Model Reconciliation in Logic Programs*

Tran Cao Son[1], Van Nguyen[1], Stylianos Loukas Vasileiou[2], and William Yeoh[2]

[1] New Mexico State University, Las Cruces, NM 88003, USA
{tson,vnguyen}@cs.nmsu.edu
[2] Washington University in St. Louis, St. Louis, MO 63130, USA
{v.stylianos,wyeoh}@wustl.edu

Abstract. Inspired by recent research in explainable planning, we investigate the *model reconciliation* problem between two logic programs π_a and π_h, which represent the knowledge bases of an agent and a human, respectively. Given π_a, π_h, and a query q such that π_a entails q and π_h does not entail q (or π_a does not entail q and π_h entails q), the model reconciliation problem focuses on the question of how to modify π_h, by adding $\epsilon^+ \subseteq \pi_a$ to π_h and removing $\epsilon^- \subseteq \pi_h$ from π_h such that the resulting program $\hat{\pi}_h = (\pi_h \setminus \epsilon^-) \cup \epsilon^+$ has an answer set containing q (or has no answer set containing q). The pair (ϵ^+, ϵ^-) is referred to as a *solution* for the model reconciliation problem (π_a, π_h, q) (or $(\pi_a, \pi_h, \neg q)$). We prove that, for a reasonable selected set of rules $\epsilon^+ \subseteq \pi_a$ there exists a way to modify π_h such that $\hat{\pi}_h$ is guaranteed to credulously entail q (or skeptically entail $\neg q$). Given that there are potentially several solutions, we discuss different characterizations of solutions and algorithms for computing solutions for model reconciliation problems.

Keywords: Model Reconciliation; Explainable Planning; Answer Set Programming

1 Introduction

In several problems involving two (or more) agents[3] with different knowledge bases, the agents often discuss about the truth value of an atom. Frequently, the question about the truth value of q—an atom appearing in the knowledge bases of both agents—is raised by an agent, say A, to another one, say B. Facing this question, agent B could potentially inform agent A the reason, constructed using her knowledge, for the truth value of q. This method is reasonable if agents A and B share a knowledge base. When they have different knowledge bases, this method might no longer suitable. For example, in human-aware planning problems [3, 5, 12, 13], a planning agent may inform a human user that it has a plan α for achieving a given goal. However, α may not be a feasible plan from

* This research is partially supported by NSF grants 1757207, 1812619, 1812628, and 1914635.

[3] We discuss problems involving only two agents in this paper, but our approach could be generalized to multiple agents.

the human's perspective. To address this issue, research in explainable planning proposes the *model reconciliation problem*, where the goal is to reconcile some of the differences in the models of the agent and the human (i.e., informs the human what needs to be changed in her model) such that α is an optimal plan, often the minimal length plan, in the reconciled model of the human.

In this paper, we propose a generalization of the model reconciliation problem, introduced in [1], as follows: Given a logic program π_a of a robot and a logic program π_h of a human user such that π_a entails[4] an atom q (resp. does not entail q), the goal is to identify a pair of sub-programs $\epsilon^+ \subseteq \pi_a$ and $\epsilon^- \subseteq \pi_h$ such that $\hat{\pi}_h = \pi_h \setminus \epsilon^- \cup \epsilon^+$ will also entail q (resp. will also not entail q). We refer to this problem as the *model reconciliation in logic programs* (MRLP) problem.

We note that MRLP might appear similar to *strong equivalent program transformation* (e.g., [6]) and *logic program update* (e.g., [11]), both research topics that have been extensively studied by the logic programming community. It is worth pointing out that MRLP's goal is not to make π_a and π_h equivalent. For example, if x is an atom in the languages of both π_a and π_h, π_a entails x, π_h does not entail x, and $\epsilon = (\epsilon^+, \epsilon^-)$ satisfying $\hat{\pi}_h = \pi_h \setminus \epsilon^- \cup \epsilon^+$ entails q then ϵ is an explanation for the problem (π_a, π_h, q) even when $\hat{\pi}_h$ does not entail x, i.e., $\hat{\pi}_h$ is not equivalent to π_a. Comparing to logic programming update, MRLP first needs to identify ϵ and then modifies the human program π_h by deleting or adding rules; it does not change the remaining rules of π_h. We will discuss in more detail the differences between MRLP and logic program update later. In summary, the main contributions of this paper are:

- a generalization of the model reconciliation problem in explainable planning to define the MRLP and a method for solving MRLP problems;
- different characterization of solutions of a MRLP problem that can be used to comparing solutions; and
- an algorithm for computing solutions of a MRLP problem.

The paper is organized as follows. The next section includes a short review of logic programming under answer set semantics and the notion of a justification for an atom with respect to an answer set that will be useful for later discussion. We then propose a general method for solving MRLP problems and discuss different ways to characterize a solution of a MRLP problem. Afterwards, we present algorithms for computing solutions of a given MRLP.

2 Background: Answer Set Programming

Answer set programming (ASP) [7, 9] is a declarative programming paradigm based on logic programming under the answer set semantics. A logic program Π is a set of rules of the form $a_0 \leftarrow a_1, \ldots, a_m,$ *not* $a_{m+1}, \ldots,$ *not* a_n where $0 \leq m \leq n$, each a_i is an atom of a propositional language and *not* represents

[4] In this paper, whenever we say a program entails a literal, we refer to the *credulous entailment* relationship between a program a literal. Precise definition will be provided in the next section.

(default) negation. Intuitively, a rule states that if all positive literals a_i are believed to be true and no negative literal $not\, a_i$ is believed to be true, then a_0 must be true. If a_0 is omitted, the rule is called a *constraint*. If $n = 0$, it is called a *fact*. For a rule r, $head(r)$ denotes a_0; $pos(r)$ and $neg(r)$, referred to as the *positive* and *negative* body, respectively, denotes the set $\{a_1, \ldots, a_m\}$ and $\{a_{m+1}, \ldots, a_n\}$, respectively. Also, $atoms(r)$ denotes the set of all atoms in r, viz. $\{head(r)\} \cup pos(r) \cup neg(r)$; and, $atoms(\Pi)$ denotes the set of all atoms of Π. $heads(\Pi)$ (resp. $negs(\Pi)$) denotes the set of atoms occurring in the head of rules of Π (resp. negative literals of Π).

Let Π be a program. $I \subseteq atoms(\Pi)$ is called an interpretation of Π. For an atom a, a is satisfied by I, denoted by $I \models a$, if $a \in I$. A set of atoms S is satisfied by I if $S \subseteq I$. For a rule r, $I \models body(r)$ if $pos(r) \subseteq I$ and $neg(r) \cap I = \emptyset$. A rule r is satisfied by I if $I \not\models body(r)$ or $I \models head(r)$. I is a *model* of a program if it satisfies all its rules. An atom a is *supported* by I in Π if there exists $r \in P$ such that $head(r) = a$ and $I \models body(r)$.

For an interpretation I and a program Π, the *reduct* of Π w.r.t. I (denoted by Π^I) is the program obtained from Π by deleting *(i)* each rule r such that $neg(r) \cap I \neq \emptyset$, and *(ii)* all negative literals in the bodies of the remaining rules. Formally, $P^I = \{head(r) \leftarrow pos(r) \mid r \in \Pi,\ neg(r) \cap I = \emptyset\}$. Given an interpretation I, observe that the program Π^I is a definite program (a program with no occurrence of negative literals). An interpretation I is an *answer set* [4] of Π if I is the least model of Π^I [2], which is the least fixpoint of the operator T_Π defined by $T_\Pi(I) = \{a \mid \exists r \in \Pi, head(r) = a, I \models body(r)\}$ and is denoted by $lfp(T_\Pi)$.

Given an answer set I of Π and an atom q, a justification for q w.r.t. I is a set of rules $S \subseteq \Pi$ such that $head(r) \in I$ and $I \models body(r)$ for $r \in S$ and $q \in lfp(T_{S^I})$. A justification S for q w.r.t. I is minimal if there exists no proper subset $S' \subset S$ such that S' is also a justification for q w.r.t. I. It is easy to see that if S is a minimal justification for q w.r.t. I then $negs(S) \cap heads(S) = \emptyset$ and $heads(S)$ is an answer set of S.

Given a logic program Π, an atom a. We write $\Pi \mid\!\sim a$ to indicate that a belongs to at least one answer set of Π or a is credulously entailed by Π. Furthermore, we use $\Pi \not\mid\!\sim a$ to indicate that a does not belong to any answer set of Π or $\neg a$ is cautiously entailed by Π.

3 Model Reconciliation in Logic Programs

The model reconciliation problem in logic programs (MRLP) is divided into two sub-problems, one aims at changing the human program so that it entails an atom (e-MRLP) and another focuses on achieving that the updated program does not entail an atom (n-MRLP). Inspired by the problem in explainable planning, we define three different types of MLRP.

Definition 1 (MRLP). *Let π_a and π_h be two logic programs and q be an atom in the language of π_a.*

- *The problem of* model reconciliation for entailment in logic programs *(e-MRLP) is defined by a triple (π_a, π_h, q). A pair of programs (ϵ^+, ϵ^-) such that $\epsilon^+ \subseteq \pi_a$ and $\epsilon^- \subseteq \pi_h$ is a* solution *of (π_a, π_h, q) if $\hat{\pi}_h \mid\!\sim q$ where $\hat{\pi}_h = \pi_h \setminus \epsilon^- \cup \epsilon^+$.*
- *The problem of* model reconciliation for non-entailment in logic programs *(n-MRLP) is defined by a triple $(\pi_a, \pi_h, \neg q)$. A pair of programs (ϵ^+, ϵ^-) such that $\epsilon^+ \subseteq \pi_a$ and $\epsilon^- \subseteq \pi_h$ is a* solution *of $(\pi_a, \pi_h, \neg q)$ if $\hat{\pi}_h \not\mid\!\sim q$ where $\hat{\pi}_h = \pi_h \setminus \epsilon^- \cup \epsilon^+$.*
- *The general problem of* model reconciliation in logic programs *(MRLP) is defined by a triple (π_a, π_h, ω) where $\omega = \omega^+ \wedge \neg\omega^-$ and ω^+ (resp. ω^-) is a conjunction of atoms in π_a. (ϵ^+, ϵ^-) is a solution for the MRLP problem if it is a solution for (π_a, π_h, q) for each conjunct q in ω^+ and solution for $(\pi_a, \pi_h, \neg r)$ for each conjunct r in ω^-.*

We note that e-MRLP focuses on credulous entailment of atoms while n-MRLP on skeptical entailment of negation of atoms. This is because we are interested in applying the framework in situations utilizing answer set programming for problem solving. In this context, it is often the case that the existence (resp. non-existence) of an answer set, that contains a designated atom, indicating that the problem is solvable (resp. not solvable). The combination of e-MRLP and n-MRLP, as in the general MRLP, provides us way to express various types of problems. For example, the shortest plan model reconciliation problem in explainable planning can be expressed by the triple (π_a, π_h, G) where π_a and π_h are the logic programs encoding the planning problem of the agent and the human[5], respectively, and $G = goal(n) \wedge \neg goal(n-1) \wedge \ldots \wedge \neg goal(0)$ representing that the goal of the planning problem must be satisfied after the execution of n actions but it is unsatisfied after the execution of any arbitrary $k < n$ actions.

Observe that e-MRLP implicitly requires that $\hat{\pi}_h$ is consistent. On the other hand, this requirement is missing in n-MRLP. As we are often interested in the general MRLP problem, we will therefore interested in solutions of MRLP problems that guarantee the consistency of $\hat{\pi}_h$. To simplify the presentation, we will assume that given for a MRLP problem (π_a, π_h, ω), $\pi_a \mid\!\sim \omega^+$ and $\pi_a \not\mid\!\sim \omega^-$; for a e-MRLP problem $(\pi_a, \pi_h, q_1 \wedge \ldots \wedge q_k)$, $\pi_a \mid\!\sim q_i$ for $i = 1, \ldots, q_k$; and for a n-MRLP problem $(\pi_a, \pi_h, \neg q_1 \wedge \ldots \wedge \neg q_k)$, $\pi_a \mid\!\sim \neg q_i$ for $i = 1, \ldots, q_k$. Furthermore, we will discuss the solutions of e-MRLP or n-MRLP problems with a single atom q as the solutions for more complex formulas can be computed in the same manner.

We will first discuss how to solve n-MRLP problems. Obviously, if $\pi_h \not\mid\!\sim q$ then $(\emptyset, \emptyset)$ is a solution for $(\pi_a, \pi_h, \neg q)$. Now, assume that $\pi_h \mid\!\sim q$. By definition of answer sets, we can just remove rules from π_h to achieve $\hat{\pi}_h \not\mid\!\sim q$. Let $\pi_h(q) = \{r \mid r \in \pi_h, head(r) = q\}$. It is easy to see that $P \not\mid\!\sim q$ for every $P \subseteq \pi_h \setminus \pi_h(q)$. As such, a solution $(\emptyset, \epsilon^-)$ for the n-MRLP problem $(\pi_a, \pi_h, \neg q)$ that guarantees the consistency of $\hat{\pi}_h$ could be determined with $\pi_h(q) \subseteq \epsilon^- \subseteq \pi_h$. Observe that taking π_a into consideration provide alternative solutions as well. For example,

[5] Strictly speaking, π_a also encodes the shortest plan in explainable planning.

given the two programs:

$$\pi_a = \{a \leftarrow\} \quad \pi_h = \{q \leftarrow \ not\ c; c \leftarrow \ not\ q; a \leftarrow \ not\ a,\ not\ q\}$$

It is easy to see that π_h has a unique answer set $\{q\}$ and thus $\pi_h \mid\!\sim q$ and $\pi_h \setminus \pi_h(q)$ is inconsistent. On the other hand, either $(\pi_a, \pi_h(q))$ or $(\emptyset, \{q \leftarrow not\ c; a \leftarrow \ not\ a,\ not\ q\})$ is a solution for the n-MRLP problem $(\pi_a, \pi_h, \neg q)$. In either case, $\hat{\pi}_h$ is consistent. The former adds a rule from π_a and removes $\pi_h(q)$ from π_h while the latter only removes rules from π_h.

It should be noted that sometimes, there is no need to remove rules whose head is q to achieve that $\hat{\pi}_h \mid\!\not\sim q$. For example, for the program $\pi_h = \{q \leftarrow not\ c; c \leftarrow \ not\ d; d \leftarrow\}$ we have that $\pi_h \setminus \{d \leftarrow\} \mid\!\not\sim q$. The two examples show that there are several explanations for a n-MRLP problem. As we will see later, the same holds for e-MRLP problems. In our view, which explanation should be used is application dependent.

We now discuss a method for solving e-MRLP problems (π_a, π_h, q). By definition of answer sets, $\hat{\pi}_h \mid\!\sim q$ means that there exists an answer set of $\hat{\pi}_h$ which contains a justification for q. In all likelihood, this justification must come from π_a if $\pi_h \mid\!\not\sim q$. In other words, the justification for q in $\hat{\pi}_h$ should be a part of ϵ^+. For this reason, we will focus on how to choose ϵ^+. This can be done by identifying an answer set I supporting q and selecting a justification for q w.r.t. I as ϵ^+. A solution can then determined by identifying $\epsilon^- \subseteq \pi_h$ so that (ϵ^+, ϵ^-) is a solution to the problem (π_a, π_h, q). Assume that I and ϵ^+ have been selected, we motivate the selection of ϵ^- using a series of e-MRLP problems (π_a, π_h, b), i.e., the robot wants to explain to the human that b is entailed by his program.

Example 1. Let $\pi_a = \left\{ a \leftarrow; b \leftarrow a \right\}$ $\pi_h = \{a \leftarrow\}$ Clearly, π_a has a unique answer set $I_0 = \{a, b\}$ and $\epsilon^+ = \pi_a$ is a justification for b. To explain b to the human, the robot needs to inform the human that the rule $b \leftarrow a$ exists. Furthermore, there is no need to remove anything from π_h, i.e., $\epsilon^- = \emptyset$ *since the rule* $a \leftarrow$ *is satisfied by* I_0.

The example above discusses a situation in which one needs to add rules to the human's program as part of the explanation process. The next examples discuss different situations in which one needs to also remove rules from the human's program.

Example 2. Let $\pi_a = \{a \leftarrow \ not\ b; b \leftarrow \ not\ a\}$ and $\pi_h = \{a \leftarrow\}$. π_a has two answer sets $I_1 = \{a\}$ and $I_2 = \{b\}$. Only I_2 supports b and $\epsilon^+ = \{b \leftarrow \ not\ a\}$ is the justification of b w.r.t. I_2. It is easy to see that simply adding ϵ^+ to π_h will result in a program with the unique answer set $\{a\}$ which does not support b. It means that the rule $a \leftarrow$ should be removed, i.e., $\epsilon^- = \{a \leftarrow\}$. This suggests that ϵ^- should contain any rule whose head does not belong to I_2.

Example 3. Let $\pi_a = \{b \leftarrow \ not\ a\}$ $\pi_h = \{c \leftarrow \ not\ c\}$.
π_a has a unique answer set $I_3 = \{b\}$ and the unique justification for b is $\epsilon^+ = \pi_a$. The program $\pi_h \cup \pi_a$ is also inconsistent because of the rule $c \leftarrow \ not\ c$. So, we need to have $\epsilon^- = \{c \leftarrow \ not\ c\}$. Observe that in this case, the rule $r =$ "$c \leftarrow \ not\ c$" satisfies $head(r) \not\in I_3$ but $neg(r) \cap I_3 = \emptyset$.

Example 4. Let $\pi_a = \{b \leftarrow \ not\ a\} \quad \pi_h = \{\leftarrow b\}$.
π_a has a unique answer set $I_4 = \{b\}$ and the unique justification for b is $\epsilon^+ = \pi_a$. The program $\pi_h \cup \pi_a$ is also inconsistent because of the constraint $\leftarrow b$. So, we should set $\epsilon^- = \pi_h$. In this case, the constraint $r =$“ $\leftarrow b$” satisfies $head(r) \notin I_4$ but $pos(r) \subseteq I_4$.

Observe that in Example 1, the rule $a \leftarrow$ needs not to be removed since its body and head are both satisfied by the answer set $\{a, b\}$ which happens to be the answer set of the justification ϵ^+. In Example 2, the rule $a \leftarrow$ is removed because of its body is satisfied but its head is not satisfied by the answer set $\{b\}$. Although it appears differently, Examples 3–4 are similar to Example 2: The head of the rule is not satisfied and the body of the rule is satisfied by the answer set of the program ϵ^+. So, one might wonder whether there is any reasonable situation in which a rule, whose head is not satisfied by the answer set I, should be kept. Indeed, consider an example similar to Example 4, except that $\pi_h = \{\leftarrow c\}$. In this case, it would make sense not to remove the constraint $\leftarrow c$ because it is not falsified by the answer set I_4. The discussion above leads to the following notion that is useful for the computation of solutions of e-MRLP problems.

Definition 2 (Residual of a program w.r.t. a set of rules and a set of atoms). *Let π_a and π_h be two programs. Further, let I be a set of atoms of π_a and $\epsilon^+ \subseteq \pi_a$. The* residual *of π_h with respect to ϵ^+ and I, denoted by $\otimes(\pi_h, \epsilon^+, I)$, is the collection of rules from $\pi_h \setminus \epsilon^+$ such that for each rule $r \in \otimes(\pi_h, \epsilon^+, I)$:*
(i) $head(r) \in I$ and $neg(r) \cap I = \emptyset$; or
(ii) $neg(r) \cap heads(\epsilon^+) \neq \emptyset$; or
(iii) $pos(r) \setminus I \neq \emptyset$.
We use $\epsilon^-[\epsilon^+, I, \pi_h]$ to denote the set of rules $\pi_h \setminus \otimes(\pi_h, \epsilon^+, I)$.

It is easy to verify that if we use I and ϵ^+ as in Examples 1–4, then $(\epsilon^+, \epsilon^-[\epsilon^+, I, \pi_h])$ is a solution for the problem (π_a, π_h, b) in these examples.

Observe that Examples 1–4 are somewhat unique in that, for each answer set, there exists only one possible justification for the atom b. It is easy to see that there are situations in which multiple justifications for an atom are present. For example, consider

$$\pi_a = \left\{ a \leftarrow; b \leftarrow a; b \leftarrow \right\} \qquad \pi_h = \{a \leftarrow\}$$

In this case, π_a also has a unique answer set $I = \{a, b\}$. However, there are two possible ways for justifying the presence of b in the answer set: (1) $\epsilon_1^+ = \{a \leftarrow; b \leftarrow a\}$ and (2) $\epsilon_2^+ = \{b \leftarrow\}$. It is easy to see that for $i = \{1, 2\}$, $(\epsilon_i^+, \epsilon_i^-[\epsilon_i^+, I, \pi_h])$ is a solution for (π_a, π_h, b). A natural question is then which solution should be used? We believe that choosing which solution to present to the human is application dependent; for example, if b represents a fact in the initial state of a planning problem, using ϵ_2^+ is reasonable; on the other hand, if b is a derived fact and is dependent on a, using ϵ_1^+ would be more reasonable as it informs the human of the dependency between a and b, which could potentially be useful for the human.

The above discussion shows that solutions of the e-MRLP problem (π_a, π_h, q) can be computed by identifying I, ϵ^+, and then set $\epsilon^- = \epsilon^-[\epsilon^+, I, \pi_h]$. An appropriate choice of I and ϵ^+ is specified in the next theorem.

Theorem 1. *Let (π_a, π_h, q) be an e-MRLP problem. Further, let I be an answer set of π_a supporting q and $\epsilon^+ \subseteq \pi_a$ be a minimal justification of q w.r.t. I. Then, $(\epsilon^+, \epsilon^-[\epsilon^+, I, \pi_h])$ is a solution of (π_a, π_h, q).*

Proof. Let $P = \pi_h \setminus \epsilon^-[\epsilon^+, I, \pi_h] \cup \epsilon^+$. Let $K = \mathit{heads}(P) \cap I$. Let $P_1 = \{r \in P \mid \mathit{head}(r) \in I, \mathit{neg}(r) \cap K = \emptyset\}$. Clearly, $\epsilon^+ \subseteq P_1$. Furthermore, for each rule $r \in P_1$, $\mathit{neg}(r) \cap \mathit{heads}(P_1) = \emptyset$ since $\mathit{heads}(P_1) \subseteq K$. Therefore, P_1 is consistent and has a unique answer set J containing $\mathit{heads}(\epsilon^+)$ and $J \subseteq I$.

Consider $r \in P \setminus P_1$. We have that $\mathit{head}(r) \notin I$ or $\mathit{neg}(r) \cap K \neq \emptyset$. From Definition 2, we can conclude that $\mathit{neg}(r) \cap \mathit{heads}(\epsilon^+) \neq \emptyset$ or $\mathit{pos}(r) \setminus I \neq \emptyset$. This allows us to show that $P^J = P_1^J \cup R$ and, for every $r \in R$, $\mathit{pos}(r) \setminus J \neq \emptyset$. This implies that J is an answer set of P^J, i.e., $(\epsilon^+, \epsilon^-[\epsilon^+, I, \pi_h])$ is a solution of (π_a, π_h, q). □

It is easy to see that the following holds:

Corollary 1. *For an e-MRLP problem (π_a, π_h, q), if there exists a non-trivial justification $\epsilon^+ \subset \pi_a$ w.r.t. an answer set I of π_a, then it has a non-trivial solution.*

3.1 Computing Solutions of MRLP Problems Using ASP

We will conclude the section with a discussion on how a solution for a general MRLP problem can be constructed. Without loss of generality, assume that we have the problem $(\pi_a, \pi_h, q \wedge \neg r)$ where q and r are atoms of π_a. Recall that we assume that $\pi_a \mid\!\sim q$ and $\pi_a \not\mid\!\sim r$ in this problem. A solution (ϵ^+, ϵ^-) for $(\pi_a, \pi_h, q \wedge \neg r)$ can be computed by the following steps: (i) compute an answer set I of π_a that supports q and identify a minimal justification ϵ^+ of q w.r.t. I; (ii) compute $\epsilon^- = \epsilon^-[\epsilon^+, I, \pi_h]$; (iii) identify a set of rules λ from $\pi' = \pi_h \setminus \epsilon \cup \epsilon^+$ so that $\pi' \setminus \lambda \not\mid\!\sim r$. The final solution for $(\pi_a, \pi_h, q \wedge \neg r)$ is then $(\epsilon^+, \epsilon^- \cup \lambda)$. Note that because ϵ^+ is a justification for q, $\epsilon^+ \not\mid\!\sim r$ holds. Therefore, λ always exists and Theorem 1 shows that the problem $(\pi_a, \pi_b, q \wedge \neg r)$ always has some solution.

Given a program π_a and an answer set I supporting ω^+ of π_a, let $\Pi(\pi_a, I)$ be the program such that:

- $\Pi(\pi_a, I)$ contains the constraint $\leftarrow \; \mathit{not}\ q$, for each $q \in \omega^+$.
- For each $x \in \pi_a$ s.t. $\mathit{head}(x) \in I$ and $I \models \mathit{body}(x)$:
  - $\mathit{head}(x) \leftarrow \mathit{pos}(x), \mathit{neg}(x), \mathit{ok}(x)$ is a rule in $\Pi(\pi_a, I)$.
  - $\{\mathit{ok}(x)\} \leftarrow$ is a rule of $\Pi(\pi_a, I)$.
  - $\#\texttt{mimimize}\{1, X : \mathit{ok}(X)\}$ is a rule of $\Pi(\pi_a, I)$.
- No other rule is in $\Pi(\pi_a, I)$.

```
Algorithm 1: solve($\pi_a, \pi_h, \omega$)
  Input: Programs $\pi_a$, $\pi_h$, conjunction $\omega$
  Output: a solution $(\epsilon^+, \epsilon^-)$ for $(\pi_a, \pi_h, \omega)$
1 Let $I$ be an answer set of $\pi_a \cup \{\leftarrow\ not\ q \mid q \in \omega^+\}$
2 Compute $\Pi(\pi_a, I)$
3 Compute an answer set $J$ of $\Pi(\pi_a, I)$
4 Compute $\epsilon^+ = \{head(r) \leftarrow pos(r), neg(r) \mid head(r) \leftarrow pos(r), neg(r), ok(r) \in$
    $\Pi(\pi_a, I)$ and $ok(r) \in J\}$.
5 Let $\lambda_0 = \{r \mid r \in \pi_h \setminus \epsilon^-[\epsilon^+, I, \pi_h]$ and $head(r) \in \omega^-\}$
6 Identify a set $\lambda_0 \subseteq \lambda \subseteq \pi_h \setminus \epsilon^-[\epsilon^+, I, \pi_h]$ such that $\pi_h \setminus (\epsilon^-[\epsilon^+, I, \pi_h] \cup \lambda) \cup \epsilon^+$
    is consistent
7 return $(\epsilon^+ \setminus \pi_h, \epsilon^-[\epsilon^+, I, \pi_h] \cup \lambda)$
```

We next present an algorithm which uses $\Pi(\pi_a, I)$ for generating solutions of a MRLP problem (π_a, π_h, ω).

Recall that we assume that $\pi_a \mid\!\sim \omega^+$ and $\pi_a \not\mid\!\sim \omega^-$ in this paper. Otherwise, the algorithm needs to check for the two conditions (i) $\pi_a \mid\!\sim \omega^+$, i.e., whether π_a has answer set satisfying ω^+; and (ii) $\pi_a \not\mid\!\sim \omega^-$, i.e., whether π_a has any answer set satisfying any atom occurring in ω^- before continues with the first line. The correctness of the algorithm is proved in Proposition 1 (below) and the fact that all rules whose head occurring in ω^- are removed (Line 4–5).

Proposition 1. *Given a MRLP problem (π_a, π_h, ω) and I is an answer set of π_a supporting ω^+. Let J be an answer set of $\Pi(\pi_a, I)$ and ϵ^+ be the collection of rules:*

$$\left\{ head(r) \leftarrow pos(r), neg(r) \;\middle|\; \begin{array}{l} head(r) \leftarrow pos(r), neg(r), ok(r) \in \Pi(\pi_a, I) \wedge \\ ok(r) \in J \end{array} \right\}$$

Then, $J \setminus \{ok(x) \mid x \text{ is a rule in } \pi_a\} \subseteq I$ and $(\epsilon^+, \epsilon^-[\epsilon^+, I, \pi_h])$ is a solution for (π_a, π_h, ω^+).

Proof. (Sketch) The proof of this proposition relies on the following observation: (i) $J \setminus \{ok(x) \mid x \text{ is a rule in } \pi_a\} \subseteq I$ follows immediately from the definition of $\Pi(\pi_a, I)$; (ii) J must contain q, for $q \in \omega^+$, due to the constraint "$\leftarrow\ not\ q$"; (iii) the minimization statement ensures that J is a set with minimal number of rules satisfying ω^+; and the fact that $q \in J$ for $q \in \omega^+$ implies that ϵ^+ is indeed a minimal justification for ω^+ w.r.t. I and, hence, $(\epsilon^+, \epsilon^-[\epsilon^+, I, \pi_h])$ is a solution for (π_a, π_h, ω^+). □

4 Characterizing Solutions

As we have discussed earlier, a MRLP might have several solutions and choosing a suitable solution is application dependent. We now discuss some characteristics of solutions that could influence the choice.

Definition 3. *Let (π_a, π_h, ω) be an MRLP problem and (ϵ^+, ϵ^-) be a solution of (π_a, π_h, ω). We say:*

- (ϵ^+, ϵ^-) *is* optimal *if there exists no solution* (λ^+, λ^-) *such that* $\lambda^+ \cup \lambda^- \subset \epsilon^+ \cup \epsilon^-$*.*
- (ϵ^+, ϵ^-) *is* π-restrictive *for* $\pi \subseteq \pi_a$ *if* $\epsilon^+ \subseteq \pi$*; it is* minimally-restrictive *if there exists no solution* (λ^+, λ^-) *such that* $\lambda^+ \subset \epsilon^+$*.*
- (ϵ^+, ϵ^-) *is* π-preserving *for* $\pi \subseteq \pi_h$ *if* $\pi \cap \epsilon^- = \emptyset$*; it is* maximally-preserving *if there exists no solution* (λ^+, λ^-) *such that* $\lambda^- \subset \epsilon^-$*.*
- (ϵ^+, ϵ^-) *is* assertive *if every answer set of* $\pi_h \setminus \epsilon^- \cup \epsilon^+$ *satisfies* ω^+*.*
- (ϵ^+, ϵ^-) *is a* solution with justification *(or* j-solution*) if* ϵ^+ *contains a justification for* ω^+ *w.r.t. some answer set* I *of* π_a*.*

Each class of solutions has its own merits and could be useful in different situations. Optimal solutions could be useful when solutions are associated with some costs. Minimally-restrictive solutions focus on minimizing the amount of information that the robot needs to introduce to the human. They will be useful when explaining a new rule is expensive. On the other hand, maximally-preserving solutions is appropriate when one seeks to minimize the amount of information that needs to be removed from the human knowledge. Solutions with justifications are those that come with their own support. Assertive solutions do not leave the human any reason for questioning the atom in discussion. In Examples 1-4, we can see that the solution in Example 1 is not optimal but all others are optimal, minimally-restrictive and maximally-preserving, and solutions with justification. We make the following observations:

Observation 1 • *A minimal solution always exists. Similarly, a minimally-restrictive (resp. maximally-preserving) solution always exists.*

- *If a solution is minimally-restrictive and maximally-preserving, then it is optimal.*
- *For some* π*, there exists no* π*-preserving solution. For example, in Example 2, a* π*-preserving solution does not exist for* $\pi = \{a \leftarrow\}$*. Likewise, for some* π *(e.g.,* $\pi = \emptyset$*), there exists no* π*-restrictive solution.*
- *Not every solution of an MRLP problem is a j-solution. For example,* $(\{b \leftarrow a\}, \emptyset)$ *is not a j-solution for the problem* $(\{a \leftarrow; b \leftarrow a; b \leftarrow\}, \{a \leftarrow\}, b)$*.*
- *Not every solution of an MRLP is assertive. For example,* $(\{b \leftarrow \textit{not } a\}, \emptyset)$ *is not an assertive solution for the problem* $(\{a \leftarrow \textit{not } b; b \leftarrow \textit{not } a\}, \{a \leftarrow \textit{not } b\}, b)$*.*

While it is natural to think of optimal solutions, there exists subprogram π of π_h such that π-preserving solutions are reasonable. For example, it is reasonable to consider solutions that is $(\pi_a \cap \pi_h)$-preserving since $\pi_a \cap \pi_h$ represents the *common knowledge* between the robot and the human. Examples 2-4 show that, for some π, there might not exists a π-preserving solution (i.e., for $\pi = \{a \leftarrow\}$ in Example 2, a π-preserving solution does not exist). Theorem 1 shows that j-solutions can be constructed from an answer set I of π_a that supports q. It is easy to see that not every solution of the problem must be a j-solution. For example, $(\{b \leftarrow a\}, \emptyset)$ is not a j-solution for the problem $(\{a \leftarrow; b \leftarrow a; b \leftarrow\}, \{a \leftarrow\}, b)$.

4.1 Cost-Based Characterization

An alternative for characterizing solutions is to associate a cost to a solution (ϵ^+, ϵ^-) and use it as a means to compare solutions. In this paper, we are interested in the following cost functions.

Definition 4 (Cost Function and Cost-Optimal Solutions). *A* cost function *of an MRLP problem (π_a, π_h, ω) is a function $\mathcal{C}$ that maps each rule of $\pi_a \cup \pi_h$ to a non-negative number: $\mathcal{C} : \pi_a \cup \pi_h \rightarrow \mathcal{R}^{\geq 0}$.*

The cost of a solution (ϵ^+, ϵ^-) w.r.t. $\mathcal{C}$, denoted by $\mathcal{C}(\epsilon^+, \epsilon^-)$, is then defined as $\Sigma_{r \in \epsilon^+ \cup \epsilon^-} \mathcal{C}(r)$.

Given a cost function $\mathcal{C}$, a solution (ϵ^+, ϵ^-) is cost optimal *w.r.t. $\mathcal{C}$ if $\mathcal{C}(\epsilon^+, \epsilon^-)$ is minimal among all solutions.*

We define some special cost functions as follows. $\mathcal{C}$ of (π_a, π_h, q) is:

1. *uniform* if $\mathcal{C}(r) = c$ for each rule $r \in \pi_a \cup \pi_h$, where $c > 0$ is a constant.
2. *agent-biased* if $\mathcal{C}(r) = c$ for each rule $r \in \pi_a$, where $c > 0$ is a constant, and $\mathcal{C}(r) = 0$ for each rule $r \in \pi_h$.
3. *human-biased* if $\mathcal{C}(r) = 0$ for each rule $r \in \pi_a$ and $\mathcal{C}(r) = c$, where $c > 0$ is a constant, for each rule $r \in \pi_h$.

Because minimality in cardinality of a set implies minimality with respect to the subset relation, we can easily prove the following.

Proposition 2. *Given a cost function $\mathcal{C}$:*

- *If it is uniform, then a cost-optimal solution w.r.t. $\mathcal{C}$ is optimal (as in Definition 3).*
- *If it is agent-biased, then a cost-optimal solution w.r.t. $\mathcal{C}$ is minimally-restrictive.*
- *If it is human-biased, then a cost-optimal solution w.r.t. $\mathcal{C}$ is maximally-preserving.*

Observe that more general or specific cost functions could be defined and used to compare solutions. More specifically, a cost function discussed above is rule-based. A more specific one could be an atom-level cost function that assigns each atom some cost. A more general one is a solution-level cost function that assigns each solution a cost. While all are theoretically reasonable, we believe that a rule-based cost function is more appropriate because each rule is supposed to encode a piece of knowledge from each agent (robot or human). Alternatively, preferences among atoms that could be added or should be removed can be defined and used in determining most preferred solutions. We will leave this for the future work.

4.2 Assertiveness Characterization

We now propose an alternative perspective that is orthogonal to the characterization defined above. Given an MRLP problem (π_a, π_h, ω), the goal of the robot in providing a solution ϵ is to convince the human that ω is true given its knowledge base. Thus, the success of this process depends on *how much the human believes the solution presented by the robot*. Following this line of thought, we define the notion of an *assertive score* for solutions:

Definition 5 (Assertive Score). *The* assertive score *of a solution* (ϵ^+, ϵ^-) *of an MRLP problem* (π_a, π_h, ω) *is:*

$$S(\epsilon^+, \epsilon^-) = \frac{\#\textit{answer sets of } \pi_h \setminus \epsilon^- \cup \epsilon^+ \textit{ where } \omega^+ \textit{ is true}}{\#\textit{answer sets of } \pi_h \setminus \epsilon^- \cup \epsilon^+}$$

A solution (ϵ^+, ϵ^-) *is* assertive-score-maximal *if* $S_{(\pi_h, q)}(\epsilon)$ *is maximal among all solutions.*

Intuitively, $S(\epsilon^+, \epsilon^-)$ represents the probability of the human believing the solution. As we have remarked earlier, $S(\epsilon^+, \epsilon^-)$ is always positive (cf. Theorem 1). The last bullet in Observation 1 shows that can be less than 1. We can prove the following proposition that certain solutions are assertive.

Proposition 3. *For a MRLP problem* (π_a, π_h, ω)*. Assume that* I *is an answer set of* π_a *and* ϵ^+ *is a minimal justification of* ω^+ *w.r.t.* I*. If the residual of* π_h *w.r.t.* ϵ^+ *and* I *contains only definite rules then there exists a solution* (ϵ^+, ϵ^-) *for* (π_a, π_h, ω) *with* $\epsilon^-[\epsilon^+, I, \pi_h] \subseteq \epsilon^-$ *such that* $S(\epsilon^+, \epsilon^-) = 1$*.*

Proof. Let $P = \pi_h \setminus \epsilon^-[\epsilon^+, I, \pi_h] \cup \epsilon^+$. Because $P \setminus \epsilon^+$ is a positive program, we have that $negs(P) = negs(\epsilon^+)$. As such $negs(P) \cap heads(P) = \emptyset$. Hence, any answer set X of P would satisfy that $X \cap negs(P) = \emptyset$. This implies that P has a unique answer set satisfying ω^+. To obtain a solution for (π_a, π_h, ω), we can remove the set λ of rules whose heads occur in ω^- from P. The remaining program $P \setminus \lambda$ is a positive program and entails ω^+. This shows that $S(\epsilon^+, \epsilon^-[\epsilon^+, I, \pi_h] \cup \lambda) = 1$. □

5 Related Work and Discussions

The paper takes inspiration from the discussion in explainable planning (XAIP) [1, 12, 13] and generalizes it to define MRLP problems. Solutions to a MRLP problem could be viewed as explanations defined in XAIP. It is therefore closely related to the recent paper [8]. Both [8] and this paper employ ASP as the underlying representation language. However, [8] focuses on the development of an ASP-based system for solving XAIP problems while the present work emphasizes the knowledge representation aspects of a generalization of XAIP. This difference in focus leads to the fact that the algorithms proposed in this paper are general in that they are applicable in different classes of problems representable by logic programs and are not as specific as the ones developed in [8]. Furthermore, [8] does not include any characterizations of the solutions of MRLP problems as discussed in this paper.

It is worth noticing that Definition 2 appears to define an update operator to a program π_h with a set of rules ϵ^+ and a set of atoms I. This operator, however, differs from *all* update operators defined in the vast literature on logic programming updates (see, e.g., the survey by [11]). In earlier operators, the inputs are two programs π_h and ϵ^+, and the resulting program $\pi_h \oplus \epsilon^+$ should include ϵ^+ and *retain as much as possible from* π_h or satisfy certain postulates related to belief revision (e.g., the AGM postulates). This is because update models are defined for revising the beliefs of an agent π_h when some new information ϵ^+ arrives. There is no consideration of the third parameter I and there is no requirement that $\pi_h \oplus \epsilon^+ \mid\!\sim \omega^+$ even if ϵ^+ satisfies the conditions in Theorem 1. We note that the idea of eliminating rules in π_h that are "conflicting" with the new rules ϵ^+, presented by [15] and later by [14], could potentially be useful. However, the operator in this work adds rules that are not in $\pi_h \cup \epsilon^+$ to the resulting program.

Last but not least, we observe that Algorithm 1 only computes j-solutions for an MRLP problem. It is easy to see that an an arbitrary solution (ϵ^+, ϵ^-) for (π_a, π_h, ω) could be computed by randomly selecting $\epsilon^+ \subseteq \pi_a$ and $\epsilon^- \subseteq \pi_h$ and testing whether (ϵ^+, ϵ^-) is a solution for the problem, i.e., verifying $\pi_h \setminus \epsilon^- \cup \epsilon^+ \mid\!\sim \omega^+$ and $\pi_h \setminus \epsilon^- \cup \epsilon^+ \not\mid\!\sim \omega^-$. This idea is similar to the proposed method of computing explanations of abductive logic programs discussed by [10] . Although this idea is simple and generic, we observe that it can only be applied whenever the symmetric difference between π_a and π_h is small and thus is not practically useful.

It is worth noting that different methods proposed in the literature for computing a justification (sometimes referred to as explanation) for an atom (set of atoms) given a logic program could be used to replace the steps 1–4 in Algorithm 1. The present work does not intend to provide a method for computing such a justification.

6 Conclusions and Future Work

In this paper, we investigate MRLP problems between logic programs, represented by a tuple (π_a, π_h, ω), that focus on identifying a solution (ϵ^+, ϵ^-) where $\epsilon^+ \subseteq \pi_a$ and $\epsilon^- \subseteq \pi_h$ such that $\hat{\pi}_h = \pi_h \setminus \epsilon^- \cup \epsilon^+$ satisfying $\hat{\pi}_h \mid\!\sim \omega^+$ and $\hat{\pi}_h \not\mid\!\sim \omega^-$, i.e., $\hat{\pi}_h \mid\!\sim q$ for every $q \in \omega^+$ and $\hat{\pi}_h \not\mid\!\sim r$ for every $r \in \omega^-$.

We show that if $\pi_a \mid\!\sim \omega^+$ and $\pi_a \not\mid\!\sim \omega^-$ and there exists a justification $\epsilon^+ \subset \pi_a$ for ω^+ then there exists a non-trivial solution (ϵ^+, ϵ^-) for the problem. We discuss different types of solutions of a MRLP problem and algorithms for computing a solution. We also present the notion of a cost-based and assessertive characterization of solutions.

In this paper, we focus on the development of the theoretical foundation of the MRLP problems. One of our immediate future work is to develop a system for computing solutions of MRLP problems. The next goal is to experimentally comparing this system with the system described in [8].

For future work, we note that our work assumes that the robot, who needs to computes solutions, has the knowledge of both programs π_a and π_h, which is the assumption in early work in explainable planning. In practice, this assumption is likely invalid and the robot might also needs to change its program through communication or dialogue with the human. For example, if the robot explains to the human that its plan for going from location a to location c through location b is feasible and the human informs the robot that the path from location a to location b is currently blocked, then the robot should eliminate the action of going from location a to location b from its action description and replan a new path to get to location c. Therefore, we plan to take such dialogue into account and to formalize the process of reaching a consensus between the robot and the human in the near future.

References

1. Chakraborti, T., Sreedharan, S., Zhang, Y., Kambhampati, S.: Plan explanations as model reconciliation: Moving beyond explanation as soliloquy. In: IJCAI. pp. 156–163 (2017)
2. van Emden, M., Kowalski, R.: The semantics of predicate logic as a programming language. Journal of the ACM **23**(4), 733–742 (1976)
3. Fox, M., Long, D., Magazzeni, D.: Explainable planning. CoRR **abs/1709.10256** (2017), http://arxiv.org/abs/1709.10256
4. Gelfond, M., Lifschitz, V.: Logic programs with classical negation. In: LP. pp. 579–597 (1990)
5. Kambhampati, S.: Synthesizing explainable behavior for human-AI collaboration. In: AAMAS. pp. 1–2 (2019)
6. Lifschitz, V., Pearce, D., Valverde, A.: Strongly equivalent logic programs. ACM Transactions on Computational Logic **2**(4), 526–541 (2001)
7. Marek, V., Truszczyński, M.: Stable models and an alternative logic programming paradigm. In: The Logic Programming Paradigm: a 25-year Perspective. pp. 375–398 (1999)
8. Nguyen, V., Vasileiou, S.L., Son, T.C., Yeoh, W.: Explainable Planning Using Answer Set Programming. In: KRR. pp. 662–666 (2020)
9. Niemelä, I.: Logic programming with stable model semantics as a constraint programming paradigm. Annals of Mathematics and Artificial Intelligence **25**(3,4), 241–273 (1999)
10. Sakama, C., Inoue, K.: Updating extended logic programs through abduction. In: LPNMR. pp. 147–161 (1999)
11. Slota, M., Leite, J.: Exception-based knowledge updates. CoRR **abs/1706.00585** (2017), http://arxiv.org/abs/1706.00585
12. Sreedharan, S., Chakraborti, T., Kambhampati, S.: Handling model uncertainty and multiplicity in explanations via model reconciliation. In: ICAPS. pp. 518–526 (2018)
13. Vasileiou, S.L., Previti, A., Yeoh, W.: On exploiting hitting sets for model reconciliation. In: AAAI (2021)
14. Zhang, Y.: Logic program-based updates. ACM Transactions on Computational Logic **7**(3), 421–472 (2006)
15. Zhang, Y., Foo, N.Y.: Updating logic programs. In: ECAI. pp. 403–407 (1998)