On Generating Personalized Explanations via Knowledge Forgetting

Stylianos Loukas Vasileiou , Ashwin Kumar , William Yeoh
Washington University in St. Louis
{v.stylianos, ashwinkumar, wyeoh}@wustl.edu

Abstract

We investigate the problem of generating explanations within the context of Human-aware AI Planning. Particularly, we focus on an explanatory setting for tasks encoded in a logical formalism, where given an agent model (encoding the task), an explanandum entailed by the agent, and a user vocabulary specifying terms in the task, the goal is to find an explanation that is at an appropriate abstraction level with respect to the user's vocabulary. We propose a logic-based framework for generating explanations via a logic-based method called *knowledge forgetting*, as well as present an algorithmic procedure for computing them.

1 Introduction

Human-aware AI Planning (HAIP) is a paramount area of research concentrating on helping human users interface with AI agents in complex (sequential) decision-making tasks [Kambhampati, 2019]. A typical HAIP scenario involves an agent M_a explaining an explanandum φ that is inexplicable to a human user M_h, where M_a and M_h are the models that encode the agent's and the user's version of a task, respectively. For example, given a planning task Π, M_a and M_h encode the agent's and the user's assumptions of the (action) dynamics of Π, respectively. This paradigmatic approach is referred to as the *Model Reconciliation Problem* (MRP) [Chakraborti *et al.*, 2017], and its predominant goal is to explain to users, in terms of model differences, why the agent's decision (e.g., a proposed plan π for Π) is valid (or optimal). A common thread around MRP approaches is the assumption that the user's model M_h is known to the agent a-priori and it is at the same abstraction level as the agent's model M_a [Chakraborti *et al.*, 2017; Sreedharan *et al.*, 2018; Chakraborti *et al.*, 2020; Vasileiou *et al.*, 2022]. Albeit a reasonable assumption whose role was to help establish the foundations of MRP frameworks, it, arguably, limits their practicality, insofar as M_h may diverge from the actual user's model. Consequently, this could lead to the generation of explanations that are either incoherent to the user's or at a lower (or higher) abstraction level than their expertise (i.e., familiarity with the task) requires.

As the primary motive of explanatory systems is to produce coherent and intelligible explanations for human users, it is necessary to consider the relaxation of the aforementioned assumption. In particular, one possible direction is to assume that the agent does not possess an expllicit user model, but rather a vocabulary that specifies terms the user knows about the given task. Implicit in the vocabulary is the user's expertise level of the given task, which the agent could exploit and use to generate explanations consisting of terms the user is familiar with, and abstract away terms the user does not know. In other words, the agent could generate *personalized explanations*, that is, explanations that are at an appropriate abstraction level with respect to vocabulary.

To this end, we center our attention on (sequential) decision-making tasks that can be encoded in a logical formalism and consider a framework for generating personalized explanations for human users. Particularly, we present a general logic-based framework, where given an agent knowledge base KB_a encoding the task, an explanandum φ entailed by KB_a, and a human user vocabulary $\mathcal{V}_h$ consisting of user specified terms, the goal is to generate an explanation that is at an appropriate abstraction level with respect to $\mathcal{V}_h$. To generate the explanations, we leverage a method called *knowledge forgetting* [Delgrande, 2017] and utilize it to define the notion of *personalized explanations*. We present a simple algorithmic approach for computing personalized explanations based on [Vasileiou *et al.*, 2021], and evaluate its efficacy on a set of SAT-based benchmarks. While the operation of knowledge forgetting has been extensively studied in various settings [Zhang and Zhou, 2009; Lutz and Wolter, 2011; Wang *et al.*, 2014], its applicability in the context of HAIP and explanation gneration has not been explored, to the best of our knowledge.

Knowledge forgetting has an ordering function in the human mind. It can be seen as a process of ommiting information or knowledge from one's memory in such a way that it is no longer present or reproducible. From a cognitive point of view, it is a gradual process in which information that is less used is moved to the "background", from which it eventually dissipates or recovered through remembering to the foreground [Ebbinghaus, 2013]. This basic mechanism helps us to deal with information overload by suppressing irrelevant information as well as increase our attention. Interestingly, this view of knowledge forgetting aligns with a prag-

matic framework in cognitive linguistics called *relevance theory* [Wilson and Sperber, 2002]. Relevance theory suggests that the relevance of an utterance depends on two aspects– maximizing the recipient's cognitive effect and minimizing their cognitive effort. In essence, knowledge forgetting can be seen as an operation for achieving those two aspects, as suppressing irrelevant information from an utterance can minimize effort, and increased attention can maximize its effect. In our context, this can be translated as searching for an explanation that is at an appropriate abstraction level (e.g., by forgetting uncessary and irrelevant information) for the user given their vocabulary.

2 Logical Preliminaries

We assume a propositional language L consisting of a finite set of propositional letters Γ. The simplest formulae in L are *literals*, which are letters or their negations, while more complex formulae can be recursively built up from letters and the classical logical connectives. A *knowledge base* KB is a set of formulae. The set of letters used in the formulae of KB is called the vocabulary of KB, denoted by $\mathcal{V}_{KB}$. An *interpretation* is a function $\mathcal{I} : \Gamma \rightarrow \{\top, \bot\}$, and if there exists an interpretation that satisfies a KB we say that KB is *satisfiable*, otherwise KB is *unsatisfiable*, denoted by $KB \models \bot$. A KB entails a formula φ, denoted by $KB \models \varphi$, if and only if $KB \cup \{\neg\varphi\} \models \bot$. Moreover, unless stated otherwise, in what follows we assume that a KB is satisfiable and expressed in *conjunctive normal form* (CNF), that is, a conjunction of clauses, each of which is a disjunction of literals.

Definition 1 (Explanation)**.** *Given $KB \models \varphi$, $\epsilon \subseteq KB$ is an explanation for φ from KB if $\epsilon \models \varphi$ and $\forall \epsilon' \subset \epsilon$, $\epsilon' \not\models \varphi$.*

Definition 2 (Minimal Unsatisfiable Set (MUS))**.** *Given $KB \models \bot$, a subset $\mathcal{U} \subseteq KB$ is an MUS if $\mathcal{U} \models \bot$ and $\forall \mathcal{U}' \subset \mathcal{U}$, $\mathcal{U}'$ is satisfiable.*

By definition, every unsatisfiable KB contains at least one MUS. MUSes and explanations are related by the following:

Proposition 1. *Given $KB \models \varphi$, $\epsilon \subseteq KB$ is an explanation for φ ($\epsilon \models \varphi$) iff $\epsilon \cup \{\neg\varphi\}$ is an MUS of $KB \cup \{\neg\varphi\}$.*

Additionally, given a knowledge base KB, we will write KB^* with $* \in \{s, h\}$ to denote a set of formulae that will be treated as soft and hard, respectively. Intuitively, the hard formulae are those formulae that will not be removed by the minimization procedure [Marques-Silva, 2012].

The framework presented here is closely tied to the foundations laid in [Vasileiou *et al.*, 2022], namely the logic-based version of the model reconciliation problem (L-MRP).[1] In L-MRP, one is given two knowledge bases KB_a and KB_h of the agent providing an explanation and the human receiving the explanation, respectively, such that $KB_a \models \varphi$ and $KB_h \not\models \varphi$, and the goal is to find an explanation $\epsilon = \langle \epsilon^+, \epsilon^- \rangle$, where $\epsilon^+ \subseteq KB_a$ and $\epsilon^- \subseteq KB_h$, such that $(KB_h \cup \epsilon^+) \setminus \epsilon^- \models \varphi$.

The set of formulae ϵ are referred to as the update of the knowledge base KB_h, where new formulae ϵ^+ from KB_a are added, and erroneous formulae ϵ^- from KB_h are removed to ensure consistency.

It is important to note that L-MRP is based on the following fundamental assumptions: (*i*) the agent's knowledge base KB_a serves the ground truth; and *(ii)* the user's knowledge base KB_h is known to the agent a-priori. In this paper, we will be relaxing the second assumption, where instead of a full-fledged KB_h, we will only require that a user-defined vocabulary $\mathcal{V}_h$ is provided. As this assumption is significantly more reasonable and realistic, we anticipate that our work is a move in the right direction towards practicality.

3 Abstractions via Knowledge Forgetting

As introduced in Section 1, *knowledge forgetting*, henceforth forgetting, is an operation of the human mind that suppresses irrelevant information and improves cognitive capabilities in order to focus on the relevant aspects of a given problem. For instance, when humans are focusing on a specific problem, they tend to "forget" irrelevant aspects, or when trying to find a solution under restricted conditions, they have to intentionally "forget" ways of solving the problem in richer environments. These examples elucidate that intentional forgetting in humans is a necessary complex cognitive process that involves many aspects of knowledge and reasoning. Without delving into the details of the cognitive characterizations of forgetting in humans, in this section we consider a logic-based method for intentional forgetting in AI agents.

Forgetting is taken to be an operation that decreases the language of an agent, insofar as the vocabulary of the agent's language is reduced. Specifically, assume a knowledge base KB over a vocabulary $\mathcal{V}_{KB}$. The operation of forgetting $\lambda \subset \mathcal{V}_{KB}$ from KB is the logical consequences of KB expressible over $\mathcal{V}_{KB} \setminus \lambda$. Forgetting is applied in the contents of an agent's knowledge base and is independent of the underlying formalism.

Delgrande [2017] presents a succinct mechanism for computing forgetting for various logics, however, in this paper we focus on its propositional logic treatment.[2]

Definition 3. *Let KB be a knowledge base and $\lambda \in \mathcal{V}_{KB}$ a letter in its vocabulary. Define $KB_{\downarrow\lambda} = \{\varphi \in KB \mid \lambda \notin \mathcal{V}_\varphi\}$. Similarly, $KB_{\uparrow\lambda} = \{\varphi \in KB \mid \lambda \in \mathcal{V}_\varphi\}$.*

That is, $KB_{\downarrow\lambda}$ is simply those formulae of KB that do not mention λ, and $KB_{\uparrow\lambda}$ is the formulae that mention λ. In the next definition, $Res(KB_\uparrow, \lambda)$ is the set of formulae obtained from $KB_{\uparrow\lambda}$ by carrying out all possible resolutions with respect to letter λ.

Definition 4. *Let KB be a knowledge base and $\lambda \in \mathcal{V}_{KB}$ a letter in its vocabulary. Define $Res(KB_{\uparrow\lambda}, \lambda) = \{\varphi \mid \exists \varphi_1, \varphi_2 \in KB_\uparrow$ s.t. $\lambda \in \varphi_1, \neg\lambda \in \varphi_2$, and $\varphi = (\varphi_1 \setminus \{\lambda\}) \cup (\varphi_2 \setminus \{\neg\lambda\})\}$.*

Now, Definitions 3 and 4 can be combined to compute forgetting, resulting in the following definition:

[1] The MRP problem was originally developed by [Chakraborti *et al.*, 2017] for (classical) planning tasks.

[2] For a thorough analysis of forgetting and its various theoretical properties, we refer the interested reader to the work by [Delgrande, 2017].

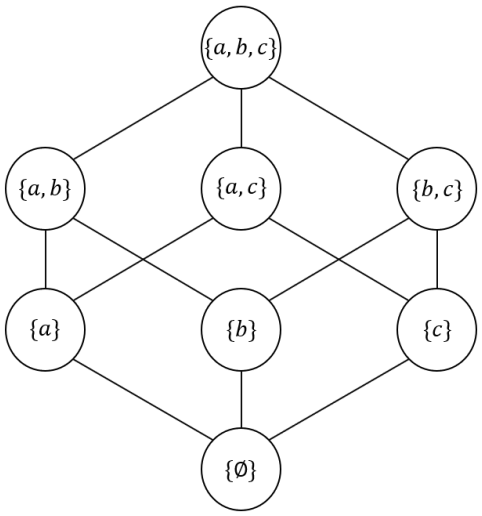

Figure 1: A level-3 abstraction lattice for $KB = \{a, b, \neg a \vee c, \neg b \vee \neg c \vee d\}$. At the root is level-0 of the lattice, i.e., the initial $\mathcal{F}(KB, \{\emptyset\}) = KB$. The child nodes of the root form level-1 of the lattice and represent (from left to right): $\mathcal{F}(KB, \{a\}) = \{b, c, \neg b \vee \neg c \vee d\}$, $\mathcal{F}(KB, \{b\}) = \{a, \neg a \vee c, \neg c \vee d\}$, and $\mathcal{F}(KB, \{c\}) = \{a, b, \neg a \vee \neg b \vee d\}$. Similarly, the subsequent nodes form level-2, and so on.

Definition 5 (Forgetting)**.** *Let KB be a knowledge base and $\lambda \in \mathcal{V}_{KB}$ a letter in its vocabulary. Then, forgetting λ from KB is defined as $\mathcal{F}(KB, \lambda) = KB_{\downarrow\lambda} \cup Res(KB_{\uparrow\lambda}, \lambda)$.*

Definition 5 can be interpreted as follows: Perform all possible resolutions with respect to the letter to be forgotten, and add these resolvents to those formulae in KB that do not mention that letter.[3] While the resulting KB will be weaker than before, one of the key advantages of this mechanism is that the resulting KB entails the same set of formulae that are irrelevant to what was forgotten. More formally,

Corollary 1. *Let KB be a knowledge base and $\mathcal{V}_{KB}$ its vocabulary. If $KB \models \varphi$, then $\forall \lambda \subset \mathcal{V}_{KB} \backslash \mathcal{V}_{\varphi}$, $\mathcal{F}(KB, \lambda) \models \varphi$.*

Example 1. *Let $KB = \{a, b, \neg a \vee c, \neg b \vee \neg c \vee d\}$ and $\mathcal{V}_{KB} = \{a, b, c, d\}$, and $\lambda = \{a\}$. Notice that $KB \models d$. Now, we have $KB_{\downarrow}a = \{b, \neg b \vee \neg c \vee d\}$ and $KB_{\uparrow}a = \{a, \neg a \vee c\}$, and $Res(KB_{\uparrow}a, a) = \{c\}$. Then, $\mathcal{F}(KB, \lambda) = \{b, c, \neg b \vee \neg c \vee d\}$, where $\mathcal{F}(KB, a) \models d$.*

Importantly now, in this paper we want to utilize the forgetting operation as an abstraction method on a knowledge base. In essence, this abstraction method simplifies the formulae of the knowledge base by forgetting a set of letters from its vocabulary. Formally,

Definition 6 (Abstraction)**.** *Let KB be a knowledge base and $\lambda \subset \mathcal{V}_{KB}$ a subset of its vocabulary. Then, $\mathcal{F}(KB, \lambda)$ is an abstraction of KB.*

In what follows, given a knowledge base KB such that $KB \models \varphi$, we assume $\mathcal{V}_{KB} = \mathcal{V}_{KB} \setminus \mathcal{V}_{\varphi}$ such that $\forall \lambda \subset \mathcal{V}_{KB}$, $\mathcal{F}(KB, \lambda) \models \varphi$ (Corollary 1).

[3]Note that computing forgetting for a set of letters can be done iteratively, i.e., $\mathcal{F}(KB, \lambda_1 \cup \lambda_2) = \mathcal{F}(\mathcal{F}(KB, \lambda_1), \lambda_2)$.

What is interesting here is that, through the operation of forgetting, we can define an *abstraction lattice* specifying the abstraction levels that can be achieved on a knowledge base given a set of letters. Figure 1 shows an abstraction lattice based on Example 1. Finally, one important effect of the machinery of forgetting is that we can generate formulae that were not previously contained in KB_a. This gives us the opportunity to explore the construction of personalized explanations, as we will see in the next section.

4 Explanation Generation Framework

Similar to the concept of MRP, our explanation generation setting concerns an agent explaining an explanandum to a human user. In particular, we assume the following:

- An agent knowledge base KB_a encoding a task (e.g., planning) in a logical language. The agent's knowledge base KB_a is logically closed, insofar as the agent is "logically omniscient" about the problem.[4]
- The user provides the following information to the agent: (i) The explanandum φ, where $KB_a \models \varphi$; and (ii) a vocabulary $\mathcal{V}_h \subseteq \mathcal{V}_{KB_a}$.

Thus, given an agent KB_a, an explanandum φ such that $KB_a \models \varphi$, and a user vocabulary $\mathcal{V}_h$, our goal is to *find an explanation from KB_a for φ that is at an appropriate abstraction level for the user.* We will call such explanations *personalized explanations*. But how can we generate them?

The first thing to note is that there may be multiple explanations from KB_a for φ, and as such, we need a way to differentiate between them. This leads us to the definition of the *most-relevant explanation* (with respect to $\mathcal{V}_h$), that is, the smallest explanation containing the most letters from $\mathcal{V}_h$:

Definition 7 (Most-Relevant Explanation)**.** *Let $KB_a \models \varphi$ and $\mathcal{V}_h \subseteq \mathcal{V}_{KB_a}$. Then, $\epsilon \subseteq KB_a$ is a most-relevant explanation iff $\epsilon \models \varphi$ and $\nexists \epsilon' \subseteq KB_a$ s.t. $\epsilon' \models \varphi$ and $\mathcal{R}(\epsilon', \mathcal{V}_h) > \mathcal{R}(\epsilon, \mathcal{V}_h)$, where $\mathcal{R}(\epsilon, \mathcal{V}_h) = \frac{|\mathcal{V}_h \cap \mathcal{V}_\epsilon|}{|\mathcal{V}_\epsilon|} + \frac{1}{|\epsilon|}$ is a relevance measure.*

Example 2. *Let $KB_a = \{a, b, \neg a \vee \neg b \vee c, d, \neg d \vee b\}$ and $\mathcal{V}_h = \{a, d\}$, and let $\epsilon_1 = \{a, b, \neg a \vee \neg b \vee c\}$ and $\epsilon_2 = \{a, d, \neg d \vee b, \neg a \vee \neg b \vee c\}$ be two explanations for c from KB_a. Then, ϵ_2 is the most-relevant explanation as $\mathcal{R}(\epsilon_2, \mathcal{V}_h) = \frac{11}{12} > \mathcal{R}(\epsilon_1, \mathcal{V}_h) = \frac{5}{6}$.*

Now, given a most-relevant explanation ϵ and a user vocabulary $\mathcal{V}_h$, the next step is to compute an abstraction of ϵ using Definition 6. Ultimately, computing such an abstraction boils down to searching for the appropriate abstraction level for ϵ, i.e., the set of letters to forget from ϵ. Nevertheless, not all letters carry the same importance with respect to a knowledge base and a user vocabulary:

Definition 8 (Vocabulary Importance)**.** *Let KB be a knowledge base and $\lambda \subseteq \mathcal{V}_{KB}$, and $\mathcal{V}_h$ a user vocabulary. Define*

[4]Albeit a strong assumption, in this paper we suppose that the user is trying to learn from the agent, and as such, the agent encodes the correct information about a specific problem. We plan to relax this in the future work.

vocabulary importance as $\mathcal{VI}(KB, \mathcal{V}_h, \lambda) = \sum_{l \in \lambda} w * \frac{|KB_{\uparrow l}|}{|\mathcal{V}_{KB_{\uparrow l}}|}$, *where* $w > 1$ *if* $l \in \mathcal{V}_h$, *and* $w = 1$ *otherwise.*

The importance of letter λ with respect to a knowledge base KB and a user vocabulary $\mathcal{V}_h$ is then taken to be the number of formulae in KB that mention λ over the number of letters comprising those formulae. The constant w is used to assign higher importance to the letters that appear in $\mathcal{V}_h$. This intuitive definition is based on the premise that letters appearing in literals, i.e., facts, must have higher importance than letters appearing only in non-literal formulae. As such, the more letters in a formula, the less the importance of each individual letter in that formula.[5]

Example 3. *Let* $KB = \{a, \neg a \vee \neg b \vee c\}$, $\mathcal{V}_h = \{a\}$ *and* $w = 2$ *if* $\lambda \in \mathcal{V}_h$.*Then, the letter* a *yields* $\mathcal{VI}(KB, \mathcal{V}_h, \{a\}) = 2 * \frac{2}{3}$, *whereas* b *yields* $\mathcal{VI}(KB, \mathcal{V}_h, \{b\}) = \frac{1}{3}$.

Definition 8 can be used to generate *personalized explanations* by selecting and forgetting a set of letters from ϵ based on their importance in ϵ and $\mathcal{V}_h$. However, a personalized explanation should also be generated according to a desired size, as a novice user may require a longer explanation than, say, an expert one:

Definition 9 (Personalized Explanation)**.** *Let* ϵ *be a most-relevant explanation and* $\mathcal{V}_h$ *a user vocabulary. Then, for* $\lambda \subset \mathcal{V}_\epsilon$, $\mathcal{F}(\epsilon, \lambda)$ *is a personalized explanation iff* $|\mathcal{F}(\epsilon, \lambda)| \leq b_s$ *and* $\nexists \lambda' \subset \mathcal{V}_\epsilon$ *such that* $\mathcal{VI}(\epsilon, \mathcal{V}_h, \lambda') < \mathcal{VI}(\epsilon, \mathcal{V}_h, \lambda)$ *and* $|\mathcal{F}(\epsilon, \lambda')| \leq b_s$, *where* $2 \leq b_s < |\epsilon|$.

A personalized explanation is then a most-relevant explanation that has forgotten the least important set of letters according to Definition 8 and its size is bounded by a specified threshold. Note that the lower bound on the personalized explanation size ($2 \leq b_s$) is placed in order to avoid explanations of the form "why φ?", "because φ!", as such trivial explanation are not preferred in most cases.[6]

Example 4. *Let* $\epsilon = \{a, d, \neg d \vee b, \neg a \vee \neg b \vee c\}$ *be the most-relevant explanation with respect to* $\mathcal{V}_h = \{a\}$, *and* $b_s = 2$. *Then,* $\mathcal{F}(\epsilon, \{b, d\}) = \{a, \neg a \vee c\}$ *is a personalized explanation.*

4.1 Computing Personalized Explanations

We now present a simple greedy algorithm for computing personalized explanations for φ from KB_a with respect to $\mathcal{V}_h$ that is based by on an algorithmic procedure proposed in [Vasileiou *et al.*, 2021]. At a high level, the algorithm finds the most-relevant explanation (Definition 7) with respect to the user's vocabulary $\mathcal{V}_h$ through a weighted MUS procedure that prioritizes the formulae that consist of the most letters with respect to the user's vocabulary, and then computes the appropriate abstraction level for that explanation by finding the set of letters with the lowest vocabulary importance (Definition 8) such that when forgotten from the explanation, the explanation is at most the size of the specified bound b_s (Definition 9).

Algorithm 1 presents the pseudocode. The algorithm first weights the formulae in KB_a according to the number of intersections of their letters with those in $\mathcal{V}_h$ and inserts them into the knowledge base KB (Lines 3-5). The main loop starts at Line 6. At Line 7, the algorithm computes a weighted MUS ϵ on $KB \cup \{\neg\varphi\}$ by treating the formulae in KB as soft and the formula φ as hard. We remind the reader that the soft formulae are those formulae that will be removed by the optimization procedure of $weightedMUS$, while hard formulae will not. If $\epsilon \models \varphi$ (Line 8), then ϵ is a most-relevant explanation, otherwise the algorithm continues by computing a new MUS, where already computed MUSes are blocked in order to avoid infinite loops. When a most-relevant explanation ϵ is found, the algorithm proceeds to compute a personalized explanation (Lines 9-20). At Lines 9-10, the algorithm takes each letter in $\mathcal{V}_\epsilon$ and store it together with its vocabulary importance in $\mathcal{I}$. Then, starting from $N = 1$, it selects the N letters with the lowest vocabulary importance (Line 14)[7] and forgets them from ϵ (Line 15). If the newly generated explanation $\tilde{\epsilon}$ is less than or equal to the bound b_s, it returns it (Lines 16-17), otherwise it increases N by 1 (Line 19) and proceeds to select the next N smallest elements. Finally, if no explanation $\tilde{\epsilon}$ can satisfy the starting size bound b_s, the algorithm increases the bound by k and repeats Lines 12-19.

The completeness and correctness of Algorithm 1 rests on finding a most-relevant explanation, as given any explanation ϵ such that $\epsilon \models \varphi$, a personalized explanation for a properly defined bound b_s always exists (Definition 9). We sketch a proof by utilizing Proposition 1.

Theorem 1. *Algorithm 1 is complete and correct.*

Proof. (Completeness) First, Algorithm 1 always returns a solution, i.e., an explanation from KB_a for φ. Notice that since $KB_a \models \varphi$, then $KB_a \cup \{\neg\varphi\} \models \bot$, and thus, there exist a set of MUSes $\mathcal{U}$ from $KB_a \cup \{\neg\varphi\}$. From Proposition 1, $\exists u \in \mathcal{U}$ such that $\epsilon = u \setminus \{\neg\varphi\} \models \varphi$ is an explanation for φ. Since Algorithm 1 computes all possible MUSes from $KB_a \cup \{\neg\varphi\}$ (Line 7), it will eventually find and return an explanation ϵ.

(Correctness) Algorithm 1 is guaranteed to return a personalized explanation given $KB_a \models \varphi$, $\mathcal{V}_h \subseteq \mathcal{V}_{KB_a}$, and b_s. This is due to the fact that it uses a $weightedMUS$ function for computing explanations.[8] Firstly, the algorithm creates a weighted knowledge base KB (Lines 3-5), where the weights of the formulae in KB denote how many of their letters are in $\mathcal{V}_h$. The $weightedMUS$ function on Line 7 uses an implicit hitting set process (see the work by [Ignatiev *et al.*, 2015] for more) by iteratively building up MUSes from $KB \cup \{\neg\varphi\}$, where its optimization function maximizes the weights and minimizes the cardinality of the com-

[5]Note that our framework will be applicable with other definitions of vocabulary importance. For example, a possible definition may consider weights on the individual letters, where each weight will signify the importance of the letter.

[6]There might be cases where we need to explain an assumption or a fact, and therefore, trivial explanations will be succinct and acceptable.

[7]Note that vocabulary importance function is cumulative.

[8]The $weightedMUS$ procedure finds the MUS with the highest cost, where the cost is the sum of the weights of the formulae in the MUS.

Algorithm 1: Personalized Explanation Generation

Input: Agent knowledge base KB_a, explanandum φ, user vocabulary $\mathcal{V}_h$, initial size bound b_s, and increment k

Result: A *personalized explanation* $\tilde{\epsilon}$

$KB \leftarrow \{\emptyset\}$
$\mathcal{I} \leftarrow \{\emptyset\}$
for $\phi \in KB_a$ **do**
  $w = \frac{|\mathcal{V}_\varphi \cap \mathcal{V}_h|}{|\mathcal{V}_\varphi|}$
  $KB \leftarrow KB \cup \{(\phi, w)\}$
while *true* **do**
  $\epsilon \leftarrow weightedMUS(KB^s \cup \{\neg\varphi^h\})$
  if $\epsilon \models \varphi$ **then**
    for $\lambda \in \mathcal{V}_\epsilon$ **do**
      $\mathcal{I} \leftarrow \mathcal{I} \cup \{\lambda, \mathcal{VI}(\epsilon, \mathcal{V}_h, \lambda)\}$
    while *true* **do**
      $N = 1$
      while $N \leq |\mathcal{I}|$ **do**
        $\lambda \leftarrow getNsmallest(\mathcal{I}, N)$
        $\tilde{\epsilon} \leftarrow \mathcal{F}(\epsilon, \lambda)$
        if $|\tilde{\epsilon}| \leq b_s$ **then**
          return $\tilde{\epsilon}$
        else
          $N = N + 1$
      $b_s = b_s + k$

puted MUSes. In other words, the optimization function of $weightedMUS$ is akin to our relevance metric described in Definition 7. This means that the algorithm evaluates candidate explanations ϵ by prioritizing those with the highest relevance. Therefore, if $\epsilon \models \varphi$ evaluates to true, then ϵ is a most-relevant explanation according to Definition 7. Consequently, a personalized explanation for a properly defined bound b_s (Definition 9) is guaranteed to be returned. □

5 Experimental Evaluation

We now present an experimental evaluation of Algorithm 1 on some sample instances from the SAT competition.[9] We ran the experiments on a MacBook Pro machine comprising an Intel Core i7 2.60 GHz processor with 16GB of memory. The time limit was set to 500s. Algorithm 1 was implemented in Python and integrates calls to a weighted MUS oracle through the PySAT toolkit [Ignatiev *et al.*, 2018]. We used our own implementation for the knowledge forgetting operation.[10]

In our experiments, we used the SAT instances as the agent's knowledge base KB_a. The explanandum φ we used for each instance was a conjunction of backbone literals (e.g., a set of literals entailed by KB_a), which we pre-computed using the minibones algorithm [Janota *et al.*, 2015]. To generate user vocabularies $\mathcal{V}_h$, we created four scenarios, in which $\mathcal{V}_h$ consisted of a random selection of $5\%, 10\%, 15\%$, and 20% of the letters from the agent's vocabulary $\mathcal{V}_{KB_a}$ (Scenarios 1 to 4, respectively). Further, we placed a starting upper bound b_s on the explanation size of $10, 20, 30$, and 40 for Scenarios 1 to 4, respectively, and used $k = 2$ to increase b_s if the starting bound could not be satisfied.

[9] www.satcompetition.org

[10] The code and benchmarks can be found here https://github.com/vstylianos/explanations-via-knowledge-forgetting

Prob.		Scenario 1			Scenario 2			Scenario 3			Scenario 4		
		$\lvert\mathcal{V}_h\rvert$	$\lvert\tilde{\epsilon}\rvert$	ALG1	$\lvert\mathcal{V}_h\rvert$	$\lvert\tilde{\epsilon}\rvert$	ALG1	$\lvert\mathcal{V}_h\rvert$	$\lvert\tilde{\epsilon}\rvert$	ALG1	$\lvert\mathcal{V}_h\rvert$	$\lvert\tilde{\epsilon}\rvert$	ALG1
BN	1	35	9	0.5s	71	20	0.5s	106	21	0.1s	142	21	0.05s
	2	34	9	0.04s	69	18	0.05s	104	18	0.05s	138	19	0.05s
	3	37	10	0.1s	75	20	0.2s	113	20	0.05s	151	20	0.05s
BLOCK WORLD	1	9	9	0.05s	18	20	0.5s	28	26	0.5s	37	30	0.3s
	2	21	10	0.5s	43	11	0.5s	63	19	0.5s	84	20	0.2s
	3	13	10	1.0s	27	15	0.5s	40	14	0.5s	54	16	1.0s
COMM	1	693	28	4.0s	1,386	28	4.5s	2,079	30	4.0s	2,772	40	4.5s
	2	776	48	5.5s	1,552	48	5.0s	2,392	48	5.0s	3,105	48	4.5s
	3	777	28	3.5s	1,554	28	3.0s	2,332	30	2.5s	3,109	40	3.0s
LOGI-STICS	1	35	7	0.2s	99	18	0.3s	196	26	0.4s	261	24	0.4s
	2	35	4	0.2s	99	9	0.4s	196	21	0.5s	261	29	0.55s
	3	15	5	0.1s	31	5	0.1s	47	5	0.1s	63	5	0.1s
ROVER	1	28	10	1.0s	56	20	0.5s	84	21	0.4s	112	21	0.5s
	2	21	10	0.2s	15	15	0.6s	65	15	0.3s	87	15	0.2s
	3	65	16	0.5s	177	27	0.6s	343	28	1.0s	453	37	1.5s
ACE	1	198	9	5.0s	397	18	3.5s	594	24	2.5s	792	35	1.0s
	2	594	8	10.0s	1,188	17	8.0s	1,782	25	5.5s	2,376	33	3.0s
	3	1,200	10	20.0s	2,400	20	16.5s	3,600	29	11.0s	4,800	36	9.0s

Table 1: Evaluation of ALG1 on SAT Competition Problem Instances.

Table 1 tabulates the results, where we report the cardinality of $\mathcal{V}_h$, the cardinality of the personalized explanation $|\tilde{\epsilon}|$ returned, and the runtimes of Algorithm 1, referred to as ALG1. We observe that ALG1 performed relatively well and managed to find a personalized explanation in a short amount of time. Moreover, we notice a decrease in the runtimes of ALG1 on each subsequent scenario, which suggests that the algorithm's efficiency is correlated to the generated explanation's size, i.e., the smaller the explanation we seek based on b_s the more letters the algorithm has to consider and forget in order to satisfy Definition 9, which in consequence increases its runtime. Finally, it is important to mention that, in general, finding MUSes is an inherently hard problem, as for instance extracting an MUS from a knowledge base is in $FP^{\Sigma_P^2}$ [Liberatore, 2005]. Now, part of the performance advantage in ALG1 lies in the effectiveness of the underlying SAT and MUS solvers. This also implies that any advancement in those solvers will automatically reflect in performance gains in our algorithm.

6 Discussion

In this paper, we developed a simple framework that is able to generate explanations at an appropriate abstraction level with respect to a user-defined vocabulary. These explanations can be thought of as a step towards creating more personalized and intelligible explanations for human users, that is, explanations that minimize the user's cognitive effort and maximize their cognitive effect (cf. relevance theory).

Noteworthy, within the context of human-aware planning, Sreedharan et al. [2021] has also considered the problem of generating explanations for users that understand the task at different levels of abstraction. Nonetheless, their approach is restricted to classical planning problems, whilst our approach can be used for any problem that can be encoded in a logical formalism (and for which satisfiability of sets is feasi-

ble). [11] More specifically, they assume that the user's model is part of an abstraction lattice that the agent has, where each node in the lattice is an abstracted model of the task that is produced by projecting out a set of state fluents. The agent then estimates the appropriate level of the user based on questions that the user asks and provides explanations consistent with this estimate. This is achieved by the user providing a foil set (i.e., a set of plans) that the agent uses to find a minimal set of models from the lattice consistent with the foil. Having an estimate of the user's model (which captures the user's expertise), the agent refutes the foil with an explanation that contains information about properties that are missing from the user's model. This method can be seen as a special case of model reconciliation, where rather than assuming an explicit user model, it assumes that the user model belongs in the set of possible models that corresponds to the various abstractions of the agent's model. In contrast, we eliminate the assumption of a user model and replace it with a user vocabulary containing user-specified terms about the task. Moreover, our method is able to alter the explanation itself at an appropriate abstraction level with respect to the vocabulary.

At the other end of the spectrum, we view this work as a necessary step towards realizing an interactive, multi-shot explanation generation scheme, where users will be able to interact with an agent in a dialogical fashion. For example, the personalized explanations presented here can serve as the information that instigates the dialogue between the user and the agent. Specifically, we conceptualize a framework consisting of an agent model $M_a = \langle KB_a, KB_h^a \rangle$, where KB_h^a is an approximation of the user's knowledge (initially empty or filled with domain-specific common knowledge) that is aimed to be updated through interactions from the user. For example, upon receiving the initial explanation from the agent, the user would have the option to request further clarification on the explanation by asking for more information, in which case the agent will increase the explanation's granularity, or for less information, in which case the agent will decrease the granularity. Another option for the user would be to provide information to the agent that refutes the explanation, i.e., to engage in an argumentative process. We plan to explore this option in future work. Then, once the user is satisfied from the interactions and the information provided by the agent, the agent will update the user's approximate knowledge base KB_h^a with the information accepted by the user, and as such a more accurate representation of the user's knowledge will be learned. We hypothesize that this direction will lead to the practical inception of MRP, and hope that it will assist in engendering trust between AI systems and human users, something that is pivotal in today's AI agenda.

Finally, we note that *explanation communication* is a crucial aspect of explanatory systems. In the context of Human-Aware AI Planning, it has been shown that *explanations as model reconciliation* serve an important and intuitive way of explaining decisions to users.[12] Indeed, the authors in [Zahedi *et al.*, 2019; Chakraborti *et al.*, 2019] have empirically demonstrated that users not only understand explanations in the form of model reconciliation but also believe that such explanations are necessary to explain the agent's decisions. On that premise, and given the logical nature of our framework, we ought to say that we do not aim at presenting explanations to users in a logical format. The final form of our explanations can be, for instance, translated in natural language before communicated to a user.

[11] Indeed, knowledge forgetting has been analyzed for various logics, such as predicate logic, modal logic, and answer set programming [Eiter and Kern-Isberner, 2019].

7 Conclusion

To conclude, we proposed a logic-based framework that given an agent knowledge base, an explanandum, and a user vocabulary, generates personalized explanations. Due to its logic-based nature, our approach has the additional advantage of being able to deal with tasks coming from different settings, so long as the tasks can be encoded into a logical formalism. In this paper, we showed its utility on propositional encodings. Concerning this paper's future work, we aim to extend our framework to more expressive logics, such as first-order logic, and thus capture more real-world problems. In addition, we plan to evaluate the real-world efficacy of the explanations generated by our framework through human user studies.

Acknowledgements

Stylianos Loukas Vasileiou, Ashwin Kumar, and William Yeoh are partially supported by the National Science Foundation (NSF) under award 1812619. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the sponsoring organizations, agencies, or the United States government.

References

[Chakraborti *et al.*, 2017] Tathagata Chakraborti, Sarath Sreedharan, Yu Zhang, and Subbarao Kambhampati. Plan explanations as model reconciliation: Moving beyond explanation as soliloquy. In *IJCAI*, pages 156–163, 2017.

[Chakraborti *et al.*, 2019] Tathagata Chakraborti, Sarath Sreedharan, Sachin Grover, and Subbarao Kambhampati. Plan explanations as model reconciliation–an empirical study. In *HRI*, pages 258–266, 2019.

[Chakraborti *et al.*, 2020] Tathagata Chakraborti, Sarath Sreedharan, and Subbarao Kambhampati. The emerging landscape of explainable automated planning & decision making. In *IJCAI*, pages 4803–4811, 2020.

[Delgrande, 2017] James P Delgrande. A knowledge level account of forgetting. *Journal of Artificial Intelligence Research*, 60:1165–1213, 2017.

[Ebbinghaus, 2013] Hermann Ebbinghaus. Memory: A contribution to experimental psychology. *Annals of neurosciences*, 20(4):155, 2013.

[12] Unsurprisingly, this has been suggested by various psychological theories [Lombrozo, 2012; Hilton, 2017].

[Eiter and Kern-Isberner, 2019] Thomas Eiter and Gabriele Kern-Isberner. A brief survey on forgetting from a knowledge representation and reasoning perspective. *KI-Künstliche Intelligenz*, 2019.

[Hilton, 2017] Denis Hilton. Social attribution and explanation. 2017.

[Ignatiev *et al.*, 2015] Alexey Ignatiev, Alessandro Previti, Mark H. Liffiton, and João Marques-Silva. Smallest MUS extraction with minimal hitting set dualization. In *CP*, pages 173–182, 2015.

[Ignatiev *et al.*, 2018] Alexey Ignatiev, Antonio Morgado, and Joao Marques-Silva. PySAT: A Python toolkit for prototyping with SAT oracles. In *SAT*, pages 428–437, 2018.

[Janota *et al.*, 2015] Mikoláš Janota, Inês Lynce, and Joao Marques-Silva. Algorithms for computing backbones of propositional formulae. *AI Communications*, 28(2):161–177, 2015.

[Kambhampati, 2019] Subbarao Kambhampati. Synthesizing explainable behavior for human-ai collaboration. In *Proceedings of AAMAS*, pages 1–2, 2019.

[Liberatore, 2005] Paolo Liberatore. Redundancy in logic i: Cnf propositional formulae. *Artificial Intelligence*, 163(2):203–232, 2005.

[Lombrozo, 2012] Tania Lombrozo. Explanation and abductive inference. 2012.

[Lutz and Wolter, 2011] Carsten Lutz and Frank Wolter. Foundations for uniform interpolation and forgetting in expressive description logics. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[Marques-Silva, 2012] Joao Marques-Silva. Computing minimally unsatisfiable subformulas: State of the art and future directions. *Journal of Multiple-Valued Logic & Soft Computing*, 19, 2012.

[Sreedharan *et al.*, 2018] Sarath Sreedharan, Tathagata Chakraborti, and Subbarao Kambhampati. Handling model uncertainty and multiplicity in explanations via model reconciliation. In *ICAPS*, pages 518–526, 2018.

[Sreedharan *et al.*, 2021] Sarath Sreedharan, Siddharth Srivastava, and Subbarao Kambhampati. Using state abstractions to compute personalized contrastive explanations for ai agent behavior. *Artificial Intelligence*, 2021.

[Vasileiou *et al.*, 2021] Stylianos Loukas Vasileiou, Alessandro Previti, and William Yeoh. On exploiting hitting sets for model reconciliation. In *AAAI*, 2021.

[Vasileiou *et al.*, 2022] Stylianos Loukas Vasileiou, William Yeoh, Tran Cao Son, Ashwin Kumar, Michael Cashmore, and Daniele Magazzeni. A logic-based explanation generation framework for classical and hybrid planning problems. *Journal of Artificial Intelligence Research*, 73:1473–1534, 2022.

[Wang *et al.*, 2014] Yisong Wang, Yan Zhang, Yi Zhou, and Mingyi Zhang. Knowledge forgetting in answer set programming. *Journal of Artificial Intelligence Research*, 2014.

[Wilson and Sperber, 2002] Deirdre Wilson and Dan Sperber. Relevance theory, 2002.

[Zahedi *et al.*, 2019] Zahra Zahedi, Alberto Olmo, Tathagata Chakraborti, Sarath Sreedharan, and Subbarao Kambhampati. Towards understanding user preferences for explanation types in model reconciliation. In *HRI*, pages 648–649, 2019.

[Zhang and Zhou, 2009] Yan Zhang and Yi Zhou. Knowledge forgetting: Properties and applications. *Artificial Intelligence*, 2009.