

Revision Programming with Preferences

by

Inna Pivkina, Enrico Pontelli, Tran Cao Son

Outline

1. Basic concepts of revision programming (RP).
2. Two approaches to express preferences
 - (a) Control program
 - Example
 - Syntax and Semantics
 - Properties
 - (b) Soft revision rules with weights
 - Examples
 - Definitions
 - Implementation

RP:

- Formalism for describing and enforcing constraints on databases
- Database - a collection of atomic facts from some universe.
- Revision rules
 - specify constraints on a database,
 - specify a preferred way to satisfy constraints.
- Arbitrary initial database.
- Justified revisions
 - satisfy all constraints,
 - all changes are justified by revision rules.

Basic concepts

- Universe U . Elements of U - *atoms*. Subsets of U - *databases*.
- Revision literals: $in(a)$, $out(a)$ ($a \in U$).
- *Revision rules*:

$$in(a) \leftarrow in(a_1), \dots, in(a_m), out(b_1), \dots, out(b_n), \quad (in\text{-rule})$$

$$out(a) \leftarrow in(a_1), \dots, in(a_m), out(b_1), \dots, out(b_n), \quad (out\text{-rule})$$

where $a, a_i, b_i \in U$ ($1 \leq i \leq n$).

- *Revision program* - collection of revision rules.

Necessary change

- α^D - dual of a literal α . $in(a)^D = out(a)$, $out(a)^D = in(a)$.
- A set of literals is *coherent* if it does not contain a pair of dual literals.
- P – a revision program. The *necessary change* of P , $NC(P)$, is the least model of P , treated as a Horn program built of independent propositional atoms of the form $in(a)$ and $out(b)$.
- Coherent $NC(P)$ specifies a revision.

Example.

$$\begin{array}{l} P : in(Ann) \leftarrow \\ \quad out(Bob) \leftarrow in(Ann) \\ \quad out(Tom) \leftarrow out(Ann) \end{array} \quad NC(P) = \{in(Ann), out(Bob)\}$$

Justified revisions

- Given a database I and a coherent set of literals L , define

$$I \oplus L = (I \cup \{a: in(a) \in L\}) \setminus \{a: out(a) \in L\}.$$

- *Inertia set* for databases I, R :

$$I(I, R) = \{in(a) : a \in I \cap R\} \cup \{out(a) : a \notin I \cup R\}.$$

- *Reduct of P with respect to (I, R)* (denoted $P_{I,R}$) – the revision program obtained from P by eliminating from the body of each rule in P all literals in $I(I, R)$.
- P - a revision program, I and R - databases. R is called a *P -justified revision* of I if $NC(P_{I,R})$ is coherent and $R = I \oplus NC(P_{I,R})$.

Example

$P :$ $in(Ann) \leftarrow out(Bob)$

$in(Bob) \leftarrow out(Ann)$

$in(David) \leftarrow in(Tom)$

$out(Tom) \leftarrow out(David)$

$out(Ann) \leftarrow in(David)$

$out(David) \leftarrow in(Bob)$

$P_{I,R} :$ $in(Ann) \leftarrow out(Bob)$

$in(Bob) \leftarrow$

$in(David) \leftarrow in(Tom)$

$out(Tom) \leftarrow out(David)$

$out(Ann) \leftarrow in(David)$

$out(David) \leftarrow in(Bob)$

Initial database: $I = \{David, Tom\}$.

Revision: $R = \{Bob\}$.

Inertia (no justification is needed): $out(Ann)$.

Necessary change: $in(Bob), out(David), out(Tom)$.

Updating I : $(I \cup \{Bob\}) \setminus \{David, Tom\}$.

Basic properties

1. If a database R is a P -justified revision of I , then R is a model of P .
2. If a database B satisfies a revision program P then B is a unique P -justified revision of itself.
3. If R is a P -justified revision of I , then $R \div I$ is minimal in the family $\{B \div I : B \text{ is a model of } P\}$

Relation to Logic Programming

P -justified revisions of \emptyset coincide with stable models of the logic program with constraints, $lp(P)$, obtained from P by replacing revision rules of the form

$$in(a) \leftarrow in(a_1), \dots, in(a_m), out(b_1), \dots, out(b_n)$$

by

$$a \leftarrow a_1, \dots, a_m, not\ b_1, \dots, not\ b_n$$

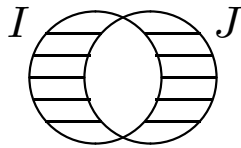
and replacing revision rules of the form

$$out(a) \leftarrow in(a_1), \dots, in(a_m), out(b_1), \dots, out(b_n)$$

by constraints

$$\leftarrow a, a_1, \dots, a_m, not\ b_1, \dots, not\ b_n.$$

Shifting



$$W = I \div J = (I \setminus J) \cup (J \setminus I)$$

W - a set of atoms that change status

Define a W -transformation (*shift*) as follows.

For a literal α ($\alpha = in(a)$ or $\alpha = out(a)$), $T_W(\alpha) = \begin{cases} \alpha^D, & \text{when } a \in W \\ \alpha, & \text{when } a \notin W. \end{cases}$

For a set of literals L , $T_W(L) = \{T_W(\alpha) : \alpha \in L\}$.

For a set of atoms X ,

$$T_W(X) = \{ a : in(a) \in T_W(\{in(b) : b \in X\} \cup \{out(b) : b \notin X\}) \}.$$

For a revision program P , $T_W(P)$ is obtained from P by applying T_W to each literal in P .

Shifting theorem

For any databases I_1 and I_2 , database R is a P -justified revision of I_1 if and only if $T_{I_1 \div I_2}(R)$ is a $T_{I_1 \div I_2}(P)$ -justified revision of I_2 .

Corollary. *For each I and R , R is P -justified revision of I if and only if $T_I(R)$ is $T_I(P)$ -justified revision of \emptyset .*

Computing justified revisions

by means of LP

1. Given P and I , apply T_I to obtain $T_I(P)$ and \emptyset .
2. Convert $T_I(P)$ into the logic program $lp(T_I(P))$.
3. Compute its answer sets.
4. Apply T_I to the answer sets to obtain the P -justified revisions of I .

Robot example

A robot is equipped with sensors which provide observations:

observation(Par, Value, Sensor)

View of the world has exactly one value for each parameter:

world(Par, Value, Sensor)

RP updates the view of the world, consists of rules of types:

in(*observation(Par, Value, Sensor)*) ←

in(*world(Par, Value, Sensor)*) ← **in**(*observation(Par, Value, Sensor)*).

out(*world(Par, Value, Sensor)*) ← **in**(*world(Par, Value1, Sensor1)*).

(where *Sensor* ≠ *Sensor1* and/or *Value* ≠ *Value1*)

Syntax

An *ordered revision program* is a pair (P, \mathcal{L}) where \mathcal{L} is a function which assigns to revision rules in P unique labels. $\mathcal{L}(P)$ - set of labels in P .

$$l : \alpha_0 \leftarrow \alpha_1, \dots, \alpha_n$$

A *preference* on rules in (P, \mathcal{L}) is an expression of the form

$$prefer(l_1, l_2) \leftarrow initially(\alpha_1, \dots, \alpha_k), \alpha_{k+1}, \dots, \alpha_n,$$

where l_i are labels, α_j are revision literals.

A *revision program with preferences* is a triple (P, \mathcal{L}, S) , where (P, \mathcal{L}) is an ordered revision program and S is a set of preferences on rules in (P, \mathcal{L}) .

S - the *control program*.

(P, \mathcal{L}, S) is translated into ordinary RP:

$$U^{\mathcal{L}(P)} = U \cup \{ok(l), defeated(l), prefer(l, l') : l, l' \in \mathcal{L}(P)\}$$

Define $P^{S,I}$ over $U^{\mathcal{L}(P)}$ to be a revision program consisting of rules:

- for each $l \in \mathcal{L}(P)$

$$\begin{aligned} head(l) &\leftarrow body(l), in(ok(l)) \\ in(ok(l)) &\leftarrow out(defeated(l)) \end{aligned}$$

- for each preference

$$prefer(l_1, l_2) \leftarrow initially(\alpha_1, \dots, \alpha_k), \alpha_{k+1}, \dots, \alpha_n,$$

in S such that $\alpha_1 \dots, \alpha_k$ are satisfied by I

$$\begin{aligned} in(prefer(l_1, l_2)) &\leftarrow \alpha_{k+1}, \dots, \alpha_n \\ in(defeated(l_2)) &\leftarrow body(l_1), in(prefer(l_1, l_2)) \end{aligned}$$

(P, \mathcal{L}, S) -justified revisions

A database R is a (P, \mathcal{L}, S) -justified revision of I if there exists $R' \subseteq U^{\mathcal{L}(P)}$ such that R' is a $P^{S,I}$ -justified revision of I , and $R = R' \cap U$.

Properties

- Justified revision semantics for revision programs with preferences extends justified revision semantics for ordinary revision programs
- Shifting property holds
- Not every (P, \mathcal{L}, S) -justified revision is a model of P

When (P, \mathcal{L}, S) -justified revisions are models of P ?

Two rules r, r' of P are *in conflict* if one of the following conditions is satisfied:

1. $(\text{head}(r))^D \in \text{body}(r')$ and $(\text{head}(r'))^D \in \text{body}(r)$; or
2. $\text{body}(r) \cup \text{body}(r')$ is incoherent.

A set of preferences is *conflict-resolving* if it contains only preferences between conflicting rules.

Theorem. *Let (P, \mathcal{L}, S) be a revision program with preferences where S is a set of conflict-resolving preferences and is cycle-free. For every (P, \mathcal{L}, S) -justified revision R of I , R is a model of P .*

Soft revision rules with weights

Revision program is divided into hard and soft rules: $P = HR \cup SR$

All hard rules must be satisfied.

Only a subset of soft rules may be satisfied.

The subset of soft rules that is satisfied is optimal with respect to some criteria.

Maximal number of rules

Definition 1 *R is a (HR, SR) -preferred justified revision of I if R is a $(HR \cup S)$ - justified revision of I for some $S \subseteq SR$, and for all S' if $S \subset S' \subseteq SR$, then there are no $(HR \cup S')$ -justified revisions of I.*

Implementation

For each I , translate $P = HR \cup SR$ into an `smodels` program $lp(T_I(HR)) \cup lp'(T_I(SR))$.

lp' translates a rule

$$in(a) \leftarrow in(p_1), \dots, in(p_m), out(s_1), \dots, out(s_n)$$

into the rules

$$\begin{aligned} \{rule_i\} & : - \quad p_1, \dots, p_m, not\ s_1, \dots, not\ s_n. \\ a & : - \quad rule_i \end{aligned}$$

lp' translates a rule

$$out(a) \leftarrow in(p_1), \dots, in(p_m), out(s_1), \dots, out(s_n)$$

into the rules

$$\begin{aligned} \{rule_i\} & : - \quad p_1, \dots, p_m, not\ s_1, \dots, not\ s_n. \\ & : - \quad rule_i, a. \end{aligned}$$

Implementation, cont'd

smodels statement

$$\text{maximize}\{rule_1, \dots, rule_k\}.$$

(k is the number of rules in SR)

allows to compute one (not all) (HR, SR) -preferred justified revision, which has max size.

Weighted rules

Each $r \in SR$ is assigned a weight , $w(r)$ (its importance).

Definition 2 R is called a rule-weighted (HR, SR) -justified revision of I if the following two conditions are satisfied:

1. there exists a set of rules $S \subseteq SR$ such that R is a $(HR \cup S)$ -justified revision of I , and
2. for any set of rules $S' \subseteq SR$, if R' is a $(HR \cup S')$ -justified revision of I , then the sum of weights of rules in S' is less than or equal than the sum of weights of rules in S .

Implementation

Same translation of soft rules into `smodels` program, but different `maximize` statement:

```
maximize[rule1 = w(1), rule2 = w(2), ..., rulek = w(k)].
```

Weighted atoms.

Each $a \in U$ is assigned a weight $w(a)$
(the more the weight the less we want to change its status)

Definition 3 R is called an atom-weighted (HR, SR) -justified revision of I if the following two conditions are satisfied:

1. there exists a set of rules $S \subseteq SR$ such that R is a $(HR \cup S)$ -justified revision of I , and
2. for any set of rules $S' \subseteq SR$, if Q is a $(HR \cup S')$ -justified revision of I , then the sum of weights of atoms in $I \div Q$ is greater than or equal to the sum of weights of atoms in $I \div R$.

Implementation

Same translation of soft rules into `smodels` program, but different `maximize` statement:

$$\text{minimize}[a_1 = w(a_1), a_2 = w(a_2), \dots, a_n = w(a_n)]$$

where a_1, \dots, a_n are all the atoms in U .

Minimal size difference

Definition 4 *R is called a minimal size difference P-justified revision of I if the following two conditions are satisfied:*

- 1. R is a P-justified revision of I, and*
- 2. for any P-justified revision R', the number of atoms in $R \div I$ is less than or equal to the number of atoms in $R' \div I$.*

Implementation

Use minimize statement:

$$\text{minimize}\{a_1, a_2, \dots, a_n\}$$

where a_1, \dots, a_n are all the atoms in U .