299.6∓46	393.78 ∓71
T-test against **all**	-	0.096	0.0022	0.343	0.0288	0.00042

a total of 180 participants through Prolific [38]. The participants were paid $3.30 for 15 minutes. The study took the form of a timed quiz (the quiz is automatically submitted at 15 minutes) where they read an explanatory dialogue and were asked to answer questions related to the explanation. There was also a bonus of $5 offered to the top two fastest participants from a group who get all the answers right (thus ensuring that people optimize for both completion time and correctness). We required that the participants be fluent in English and it was their first language. For the maximum education degree completed: 30% of all the participants who attempted the test (including those who dropped out in the middle) reported having a Bachelor's degree, 18% a high school degree, 17% having some college credits, and 15% having a master's degree. In the group corresponding to **all-but-determinization**, 88% of participants reported that they understood probabilities. The study was performed on a simple travel planning domain, where the goal is to let the person board a flight. The domain consists of 11 actions, four of which have stochastic effects. We start with a model that contains all the transformations (identified by running an exhaustive search over the transformation space while using the same solvers as in Section 7.2 as proxies for $Comp_H(,)$). It projects out all but six propositional fluents, has a subgoal of getting to the terminal, is determinized, and uses a regression-based local-approximation method to prune out actions that can't possibly contribute to the goal of getting to the terminal. The actual explanatory dialogue consists of a system stating the estimated time taken to reach the final destination (time being a stand-in for cost) and the user in the dialogue asking why it can't be done in an hour. The explanation consists of a description of the corresponding simplified model (generated by filling templates with descriptions of the propositions) and a single execution trace as the explanatory witness. The user is asked to read this dialogue and asked to answer a series of questions. Two questions of particular interest are a filter question that just checks whether the participant read the instructions and then a question that tests whether the user understood the explanation, by asking how they could speed up the travel plan. For the second question, the participant was given five options, only two of which are correct answers. One of which requires the participant to understand the current plan and the second requires them to reason about an alternate initial state. We ignored any participant who got the filter question wrong or whose quizzes timed out. The description of all the models can be found in Appendix A.2.

The six conditions we considered were

1. **all** - A condition consisting of a model that includes all four transformations, here the main explanatory text (model description plus explanatory witness) included 1493 characters.
2. **all-but-determinization** - as the name suggests, all transformations except determinization are applied; the explanatory text included 1805 characters.
3. **all-but-local-approximation** - the explanatory text includes 2274 characters.
4. **all-but-decomposition** - the explanatory text includes 2429 characters. This group has the same model description as **all-but-local-approximation**, but the explanatory witness is longer (as opposed to just talking about reaching terminal A, the trace takes you all the way to boarding the flight).
5. **all-but-abstraction** - the explanatory text includes 3211 characters.
6. **None** - this was the baseline group where the user was exposed to the original model and the explanatory text includes 4596 characters.

We considered 30 participants per condition.

We measure the effectiveness of the explanation both on a subjective level (do people find the explanation satisfying?) and on an objective one (do people find the explanations helpful?). We measure the former by directly asking the participants if they found the answer to the question (i.e., the explanation) satisfying and the latter by checking whether they found correct answers and how long they took to find those answers. Table 1, presents the results of the user study. The table lists - the number of participants who submitted the answers in time and provided the correct answer to the filter question, the percentage of participants who said they were satisfied with the answer, the number of participants who selected at least one correct answer, the average time taken by participants who answered at least one correct answer (reported along with 95% confidence interval) and the p-value obtained from a T-test comparing the time taken by the group that considered all the transformations against each of the other groups.

We ran a one-way ANOVA to compare the time taken by the six groups. Here, the null hypothesis was whether there was a difference between the groups. We found the p-value to be 0.00268959, which allows us to reject the null hypothesis since it is much

lower than the significance level we considered ($\alpha = 0.05$). This establishes that the different model transformation does, at the very least, change the time taken.

Our first objective was to determine whether the transformation does provide an advantage over providing the original model information. To test this, we compare the **None** group against the others, particularly, with **all**. To start with, we see that **None** took the most time per the average. This supports our hypothesis about model simplification providing an advantage. The p-value calculated from the t-test (which can be roughly interpreted as the probability that the samples come from the same population) is 0.00042, which is lower than the standard significance levels ($\alpha = 0.05$) used to establish statistical significance. This means we can be confident that the data for the **None** group is different from those drawn for **all** group. We also see a pretty drastic reduction in the number of participants who got a correct answer. All these evidences point to the fact that, from an objective point of view, the model simplification transformation does provide an advantage.

On the other hand, we also see the highest level of satisfaction for this group. One potential explanation for this result could be the fact that the participants were impressed by the verbosity of the explanation, even though it may have significantly slowed down and even mislead the participant. This results further reinforces arguments made by various recent works (cf. [39]) that subjective satisfaction may be a poor indicator of explanation effectiveness.

Now in regards to the second objective, we see how the removal of each of the individual transformations has some impact on the time taken. Both the groups where abstraction and local approximation were removed reported statistically significant results in regard to the completion time. In terms of the number of participants who correctly provided an answer, we saw the most drastic drop in the group where problem decomposition was removed. Also, it's worth noting that the time taken by the participants isn't simply a function of the verbosity of the explanation. For example, while **all-but-abstraction** is much more verbose than **all-but-local-approximation**, participants in the former group took considerably less time than the latter.

Another interesting point is the difference between **all-but-local-approximation** and **all-but-decomposition**. They are nearly identical in the size of the explanation, but in one, you are asked to reason about a longer horizon (i.e., **all-but-decomposition**), and in the other, you have unnecessary actions that don't contribute to the goal being explained against (i.e., reaching the terminal). We see that in **all-but-local-approximation**, a lot of people can solve the problem but at the cost of much higher time (the p-value returned by the statistical test between the time taken under condition **all** and **all-but-local-approximation** is again under the significance level of 0.05). We believe the drop in the number of participants with correct answers could be explained by the increased cognitive demand imposed by the need to reason over a larger horizon. As such, the need to reason over a longer horizon must have caused the participants to draw the wrong conclusions. This could again point to the deficiency of verbosity as being the sole metric determining the effectiveness of an explanation.

Overall Takeaways. The results show the utility of model simplifying transformations. The transformations that made the biggest impact on the overall time where **all-but-abstraction** and **all-but-local-approximation**. However, **all-but-local-decomposition** resulted in a large drop in the number of users who could find the correct answers. Also, the effectiveness of explanations isn't just reliant on model-description sizes or verbosity.

7.2. Synthetic experiments

Here we will look at the ability of Stratified search to generate simplified models. Here the performance of the simplified model is measured by the time taken to solve the problem using standard solvers and the size of the description. We considered three different solvers, a MAXPROB solver [29], an implementation of LAO* solver [40] for SSP problems (for the cost queries) and the fast-downward planner (ran with A* search and LM-cut heuristic) for deterministic problems [41]. The problems were tested on problems from IPPC problems from 2006 and 2008 [42] and some additional problems for unsolvability. We tested problems corresponding to all three criteria classes. For criterion corresponding to cost (i.e., κ_{ϵ_1}), we only consider domains which is guaranteed to have proper policies (i.e., goal achievement probability is 1). For each domain considered, we selected only problems that could be solved by the solvers within 30 minutes and was appropriate for the specific criteria (i.e. was unsolvable for κ_{ϵ_2} and had the max probability of less than 1 for κ_{ϵ_3}). Since we are unaware of any unsolvability benchmarks (for criterion κ_{ϵ_2}) for probabilistic planning, we took a deterministic domain (from [43]) and turned them into probabilistic domains by randomly changing some of the add effects to stochastic effect with probability 0.5. For κ_{ϵ_1}) we created the query by considering the cost threshold to be 5 (note the SSP solver ignores action cost) and for κ_{ϵ_3}) we used a probability threshold of 1. We also used a constrained version of Blocksworld domain for κ_{ϵ_3} that was introduced by [29].

As clear from Table 2 shows, in every domain, the transformation results in a smaller domain, and in all but the driver domain, it results in a shorter solution time. Table 3 presents the average time taken by different types of transformation across the various unique domains. One thing to note is that the transformation that takes the most time is the problem decomposition. This is because it involves finding potential landmarks and finding the one that meets the required property from the partially ordered landmark. Table 2 also presents the average time taken by the stratified search on each domain.

All experiments were run on an Ubuntu 14.04 machine with 12 cores and 64 GB RAM.

8. Related works

While XAI as a field has been getting a lot of attention [44], explaining sequential decisions is relatively under explored. Though there is a growing recognition that explaining sequential decisions presents unique challenges. Particularly there has been a number of

Table 2
The results of the simulated experiments.

Criterion	Domain	Problem Count	Original Prob Size	Average Simplified Prob Size	Average Solver Time of Original Model	Average Solver Time of Simplified Model	Average Search Time
κ_{e_1}	Blocksworld	5	67.4	25.4	1.647	1.22	96.81
	Elevators	12	75.36	7.86	58.01	1.09	9.69
	Zenotravel	5	81	26.2	534.06	1.26	11357.21
κ_{e_2}	Bottleneck	12	145	116.25	60.38	1.62	56.09
κ_{e_3}	Horizon Constrained Blocksworld	5	115	63.2	1.03	0.88	6476.61
	Drive	15	421.53	51	0.06	0.13	1.28
	Exploding Blocksworld	9	77.44	19.11	53.59	0.17	11.64

Table 3
Average time taken for the individual transformations across unique domains.

Domain	Transformation	Average Time Taken (Secs)
Blocksworld	$\mathcal{T}_{F\setminus\Lambda}$	0.00107
	$\mathcal{T}_{\Delta}$	0.002305
	$\mathcal{T}_{\mathcal{L}}$	0.000331
	$\mathcal{T}_{\mathbb{D}}$	9.518883
Elevator	$\mathcal{T}_{F\setminus\Lambda}$	0.000712
	$\mathcal{T}_{\Delta}$	0.000758
	$\mathcal{T}_{\mathcal{L}}$	0.000282
	$\mathcal{T}_{\mathbb{D}}$	5.780696
Zenotravel	$\mathcal{T}_{F\setminus\Lambda}$	0.001473
	$\mathcal{T}_{\Delta}$	0.001899
	$\mathcal{T}_{\mathcal{L}}$	0.000358
	$\mathcal{T}_{\mathbb{D}}$	3.277179
Bottleneck	$\mathcal{T}_{F\setminus\Lambda}$	0.000781
	$\mathcal{T}_{\Delta}$	0.000637
	$\mathcal{T}_{\mathcal{L}}$	0.000412
	$\mathcal{T}_{\mathbb{D}}$	21.500897
Drive	$\mathcal{T}_{F\setminus\Lambda}$	0.002310
	$\mathcal{T}_{\Delta}$	0.004960
	$\mathcal{T}_{\mathcal{L}}$	0.000334
	$\mathcal{T}_{\mathbb{D}}$	1.124373

recent works that have looked at explaining visual RL agents (for a recent survey for RL explanations in general please refer to [45]). A lot of explanation works from the RL space seem to focus either on feature attribution explanation (cf. [46,47]) or generating policy summaries (cf. [27,48–51]). Most of the works that try to answer 'why' a specific decision was made, focus on the choice of a specific action at a state (while keeping the rest of the policy the same) (cf. [52,22,21]) rather than contrasting the current policy/behavior over whole behaviors/alternate policies. In regards to explanations for model-based sequential decisions, [53] presents a rather comprehensive survey.

This directly connects to earlier works in model reconciliation explanation [54,55], where the goal of explanation is to provide humans with enough model information that they can correctly evaluate the plan in question. In fact, the setting studied in this paper corresponds to a model-reconciliation explanation scenario where we assume the human has no previous information about the system model. Model-reconciliation explanations focus on optimizing for additional criteria like minimizing the amount of information to be provided as part of the explanation. While still relevant in our case, we will leave operationalization of such criteria as future work and focus on the basic tenet that a person would find their explanatory queries resolved if their updated model can support a justification for the query (in our case this correspond to whether the updated model support an explanatory criterion κ) However, the original model-reconciliation explanation method assumes that humans have the adequate inferential capability to correctly use the given model to make perform the necessary verification. This is not an assumption that is necessarily met in practice as the model could be quite complex.

As mentioned most work in model simplification for explanations comes from deterministic planning community. With state abstraction being a popular method investigated by [26] and [24] (many placed in the framework of model reconciliation [54]). [23] also tried to use the ideas from these papers in the context of FOND planning [56]. [24] also talks about finding the first unsolvable

Table 4
Some examples of Explanatory Witness used in the literature.

Paper	Type of Witness	Actual Information
[64,65]	Existential Information	Causal chains
[66]	Proof	Discusses providing plan trace and failure points of the foils
[24]	Abstract Proof	Unreachable subgoal
[67]	Counterfactual Information	A solvable planning problem.
[22]	Abstract Proof	Reachability information under current policy
[68]	Existential Information	Provides action factored differential values - Describes how much better the actions are in the next state given the current optimal action in comparison to the other actions.
[69]	Existential Information	They contrast the outcomes of two actions over an action influence diagram, which is a modified form of structural causal model
[70,71]	Existential Information	Produces a plan that satisfy the user specified alternative
[52]	Abstract Proof	The preference of one action over another is represented in terms of some accumulation of the policy execution after the current action (as opposed to the foil case)
[21]	Abstract Proof	The model consists of an interpretable value function, where the value is decomposed into multiple human interpretable components. Thus the auxiliary information consists of comparing the values across these components.
[72]	Existential Information	Here explanation includes opportunity chains, i.e., information derived from a decision-tree based representation of policies and also the information part of [69]
[57]	Existential Information	Describes the negative outcomes that occur as part of the foil
[73]	Abstract Proof	Provides the closest plan that may be feasible, conflict set and most likely set of actions that can be satisfied
[74]	Existential Information	If the queried formula (ϕ) could have been achieved, it generates a trajectory that satisfied ϕ and presents the outcome of following that trajectory
[75]	Existential Information	Counterfactual states

subgoal. There are also works that try to map inscrutable black box models into models expressed in terms of interpretable features or force RL algorithms to use interpretable features. Representative works in this direction include [52,11,57]. Though unless these new features are hand-selected they don't necessarily have to lead to simpler models. In addition to XAI works, the transformation methods used here also have roots in generating model approximations to speed up planning and generating heuristics. Some relevant works include [58,28,29,59] for abstraction, [60] for landmarks, [31] for determinization.

One of the points just referred to, but not expanded upon was the use of model class simplification transformation. One example is [61], where the generate contrastive explanation generation for the oversubscription planning problem. Here the explanation provides the various constraints between the different possible objectives. We could see these works as performing model class simplification, where they effectively convert a multi-objective planning problem into a constraint satisfaction problem where there are specified constraints among various objectives. A similar approach was also followed by [62,63].

Table 4, presents some example works from the explanation for sequential-decision-making literature that uses explanatory witnesses and the type of information provided by each. Note that most works that use some form of proof-based explanatory witness do so by skipping some information from the proof. For example, to establish the choice of one action over another they may report the Q values of the other actions without establishing why the Q values of the other actions have those specific values. As such we have referred to the information generated from these works as Abstract proof.

The model transformation discussed in this paper is connected to the general notion of abstraction, which has a long history within computer science and has been widely recognized as a fundamental process within the field [76]. Each of our transformations is effectively mapping one transition system to another. Such problems have been studied in the context of multiple fields including formal methods [77]. In many of these cases, the mapping may be selected so as to enforce certain properties by ensuring the mapping is a bisimulation [78], homomorphism [79], etc. Control-theoretic literature also has studied state aggregation as investigated within this paper [80]. Per [76], one of the earliest planning systems to use abstractions was the ABSTRIPS [81] system which again connects to the state abstraction presented here. Since then planning methods have considered multiple state-based [82] and temporal abstractions [83][84].

9. Conclusion and discussion

The paper presents a framework for generating a simplified model representation for the purposes of explanation. In particular, we look at transformations over model descriptions that preserve explanatory criteria. We focus on explaining policies generated using (a) factored MDP models that satisfy P assumption and (b) with respect to some contrastive query. As part of defining this framework, we also establish the space of possible explanatory criteria that can be queried by the user in this setting, perform analysis over some general class of transformations, and formalize the idea of explanatory witness. We perform user studies to validate the specific transformations studied in the paper. Our user study results show that the transformations do help improve the comprehensibility of the explanations. However, we can't just rely on computational intuitions to decide the most useful transformations. Thus, more work needs to be done to identify these transformations' strengths and develop novel transformations better suited for explanations.

CRediT authorship contribution statement

Sarath Sreedharan: Writing – review & editing, Writing – original draft, Software, Conceptualization. **Siddharth Srivastava:** Writing – review & editing, Writing – original draft, Supervision, Conceptualization. **Subbarao Kambhampati:** Writing – review & editing, Writing – original draft, Supervision, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research is supported in part by ONR grants N00014-16-1-2892, N00014-18-1-2442, N00014-18-1-2840, N00014-9-1-2119, NSF grants 1909370, 2303019, AFOSR grant FA9550-18-1-0067, DARPA SAIL-ON grant W911NF19-2-0006 and a JP Morgan AI Faculty Research grant. The authors would like to thank Dr. Dimitri Bertsekas for helpful comments on early drafts of the paper.

Appendix A

A.1. Symbol glossary

Table 5
A summary of all the symbols and notations used in the paper.

Symbol	Description
$\mathcal{M}$	MDP model
S	Set of states
A	Set of actions
P	Depending on the context, it is either the transition function or the probability of reaching the goal (which is one of the optimization objectives)
C	Cost of executing an action in a state
I	The initial state, where $I \in S$
G	The set of goal states, $G \subseteq S$
J	Expected total cost; J^* represents the optimal expected total cost, and J^π the value for a policy π
π, Π	A policy and set of policies respectively
t	A state action trace
$\mathcal{M}^D$	A PPDDL description of an MDP model
F^D	A set of propositional state factors
A^D	A set of action descriptions
prec_i	Preconditions of an action a_i
E_i	Effect set for an action, each effect takes the form $\langle \text{add}_i^j, \text{del}_i^j, p_i^j, c_i^j \rangle$, where add_i^j and del_i^j are the add and delete effect corresponding to the j-th effect. p_i^j gives the likelihood of this effect happening and c_i^j gives the cost associated with the effect
$\mathcal{M}^{R^*}$, $\mathcal{M}^{D^*}$	System's underlying model and its corresponding description
$\mathcal{Q}$	The explanatory query
κ	An optimization threshold
$Comp_{sys}(,)$ $Comp_H(,)$	The computational capability of the system and the human respectively
$\mathcal{F}$	Individual transformation function, $\mathcal{F}_{F\setminus\Lambda}$ - projective state abstraction function, $\mathcal{F}_\Delta$ - determinization function; $\mathcal{F}_\mathbb{D}$ - problem decomposition function and $\mathcal{F}_\mathcal{L}$ - local approximation function
Λ	A subset of the state propositions (i.e. $\Lambda \subseteq F^D$)
T	Bellman operator
$\mathbb{F}$	A set of transformation functions.
$\mathcal{T}$	A sequence of transformation functions

A.2. Domain used for user study

```
  (define (domain get_to_sf)
(:requirements :probabilistic-effects :conditional-effects
:negative-preconditions
:rewards :equality :typing)
(:types elevator floor pos coin)
(:predicates (at_home) (has_cash_voucher)
(on_board_express_train) (on_board_regular_train)
(at_downtown_station) (at_terminal_A) (at_city_station) (checked_in)
(luggaged_checked_in) (at_deignated_gate) (board_flight) (gate_confirmed))

(:action walk_to_train_station
  :parameters ()
  :precondition (and (at_home) )
  :effect (and
              (decrease reward 15)
              (at_city_station)
              (not (at_home))
          )
)

(:action take_taxi_to_downtown
  :parameters ()
  :precondition (and (at_home) (has_cash_voucher) )
  :effect (and
              (decrease reward 15)
              (at_downtown_station)
              (not (at_home))
              (not (has_cash_voucher))
          )
)

(:action catch_train_at_city_station
  :parameters ()
  :precondition (and (at_city_station))
  :effect (and (decrease reward 25)
                 (not (at_city_station))
                 (probabilistic 1/2 (and (on_board_express_train) )
                        1/4 (and (on_board_regular_train)))

  )
)

(:action ride_express_train
  :parameters ()
  :precondition (and (on_board_express_train))
  :effect (and
                 (decrease reward 50)
                 (probabilistic 1/2 (and (at_downtown_station)
                 (not (on_board_express_train))))
                 )
)


(:action ride_regular_train
  :parameters ()
  :precondition (and (on_board_regular_train))
  :effect (and
                 (decrease reward 85)
                 (probabilistic 1/4 (and (at_downtown_station)
```

```
                    (not (on_board_regular_train))))
                    )
)


 (:action catch_shuttle_from_downtown_station_to_terminal_a
  :parameters ()
  :precondition (and (at_downtown_station) (has_cash_voucher))
  :effect (and (decrease reward 15)
               (not (has_cash_voucher))
               (not (at_downtown_station))
               (at_terminal_a))

  )


(:action check_in_terminal_A
  :parameters ()
  :precondition (and (at_terminal_A))
  :effect (and (decrease reward 15)
               (not (has_cash_voucher))
               (checked_in))

  )


(:action check_in_luggage
  :parameters ()
  :precondition (and (at_terminal_A) (checked_in))
  :effect (and (decrease reward 15)
               (luggaged_checked_in))

  )


(:action wait_for_Gate_confirmation
  :parameters ()
  :precondition (and (at_terminal_A) (checked_in))
  :effect (and (decrease reward 15)
               (probabilistic 1/2
               (and (gate_confirmed)))

  ))



(:action go_through_security_clearance_to_gate
;; Possible place to add more stochasticity
  :parameters ()
  :precondition (and (at_terminal_A) (luggaged_checked_in)
  (gate_confirmed) (checked_in))
  :effect (and (decrease reward 5)
               (at_deignated_gate))

  )



(:action board_flight
  :parameters ()
  :precondition (and (at_deignated_gate))
```

```
    :effect (and (decrease reward 5)
                 (board_flight))

    )
)
```

A.3. Template used in translating PDDL to natural language

Each proposition in the domain is converted into an English sentence using the following rules
If the proposition starts with 'has', it get converted into

```
"The person has a " + proposition string with 'has' removed and
the underscore is replaced with space
```

else

```
"The person is " + proposition string with underscore is replaced
with space
```

For each action, we create a string of the form

```
Action: <Action name string>
 If the following facts are true:
   <string of preconditions>
Doing this will cause the following facts to be True:
   <string of add effects>
Doing this will cause the following facts to be False:
   <string of delete effects>
Time taken:
   It will take you <cost> mins to do it
```

For probabilistic effects, you add the string "with probability p", where p is replaced with actual probability value.

Readers can view the actual text shown to the study participant in the following GitHub repo: https://github.com/HAPILab/model_simplification_files/tree/main/user_study_pdfs.

Data availability

Data will be made available on request.

References

[1] P. Langley, Varieties of explainable agency, in: XAIP Workshop, XAIP, 2019, pp. 113–117.
[2] D. Gunning, D.W. Aha, Darpa's explainable artificial intelligence (XAI) program, AI Mag. 40 (2019) 44–58.
[3] D.P. Bertsekas, Dynamic Programming and Optimal Control, 3rd edition, Athena Scientific, 2005, https://www.worldcat.org/oclc/314894080.
[4] A. Newell, H.A. Simon, et al., Human Problem Solving, vol. 104, Prentice-Hall, Englewood Cliffs, NJ, 1972.
[5] A. Kolobov, Mausam, D.S. Weld, H. Geffner, Heuristic search for generalized stochastic shortest path mdps, in: Proceedings of the 21st International Conference on Automated Planning and Scheduling, ICAPS 2011, Freiburg, Germany, June 11-16, 2011, AAAI, 2011, pp. 130–137.
[6] M.L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming, John Wiley & Sons, 2014.
[7] A. Kolobov, Mausam, D.S. Weld, A theory of goal-oriented mdps with dead ends, in: Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence, Catalina Island, CA, USA, August 14–18, 2012, AUAI Press, 2012, pp. 438–447, https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1%26smnu=2%26article_id=2305%26proceeding_id=28.
[8] H.L. Younes, M.L. Littman, Ppddl1. 0: the language for the probabilistic part of ipc-4, in: Proc. International Planning Competition, 2004, pp. 70–74.
[9] M. Ghallab, D. Nau, P. Traverso, Automated Planning: Theory and Practice, Elsevier, 2004.
[10] A. Boonzaier, J. McClure, R.M. Sutton, Distinguishing the effects of beliefs and preconditions: the folk psychology of goals and actions, Eur. J. Soc. Psychol. 35 (2005) 725–740.
[11] S. Sreedharan, U. Soni, M. Verma, S. Srivastava, S. Kambhampati, Bridging the gap: providing post-hoc symbolic explanations for sequential decision-making problems with black box simulators, CoRR, arXiv:2002.01080, 2020.
[12] G. Konidaris, L.P. Kaelbling, T. Lozano-Perez, From skills to symbols: learning symbolic representations for abstract high-level planning, J. Artif. Intell. Res. 61 (2018) 215–289.
[13] T. Miller, Explanation in artificial intelligence: insights from the social sciences, Artif. Intell. 267 (2019) 1–38.
[14] J. Hoffmann, D. Magazzeni, Explainable ai planning (xaip): overview and the case of contrastive explanation, in: Reasoning Web. Explainable Artificial Intelligence, 2019, pp. 277–282.
[15] D.S. Weld, G. Bansal, The challenge of crafting intelligible intelligence, Commun. ACM 62 (2019) 70–79.
[16] T. Chakraborti, S. Kambhampati, (when) can ai bots lie?, in: Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society, 2019, pp. 53–59.
[17] D. Kahneman, A. Tversky, Prospect theory: an analysis of decision under risk, in: Handbook of the Fundamentals of Financial Decision Making: Part I, World Scientific, 2013, pp. 99–127.

[18] S. Nikolaidis, A. Kuznetsov, D. Hsu, S.S. Srinivasa, Formalizing human-robot mutual adaptation: a bounded memory model, in: C. Bartneck, Y. Nagai, A. Paiva, S. Sabanovic (Eds.), The Eleventh ACM/IEEE International Conference on Human Robot Interaction, HRI 2016, Christchurch, New Zealand, March 7-10, 2016, IEEE/ACM, 2016, pp. 75–82.
[19] J.R. Wright, K. Leyton-Brown, Predicting human behavior in unrepeated, simultaneous-move games, Games Econ. Behav. 106 (2017) 16–37.
[20] T. Zhi-Xuan, J. Mann, T. Silver, J. Tenenbaum, V. Mansinghka, Online bayesian goal inference for boundedly rational planning agents, in: Advances in Neural Information Processing Systems, vol. 33, Curran Associates, Inc., 2020, pp. 19238–19250.
[21] Z. Juozapaitis, A. Koul, A. Fern, M. Erwig, F. Doshi-Velez, Explainable reinforcement learning via reward decomposition, in: IJCAI XAI Workshop, 2019, pp. 47–53.
[22] O.Z. Khan, P. Poupart, J.P. Black, Minimal sufficient explanations for factored Markov decision processes, in: Proceedings of the 19th International Conference on Automated Planning and Scheduling, ICAPS 2009, Thessaloniki, Greece, September 19–23, 2009, AAAI, 2009, pp. 194–200.
[23] S. Sreedharan, T. Chakraborti, C. Muise, Y. Khazaeni, S. Kambhampati, D3WA+ - a case study of XAIP in a model acquisition task for dialogue planning, in: Proceedings of the Thirtieth International Conference on Automated Planning and Scheduling, ICAPS 2020, Nancy, France, October 26-30, 2020, AAAI Press, 2020, pp. 488–498.
[24] S. Sreedharan, S. Srivastava, D.E. Smith, S. Kambhampati, Why can't you do that hal? Explaining unsolvability of planning tasks, in: IJCAI 2019, ijcai.org, 2019, pp. 1422–1430.
[25] L. Li, T.J. Walsh, M.L. Littman, Towards a unified theory of state abstraction for mdps, ISAIM 4 (2006) 5.
[26] S. Sreedharan, S. Srivastava, S. Kambhampati, Using state abstractions to compute personalized contrastive explanations for ai agent behavior, Artif. Intell. 301 (2021) 103570.
[27] N. Topin, M. Veloso, Generation of policy-level explanations for reinforcement learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 2514–2521.
[28] B. Van Roy, Performance loss bounds for approximate value iteration with state aggregation, Math. Oper. Res. 31 (2006) 234–244.
[29] T. Klößner, J. Hoffmann, M. Steinmetz, A. Torralba, Pattern databases for goal-probability maximization in probabilistic planning, in: Proceedings of the International Conference on Automated Planning and Scheduling, ICAPS 2021, Guangzhou, China, August 2-13, 2021, Volume 31, 2021, pp. 201–209.
[30] R. Eifler, M. Steinmetz, A. Torralba, J. Hoffmann, Plan-space explanation via plan-property dependencies: faster algorithms & more powerful properties, in: IJCAI, ijcai.org, 2020, pp. 4091–4097.
[31] S.W. Yoon, A. Fern, R. Givan, Ff-replan: a baseline for probabilistic planning, in: Proceedings of the Seventeenth International Conference on Automated Planning and Scheduling, ICAPS 2007, Providence, Rhode Island, USA, September 22-26, 2007, AAAI, 2007, p. 352.
[32] S.W. Yoon, A. Fern, R. Givan, S. Kambhampati, Probabilistic planning via determinization in hindsight, in: AAAI, 2008, pp. 1010–1016.
[33] S. Sreedharan, S. Srivastava, S. Kambhampati, Tldr: policy summarization for factored SSP problems using temporal abstractions, in: Proceedings of the Thirtieth International Conference on Automated Planning and Scheduling, Nancy, France, October 26-30, 2020, AAAI Press, 2020, pp. 272–280.
[34] J. Hoffmann, J. Porteous, L. Sebastia, Ordered landmarks in planning, J. Artif. Intell. Res. 22 (2004) 215–278.
[35] L. Zhu, R. Givan, Landmark extraction via planning graph propagation, in: ICAPS Doctoral Consortium, 2003, pp. 156–160.
[36] M.T. Ribeiro, S. Singh, C. Guestrin, "why should I trust you?": explaining the predictions of any classifier, in: B. Krishnapuram, M. Shah, A.J. Smola, C.C. Aggarwal, D. Shen, R. Rastogi (Eds.), Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016, ACM, 2016, pp. 1135–1144.
[37] J. Hoffmann, B. Nebel, The ff planning system: fast plan generation through heuristic search, J. Artif. Intell. Res. 14 (2001) 253–302.
[38] Prolific, UK, Prolific, 2021.
[39] R.R. Hoffman, S.T. Mueller, G. Klein, J. Litman, Metrics for explainable ai: challenges and prospects, preprint, arXiv:1812.04608, 2018.
[40] E.A. Hansen, S. Zilberstein, Lao*: a heuristic search algorithm that finds solutions with loops, Artif. Intell. 129 (2001) 35–62.
[41] M. Helmert, The fast downward planning system, J. Artif. Intell. Res. 26 (2006) 191–246.
[42] D. Bryce, O. Buffet, 6th international planning competition: uncertainty part, in: Proceedings of the 6th International Planning Competition (IPC'08), 2008.
[43] J. Hoffmann, P. Kissmann, Á. Torralba, "distance"? Who cares? Tailoring merge-and-shrink heuristics to detect unsolvability, in: ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014), in: Frontiers in Artificial Intelligence and Applications, vol. 263, IOS Press, 2014, pp. 441–446.
[44] H. Lakkaraju, J. Adebayo, S. Singh, Explaining machine learning predictions: state-of-the-art, challenges, and opportunities, https://explainml-tutorial.github.io/, 2020, NeurIPS Tutorial.
[45] A. Alharin, T.-N. Doan, M. Sartipi, Reinforcement learning interpretation methods: a survey, IEEE Access 8 (2020) 171058–171077.
[46] S. Greydanus, A. Koul, J. Dodge, A. Fern, Visualizing and understanding atari agents, in: Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018, in: Proceedings of Machine Learning Research, vol. 80, PMLR, 2018, pp. 1787–1796.
[47] T. Huber, E. André, Introducing selective layer-wise relevance propagation to dueling deep Q-learning, in: XAI Workshop, 2019.
[48] B. Hayes, J.A. Shah, Improving robot controller transparency through autonomous policy explanation, in: 2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI), IEEE, 2017, pp. 303–312.
[49] D. Amir, O. Amir, Highlights: summarizing agent behavior to people, in: Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, 2018, pp. 1168–1176.
[50] I. Lage, D. Lifschitz, F. Doshi-Velez, O. Amir, Exploring computational user models for agent policy summarization, in: IJCAI, 2019.
[51] A. Koul, A. Fern, S. Greydanus, Learning finite state representations of recurrent policy networks, in: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019, OpenReview.net, 2019.
[52] Z. Lin, K. Lam, A. Fern, Contrastive explanations for reinforcement learning via embedded self predictions, in: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021, OpenReview.net, 2021.
[53] T. Chakraborti, S. Sreedharan, S. Kambhampati, The emerging landscape of explainable ai planning and decision making, in: IJCAI, 2020.
[54] T. Chakraborti, S. Sreedharan, Y. Zhang, S. Kambhampati, Plan explanations as model reconciliation: moving beyond explanation as soliloquy, in: IJCAI 2017, ijcai.org, 2017, pp. 156–163.
[55] S. Sreedharan, T. Chakraborti, S. Kambhampati, Foundations of explanations as model reconciliation, Artif. Intell. 301 (2021) 103558.
[56] A. Cimatti, M. Pistore, M. Roveri, P. Traverso, Weak, strong, and strong cyclic planning via symbolic model checking, Artif. Intell. 147 (2003) 35–84.
[57] J. Waa, J.v. Diggelen, K. Bosch, M. Neerincx, Contrastive explanations for reinforcement learning in terms of expected consequences, in: IJCAI Workshop on Explainable AI (XAI), 2018.
[58] S. Srivastava, S.J. Russell, A. Pinto, Metaphysics of planning domain descriptions, in: D. Schuurmans, M.P. Wellman (Eds.), Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA, AAAI Press, 2016, pp. 1074–1080.
[59] R. Givan, S. Leach, T. Dean, Bounded-parameter Markov decision processes, Artif. Intell. 122 (2000) 71–109.
[60] S. Richter, M. Westphal, The lama planner: guiding cost-based anytime planning with landmarks, J. Artif. Intell. Res. 39 (2010) 127–177.
[61] R. Eifler, M. Cashmore, J. Hoffmann, D. Magazzeni, M. Steinmetz, A new approach to plan-space explanation: analyzing plan-property dependencies in oversubscription planning, in: AAAI 2020, AAAI Press, 2020, pp. 9818–9826.
[62] R. Sukkerd, R. Simmons, D. Garlan, Toward explainable multi-objective probabilistic planning, in: ICSE Workshop on Software Engineering for Smart Cyber-Physical Systems (SEsCPS), 2018.
[63] R. Sukkerd, R. Simmons, D. Garlan, Tradeoff-focused contrastive explanation for MDP planning, arXiv:2004.12960, 2020.

[64] B. Seegebarth, F. Müller, B. Schattenberg, S. Biundo, Making hybrid plans more clear to human users - a formal approach for generating sound explanations, in: Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling, ICAPS 2012, Atibaia, São Paulo, Brazil, June 25-19, 2012, AAAI, 2012.
[65] P. Bercher, S. Biundo, T. Geier, T. Hoernle, F. Nothdurft, F. Richter, B. Schattenberg, Plan, repair, execute, explain - how planning helps to assemble your home theater, in: Proceedings of the Twenty-Fourth International Conference on Automated Planning and Scheduling, ICAPS 2014, Portsmouth, New Hampshire, USA, June 21-26, 2014, AAAI, 2014.
[66] S. Sreedharan, S. Srivastava, S. Kambhampati, Hierarchical expertise level modeling for user specific contrastive explanations, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI, 2018, July 13-19, 2018, Stockholm, Sweden, ijcai.org, 2018, pp. 4829–4836.
[67] M. Göbelbecker, T. Keller, P. Eyerich, M. Brenner, B. Nebel, Coming up with good excuses: what to do when no plan can be found, in: Proceedings of the International Conference on Automated Planning and Scheduling, 2010, pp. 81–88.
[68] T. Dodson, N. Mattei, J.T. Guerin, J. Goldsmith, An English-Language Argumentation Interface for Explanation Generation with Markov Decision Processes in the Domain of Academic Advising, TiiS, 2013.
[69] P. Madumal, T. Miller, L. Sonenberg, F. Vetere, Explainable reinforcement learning through a causal lens, in: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, the Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, the Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, AAAI Press, 2020, pp. 2493–2500.
[70] B. Krarup, S. Krivic, D. Magazzeni, D. Long, M. Cashmore, D.E. Smith, Contrastive explanations of plans through model restrictions, J. Artif. Intell. Res. 72 (2021) 533–612.
[71] B. Krarup, M. Cashmore, D. Magazzeni, T. Miller, Model-based contrastive explanations for explainable planning, in: XAIP Workshop, 2019.
[72] P. Madumal, T. Miller, L. Sonenberg, F. Vetere, Distal explanations for explainable reinforcement learning agents, preprint, arXiv:2001.10284, 2020.
[73] K. Valmeekam, S. Sreedharan, S. Sengupta, S. Kambhampati, Radar-x: an interactive interface pairing contrastive explanations with revised plan suggestions, preprint, arXiv:2011.09644, 2020.
[74] D. Kasenberg, R. Thielstrom, M. Scheutz, Generating explanations for temporal logic planner decisions, in: Proceedings of the International Conference on Automated Planning and Scheduling, vol. 30, 2020, pp. 449–458.
[75] M.L. Olson, L. Neal, F. Li, W.-K. Wong, Counterfactual states for atari agents via generative deep learning, preprint, arXiv:1909.12969, 2019.
[76] L. Saitta, J.-D. Zucker, Abstraction in Artificial Intelligence and Complex Systems, 2013.
[77] K.N. Oikonomou, Abstractions of random finite-state machines, Form. Methods Syst. Des. 18 (2001) 171–207.
[78] R. Givan, T. Dean, M. Greig, Equivalence notions and model minimization in Markov decision processes, Artif. Intell. 147 (2003) 163–223.
[79] R. Fitch, B. Hengst, D. Šuc, G. Calbert, J. Scholz, Structural abstraction experiments in reinforcement learning, in: Australasian Joint Conference on Artificial Intelligence, Springer, 2005, pp. 164–175.
[80] D.P. Bertsekas, et al., Dynamic Programming and Optimal Control, Volume ii, 3rd edition, Athena Scientific 1, Belmont, MA, 2011.
[81] E.D. Sacerdoti, Planning in a hierarchy of abstraction spaces, Artif. Intell. 5 (1974) 115–135.
[82] C. Backstrom, P. Jonsson, Bridging the gap between refinement and heuristics in abstraction, in: Twenty-Third International Joint Conference on Artificial Intelligence, Citeseer, 2013.
[83] K. Erol, J.A. Hendler, D.S. Nau, Semantics for Hierarchical Task-Network Planning, University of Maryland College Park, 1994.
[84] R.E. Korf, Macro-operators: a weak method for learning, Artif. Intell. 26 (1985) 35–77.