Research Problem

A robot needs to explain its decision to a human user. Questions: how to explain?

1. request
2. answer

? not according to my KB

Explanations/Info

Previous Work: Assumptions

- Planning models \( \pi \) (robot) and \( \eta \) (human) in PDDL.
- \( \pi \) is a plan wrt. \( \eta \), but not a plan wrt. \( \pi \).
- Consideration of properties of \( \pi \) (optimal plan)
- Robot is aware of \( \pi \).

Previous Work: Solution

- Defining explanation as \((e', e), e' \subseteq \pi \text{ and } e \subseteq \eta \text{ such that } \eta \models e' \cup \neg e' \models a\).
- Developing algorithms for computing explanations.

This Work

- General framework for explainable AI using answer set programming.
- Knowledge bases represented as logic programs and a literal \( q \) in the language of \( \pi \), such that \( \eta \models q \) and \( \pi \models \neg q \).
- Variety of applications: \( \pi \) and \( \eta \) are two variants of a problem that can be solved using ASP
  - Planning
  - Scheduling
  - Diagnosis
  - …

Results

- An operator called conditional update on logic programs
- \( \pi \) and \( \eta \)
- \( I \) is an answer set of \( \pi \) supporting \( q \)
  - \( \subseteq \pi \)
- Defining \( \pi \models \neg \lambda \) such that \( \pi \models \lambda \models q \)
- Algorithms for computing explanations
- Applications in answer set programming

Motivation for ✽

Example 1

Consider

\[
\pi = \{ a \leftarrow \} \quad \eta = \{ a \leftarrow \}
\]

\( \pi \) has a unique answer set \( I_\pi = \{ a \leftarrow \} \).

To explain \( b \) to the human, the robot needs to inform the human that the rule \( a \leftarrow \) exists. So, the update in this case is simply adding the justification to \( \pi \), i.e., \( \pi \models b \). \lambda \ models \( \lambda \) should result in \( \{ a \leftarrow b \} \).

In this case, we do not remove any rule from \( \pi \).

Example 2

Consider

\[
\pi = \{ a \leftarrow \neg b \}
\]

\( \pi \) has two answer sets \( I_\pi = \{ a \leftarrow \} \) and \( I_\eta = \{ b \leftarrow a \} \). Only \( I_\eta \) supports \( b \) and \( b \leftarrow a \) is the justification of \( b \) wrt. \( I_\eta \).

It is easy to see that simply adding \( \lambda \) to \( \pi \) will result in a program with the unique answer set \( \{ a \leftarrow \} \) which does not support \( b \).

So, we should have

\[
\pi \models \lambda \models \{ b \leftarrow \neg a \}
\]

\( \pi \models \lambda \models \{ \} \), should not contain any rule whose head does not belong to \( I_\eta \).

Example 3

\[
\pi = \{ b \leftarrow \neg a \} \quad \eta = \{ c \leftarrow \neg e \}
\]

\( \pi \) has a unique answer set \( I_\eta = \{ b \} \) and the unique justification for \( b \) is \( \pi \).

To remove \( e \leftarrow \neg c \),

\[
\pi \models \lambda \models \{ b \leftarrow \neg a \}
\]

\( \pi \models \lambda \models \{ \} \), results into \( \{ b \leftarrow \neg a \} \).

Conditional Update

Let \( \pi_1 \) and \( \pi_2 \) be two programs. Further, let \( I \) be an answer set of \( \pi_1 \) and \( \lambda \subseteq \pi_1 \). The conditional update of \( \pi_1 \) with respect to \( \lambda \) and \( I \) is the program \( \pi_1 \setminus \lambda \), denoted by \( \pi_1 \setminus \lambda \), where \( \pi_1 \setminus \lambda \) is the collection of rules from \( \pi_1 \) \setminus \lambda \) such that \( \lambda \models \pi_1 \setminus \lambda \) and \( I \models \pi_1 \setminus \lambda \).

If \( \pi_1 \setminus \lambda \) has a unique answer set \( \{ a \leftarrow \} \), then \( \pi_1 \setminus \lambda \models \{ a \leftarrow \} \).

Approach

Explained-Based on Conditional Update

A subprogram \( c \subseteq \pi \) is an explanation for \( q \) from \( \pi \) to \( \eta \), wrt. an answer set \( I \) of \( \pi \) (or an explanation for \( q \) wrt. \( I \)) if \( \pi \models \lambda \models q \).

Computing Explanations

Let \( \pi \) answer set supporting \( q \) of \( \pi \), define \( \Pi(\pi, I) \):

1. \( \Pi(\pi, I) \) contains the constraint \( c \leftarrow \neg q \),
2. for each \( c \in \Pi(\pi, I) \), \( \text{head}(c) \in I \) and \( I \models \text{body}(c) \),
3. \( \text{head}(c) \leftarrow \neg q \) is a rule in \( \Pi(\pi, I) \),
4. \( \text{body}(c) \leftarrow \text{rule of } \Pi(\pi, I) \),
5. \( \text{head}(c) \leftarrow \text{rule of } \Pi(\pi, I) \),
6. \( \text{body}(c) \leftarrow \text{rule of } \Pi(\pi, I) \).

Then \( \Pi(\pi, I) \models \) an explanation for \( q \) wrt. \( I \).

Algorithm Explanation \((\pi_0, \pi, \eta, q)\)

Input: Programs \( \pi_0, \pi, \eta \), atom \( q \)
Output: An explanation \( e \) for \( q \)

If \( \pi_0 \cup \{ \neg q \} \) has no answer set

Return nil

Let \( I \) be an answer set of \( \pi_0 \cup \{ \neg q \} \)

Compute \( \Pi(\pi, I) \)

Compute an answer set \( J \) of \( \Pi(\pi, I) \)

Compute \( e = \{ \text{head}(c) \leftarrow \text{body}(c) \} \)

Return \( \pi_0 \models E \models \pi \models \eta \)

Utility-Based Optimal Utility Explanations

Let \( U_{\pi_1}(\pi_2) \) be the utility of \( \pi_2 \) wrt. \( \pi_1 \).

The utility function is defined by \( U_{\pi_1}(\pi_2) \) as the minimal number of rules in \( \Pi(\pi_1, I) \).

Believability-Preferred Explanations

The believability score of an explanation \( e \) is defined by \( \text{score}(e) = \# \text{answer sets of } \pi \models \pi \models e \text{ where } e \neq \pi \text{ is true} \).

Conclusions

- Define conditional update operator ✽
- Utilize ✽ in explainable AI
- Define preferred explanations
- Propose algorithms for computing explanations