Revision programming

by

Inna Pivkina
Outline

1. Basic concepts of revision programming (RP).
2. Connection between logic programming and RP.
3. Extensions: disjunctive RP.
4. Dealing with uncertainty: Annotated RP.
RP:

- Formalism for describing and enforcing constraints on databases introduced by Marek and Truszczynski.
- 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.
Example

Candidates: Ann, Bob, David, Tom.

Constraints:  
(1) either Ann or Bob;  
(2) Tom can be only with David;  
(3) if David then no Ann;  
(4) if Bob then no David.

Initial proposal: David, Tom.

Goal: form a committee that

- satisfies all constraints
- differs minimally from the initial proposal
- all changes to the initial proposal are justified
Example, cont’d

\[
P : \quad \text{in}(Ann) \leftarrow \text{out}(Bob) \quad P_{I,R} : \quad \text{in}(Ann) \leftarrow \text{out}(Bob)
\]
\[
\neg \text{in}(Bob) \leftarrow \text{out}(Ann)
\quad \text{in}(Bob) \leftarrow 
\]
\[
\text{in}(David) \leftarrow \text{in}(Tom)
\quad \text{in}(David) \leftarrow \text{in}(Tom)
\]
\[
\text{out}(Tom) \leftarrow \text{out}(David)
\quad \text{out}(Tom) \leftarrow \text{out}(David)
\]
\[
\text{out}(Ann) \leftarrow \text{in}(David)
\quad \text{out}(Ann) \leftarrow \text{in}(David)
\]
\[
\text{out}(David) \leftarrow \text{in}(Bob)
\quad \text{out}(David) \leftarrow \text{in}(Bob)
\]

Initial database: \( I = \{David, Tom\} \).
Revision: \( R = \{Bob\} \).

Inertia (no justification is needed): \( \text{out}(Ann) \).

Necessary change: \( \text{in}(Bob), \text{out}(David), \text{out}(Tom) \).

Updating \( I \): \( (I \cup \{Bob\}) \setminus \{David, Tom\} \).
Graph 3-colorability

Problem: Given a coloring (which may be partial or inconsistent) find a coloring that differs minimally from the initial coloring and satisfies the following conditions:

- every vertex has exactly one color;
- any two vertices that are connected by an edge have different colors.

Initial database:

Revision:
Basic concepts

- Revision literals: $\text{in}(a)$, $\text{out}(a)$ ($a \in U$).
- Revision rules:

\[
\text{in}(a) \leftarrow \text{in}(a_1), \ldots, \text{in}(a_m), \text{out}(b_1), \ldots, \text{out}(b_n), \quad (\text{in-rule})
\]

\[
\text{out}(a) \leftarrow \text{in}(a_1), \ldots, \text{in}(a_m), \text{out}(b_1), \ldots, \text{out}(b_n), \quad (\text{out-rule})
\]

where $a, a_i, b_i \in U$ ($1 \leq i \leq n$).
- Revision program - collection of revision rules.
Justified revisions: necessary change

• Example.

\[ P: \text{in}(Ann) \leftarrow \text{out}(Bob) \leftarrow \text{in}(Ann) \leftarrow \text{out}(Ann) \]

\[ NC(P) = \{\text{in}(Ann), \text{out}(Bob)\} \]

• The necessary change of \( P, \ NC(P) \), is the least model of \( P \) treated as a Horn program.

• \( \alpha^D \) - dual of a literal \( \alpha \). \( \text{in}(a)^D = \text{out}(a), \ \text{out}(a)^D = \text{in}(a) \).

• A set of literals \( L \) is coherent if it does not contain a pair of dual literals.

• Coherent \( L \) specifies a revision:

\[ I \oplus L = (I \cup \{a: \text{in}(a) \in L\}) \setminus \{a: \text{out}(a) \in L\}. \]
Justified revisions

• To compute:
  – propose a candidate
  – check whether it is a justified revision or not
  – if not, consider another candidate

• Inertia set for databases $I, R$:
  \[ I(I, R) = \{ \text{in}(a) : a \in I \cap R \} \cup \{ \text{out}(a) : a \notin I \cup R \}. \]

• Literals from inertia need no justification.

• 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 $R = I \oplus NC(P_{I,R})$. 
Models

- $B \subseteq U$ is a model of (or satisfies) a literal $\text{in}(a)$ if $a \in B$.
  $B \subseteq U$ is a model of (or satisfies) a literal $\text{out}(a)$ if $a \notin B$.

- $B \subseteq U$ is a model of (or satisfies) a rule of the form $\alpha \leftarrow \alpha_1, \ldots, \alpha_n$ if either $B$ is not a model of at least one literal $\alpha_i$, or $B$ is a model of $\alpha$.

- $B \subseteq U$ is a model of (or satisfies) a revision program $P$ if $B$ is a model of every rule in $P$. 
Basic properties

- $P$-justified revisions are models of $P$;
- justified revisions of a database differ from the database by as little as possible;
- if the current database satisfies the revision program, then no nontrivial change is justified;
- additional evidence does not destroy justified revisions;
- the problem of existence of $P$-justified revisions is NP-complete.
Stable models of logic programs

A logic program is a set of clauses of the form

\[ p \leftarrow q_1, \ldots, q_m, \text{not } s_1, \ldots, \text{not } s_n \]

\( P \) - a logic program. Reduct of \( P \) relative to \( M \), \( P^M \), is obtained from \( P \) by

- removing all clauses which contain \text{“not } q\text{”} such that \( q \) is true in \( M \),
- deleting all negative premises \text{“not } q\text{”} from the remaining clauses.

\( P^M \) has a unique least model \( \text{Least}(P^M) \).

\( M \) is a \textit{stable model} of \( P \) if \( M = \text{Least}(P^M) \).
Relation to logic programming (LP)

Logic program clauses correspond to revision in-rules.

For a logic program clause \( c = p \leftarrow q_1, \ldots, q_m, \text{not } s_1, \ldots, \text{not } s_n, \)

\[
\text{in-rule } \quad r = \textbf{in}(p) \leftarrow \textbf{in}(q_1), \ldots, \textbf{in}(q_m), \textbf{out}(s_1), \ldots, \textbf{out}(s_n)
\]

We define \( rp(c) = r, \quad lp(r) = c. \)

For a logic program \( P \) define \( rp(P) = \{ rp(c) : c \in P \}. \)

For a revision program \( P \) define \( lp(P) = \{ lp(r) : r \in P \}. \)

**Theorem.** A set of atoms \( X \) is a stable model of a logic program \( P \) if and only if \( X \) is an \( rp(P) \)-justified revision of \( \emptyset \).
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 $\alpha$ ($\alpha = \text{in}(a)$ or $\alpha = \text{out}(a)$), $T_W(\alpha) = \begin{cases} \alpha^D, & \text{when } a \in W \\ \alpha, & \text{when } a \notin W. \end{cases}$

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

For a set of atoms $X$, $T_W(X) = \{ a : \text{in}(a) \in T_W(\{\text{in}(b) : b \in X\} \cup \{\text{out}(b) \mid \text{col}nb \notin X\}) \}$.

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

Let $I_1$, $I_2$ be databases. Let $W = I_1 \div I_2$ be their symmetric difference. Then $T_W(I_1) = I_2$.

**Theorem.** Let $P$ be a revision program. Let $I_1$ and $I_2$ be databases. Let $W = I_1 \div I_2$. Then, a database $R$ is a $P$-justified revision of $I_1$ if and only if $T_W(R)$ is a $T_W(P)$-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 $\emptyset$. 
Example (shifting)

\[ P : \quad \text{out}(Ann) \leftarrow \text{in}(Bob) \]
\[ \text{in}(Tom) \leftarrow \text{out}(Bob) \]
\[ \text{in}(David) \leftarrow \text{in}(Tom) \]

Let \( I = \{Ann, Bob, David\} \).
The only \( P \)-justified revision of \( I \) is \( R = \{Bob, David\} \).

\[ T_I(P) : \quad \text{in}(Ann) \leftarrow \text{out}(Bob) \]
\[ \text{in}(Tom) \leftarrow \text{in}(Bob) \]
\[ \text{out}(David) \leftarrow \text{in}(Tom) \]

\( T_I(I) = \emptyset \). The only justified revision of \( \emptyset \) is \( \{Ann\} \).

Observe that \( \{Ann\} = T_I(\{Bob, David\}) \).
Revision programming =

logic programming + constraints

\( \text{InRules}(P) \) - a set of all in-rules of \( P \)

**Theorem.** Let \( P \) be a revision program. Then, \( R \) is a \( P \)-justified revision of \( \emptyset \) if and only if \( R \) is a \( \text{InRules}(P) \)-justified revision of \( \emptyset \) and \( R \) is a model of \( P \setminus \text{InRules}(P) \).

**Corollary.** Let \( P \) be a revision program. Then, \( R \) is a \( P \)-justified revision of \( \emptyset \) if and only if \( R \) is a stable model of \( \text{lp}(\text{InRules}(P)) \) and \( R \) is a model of \( P \setminus \text{InRules}(P) \).
Computing justified revisions

by means of LP

\[ P \quad I \quad \rightarrow \quad P_1 = T_I(P) \quad \emptyset \quad \rightarrow \quad P' = \text{InRules}(P_1) \quad P'' = P_1 \setminus P' \quad \rightarrow \]

\[ \quad \rightarrow \quad \text{lp}(P') \quad \text{constraints} \quad (P'') \quad \rightarrow \quad X \quad \text{stable model of lp}(P') \quad X \quad \text{satisfies} \quad P'' \quad \rightarrow \]

\[ \quad \rightarrow \quad X \quad - \quad P_1 \text{-just. rev. of} \quad \emptyset \quad \rightarrow \quad T_I(X) \quad - \quad P \text{-just. rev. of} \quad \]

\[ I \]
Graph 3-colorability

**Problem:** Find a coloring that differs minimally from the initial coloring and satisfies the following condition. Any two vertices that are connected by an edge should have different colors.

**Graph description:** \(vtx(a), \ldots, edg(a,b), \ldots, color(red), \ldots\)

**Revision program:** (constraints)

\[
\begin{align*}
out(clr(Y,C)) & \leftarrow color(C), \ edg(X,Y), \ in(clr(X,C)). \\
out(clr(X,C)) & \leftarrow color(C), \ edg(X,Y), \ in(clr(Y,C)). \\
in(clr(X,red)) & \leftarrow vtx(X), \ out(clr(X,green)), \ out(clr(X,blue)). \\
in(clr(X,green)) & \leftarrow vtx(X), \ out(clr(X,red)), \ out(clr(X,blue)). \\
in(clr(X,blue)) & \leftarrow vtx(X), \ out(clr(X,red)), \ out(clr(X,green)).
\end{align*}
\]

**Initial database:** \(in(clr(a,red)), \ in(clr(b,red)), \ in(clr(c,green)), \ldots\)
% conv coloring | lparse -d none | smodels | convback
smodels version 2.8. Reading...done
Answer: 1
Revision: clr(b,green) clr(a,red) clr(c,green) clr(d,blue) clr(e,green)
True
%

Example

Initial database:

Revision:
Extensions
Disjunctive revision programs (DRP)

Rules are not definite:

\[
\begin{align*}
in(Ann) & \mid out(Tom) \mid in(Bob) \leftarrow in(David) \\
in(Tom) & \mid out(David) \leftarrow in(Ann), \ out(Bob) \\
out(Ann) & \leftarrow in(David) \\
out(David) & \leftarrow in(Bob)
\end{align*}
\]

Initial database: \( I = \{David, Tom\} \). Revision: \( R - ? \)

Same key concepts: reduct, necessary change.
Justified revisions for DRP

$P$ - a disjunctive revision program, $I$ and $R$ - databases.

A reduct of $P$ with respect to $(I, R)$ (denoted by $P^{I,R}$) is defined in four steps:

1. Eliminate from $P$ every rule whose body is not satisfied by $R$.
2. From the body of each remaining rule eliminate each literal that is satisfied by $I$.
3. Remove all rules $r$, such that $\text{head}(r) \cap I(I, R) \neq \emptyset$.
4. Remove from the heads of the rules all literals that contradict $R$.

A database $R$ is a $P$-justified revision of a database $I$ if for some necessary change $L$ of $P^{I,R}$, $L$ is coherent and $R = I \oplus L$. 
Example (DRP)

\[ P:\]

\[
in(Ann) \mid in(Bob) \quad \leftarrow
\]
\[
out(Tom) \mid in(David) \quad \leftarrow
\]
\[
out(Ann) \quad \leftarrow \quad in(David)
\]
\[
out(David) \quad \leftarrow \quad in(Bob)
\]

\[ I = \{Ann, Tom\}, \quad R = \{Ann\}.
\]

Inertia \( I(I, R) = \{in(Ann), out(Bob), out(David)\} \).

The reduct \( P^{I,R} = \{out(Tom) \leftarrow\} \).

\( NC(P^{I,R}) = \{out(Tom)\} \) - coherent. \( R = I \oplus NC(P^{I,R}) \).

Therefore, \( R \) is a \( P \)-justified revision of \( I \).
DRP: properties

- The semantics of disjunctive revision programming reduces to the semantics of justified revisions on DRPs consisting of rules with a single literal in the head.
- The shifting theorem generalizes to the case of DRPs.
- The semantics of disjunctive revision programming over the empty initial database reduces to the Lifschitz and Woo semantics for General Disjunctive Logic Programs (GDLP).
Dealing with uncertainty:

Annotated revision programs
Example

A group of experts is about to discuss and then vote whether to accept or reject a proposal.

Each person has an opinion on the proposal that may be changed during the discussion as follows:
- any person can convince an optimist to vote for the proposal,
- any person can convince a pessimist to vote against the proposal.

The group consists of two optimists (Ann and Bob) and one pessimist (Pete). Given everybody’s opinion on the subject before the discussion, what are possible outcomes of the vote?
Example, cont’d

\[(\text{in}(@\text{accept}):\{Bob\}) \leftrightarrow (\text{in}(@\text{accept}):\{Ann\})\]
\[(\text{in}(@\text{accept}):\{Bob\}) \leftrightarrow (\text{in}(@\text{accept}):\{Pete\})\]
\[(\text{in}(@\text{accept}):\{Ann\}) \leftrightarrow (\text{in}(@\text{accept}):\{Bob\})\]
\[(\text{in}(@\text{accept}):\{Ann\}) \leftrightarrow (\text{in}(@\text{accept}):\{Pete\})\]
\[(\text{out}(@\text{accept}):\{Pete\}) \leftrightarrow (\text{out}(@\text{accept}):\{Ann\})\]
\[(\text{out}(@\text{accept}):\{Pete\}) \leftrightarrow (\text{out}(@\text{accept}):\{Bob\})\]

Initially: \[B_I(@\text{accept}) = \langle \{Pete\}, \{Bob\} \rangle.\]
(Pete is for. Bob is against. Ann is indifferent.)

Revisions:

1. \[B_R(@\text{accept}) = \langle \{Ann, Bob, Pete\}, \{\} \rangle\] (All are for.)
2. \[B'_R(@\text{accept}) = \langle \{\}, \{Bob, Pete\} \rangle\]
(Bob and Pete are against. Ann remains indifferent.)
Example

Two sources of light: $a$ and $b$.
Dust in the air, light pollution.

$$(\text{in}(a):1) \leftarrow (\text{in}(a):0.8), (\text{out}(b):0.6)$$
$$(\text{out}(b):1) \leftarrow (\text{in}(a):0.8), (\text{out}(b):0.6)$$
$$(\text{in}(b):1) \leftarrow (\text{in}(b):0.8), (\text{out}(a):0.6)$$
$$(\text{out}(a):1) \leftarrow (\text{in}(b):0.8), (\text{out}(a):0.6)$$

Observed brightness: $B_I(a) = <0.3, 0.7>$ and $B_I(b) = <0.9, 0.1>$.
Revision (actual brightness): $B_R(a) = <0, 1>$, and $B_R(b) = <1, 0>$. 
Preliminaries

- $\mathcal{T}$ - a complete infinitely distributive lattice with a de Morgan complement (denoted by $\neg$) It satisfies de Morgan laws
  \[
  a \lor b = \neg (\neg a \land \neg b), \quad a \land b = \neg (\neg a \lor \neg b)
  \]
- $(\text{in}(b):\alpha), \ (\text{out}(b):\alpha)$ - annotated revision atoms $(\alpha \in \mathcal{T}, \ b \in U)$.
- Annotated revision rules:
  \[
  p \leftarrow q_1, \ldots, q_n,
  \]
  where $p, q_i \ (1 \leq i \leq n)$ are annotated revision atoms.
- Annotated revision program is a set of annotated revision rules.
**\( \mathcal{T} \)-valuation**

\( \mathcal{T} \)-valuation is a mapping \( v \) from revision atoms to \( \mathcal{T} \)

\[ v \text{ satisfies } (\text{in}(b) : \alpha) \text{ if } v(\text{in}(b)) \geq \alpha \]

\[ v \text{ satisfies } (\text{out}(b) : \alpha) \text{ if } v(\text{out}(b)) \geq \alpha \]

\( t_{\mathcal{P}}(v) \) - the set of all annotated revision atoms that occur as the head of a rule in \( \mathcal{P} \) whose body is satisfied by \( v \)

Operator on \( \mathcal{T} \)-valuations:

\[ T_{\mathcal{P}}(v)(l) = \bigvee \{ \alpha \mid (l:\alpha) \in t_{\mathcal{P}}(v) \} \]
\[ \mathcal{T}^2 \text{- valuation} \]

- \( \mathcal{T}^2 \) - complete, infinitely distributive lattice:
  
  - Domain: \( \mathcal{T} \times \mathcal{T} \)
  
  - \( \langle \alpha_1, \beta_1 \rangle \leq_k \langle \alpha_2, \beta_2 \rangle \) if \( \alpha_1 \leq \alpha_2 \) and \( \beta_1 \leq \beta_2 \)
  
  - \( \otimes, \oplus \) - meet and join under \( \leq_k \)
  
  - Conflation: \( -\langle \alpha, \beta \rangle = \langle \bar{\beta}, \bar{\alpha} \rangle \)

- An element \( A \in \mathcal{T}^2 \) is consistent if \( A \leq_k -A \).

**Example:** \( \mathcal{T} = \{\emptyset, \{p\}, \{q\}, \{p, q\}\} \)

\[
\begin{align*}
\langle \{p, q\}, \{p\} \rangle & \text{ - inconsistent} & \quad -\langle \{p, q\}, \{p\} \rangle & = \langle \{q\}, \{\} \rangle \\
\langle \{\}, \{q\} \rangle & \text{ - consistent} & \quad -\langle \{\}, \{q\} \rangle & = \langle \{p\}, \{p, q\} \rangle
\end{align*}
\]
\( \mathcal{T}^2 \)-valuation, cont’d

- \( \mathcal{T}^2 \)-valuations are used to represent databases

- Let \( v \) be a \( \mathcal{T} \)-valuation. Then, \( \mathcal{T}^2 \)-valuation \( \theta(v) \) is defined as
  \[
  \theta(v)(b) = \langle \alpha, \beta \rangle, \quad \text{where } v(in(b)) = \alpha \quad \text{and } v(out(b)) = \beta
  \]

- Operator on \( \mathcal{T}^2 \)-valuations:
  \[
  T^b_P = \theta \circ T_P \circ \theta^{-1}
  \]
Models and c-models (example)

Let $\mathcal{T} = \{\emptyset, \{p\}, \{q\}, \{p, q\}\}$.

$P = \{ (\text{in}(a):\{q\}) \leftarrow (\text{out}(a):\{p\}) \}$.

$B_1(a) = \langle \{p, q\}, \{p\} \rangle$

$B_2(a) = \langle \{p, q\}, \{p, q\} \rangle$

*Explicit* evidence provided by $P$ for $\text{in}(a)$ is $\{q\}$, for $\text{out}(a)$ is $\{\}$.

*Implicit* bound provided by $P$ for $\text{out}(a)$ is $\{q\} = \{p\}$,

for $\text{in}(a)$ is $\{\} = \{p, q\}$.

Both $B_1$ and $B_2$ are models of $P$. However, only $B_1$ agrees with the implicit bounds imposed by $P$. 
Let $P$ be an annotated revision program.

**Theorem.** A $\mathcal{T}^2$-valuation $B$ is a model of $P$ if and only if $B \geq_k T_P^b(B)$.

**Definition 1** Let $B$ be a $\mathcal{T}^2$-valuation. We say $B$ is a c-model of $P$ if

$$T_P^b(B) \leq_k B \leq_k T_P^b(B) \oplus (-T_P^b(B)).$$

**Theorem.** A consistent $\mathcal{T}^2$-valuation $B$ is a c-model of $P$ if and only if $B$ is a model of $P$. 
Justified revisions - goals

- Extend the semantics of justified revisions in the original case so that key properties are preserved.

- Satisfy the *invariance under join* principle: replacing rule

\[ r = \ldots \leftarrow \ldots, (l; \beta_1), \ldots, (l; \beta_2), \ldots \]

by rule

\[ r' = \ldots \leftarrow \ldots, (l; \beta_1 \lor \beta_2), \ldots \]

in annotated revision program should have no effect on justified revisions.
Justified revision - definitions

$B_I$ - initial knowledge, $P$ - annotated revision program.

For $\alpha, \beta \in \mathcal{T}$, $pcomp(\alpha, \beta)$ denotes the least $\gamma$ such that $\alpha \lor \gamma \geq \beta$.

Necessary change of $P$ (denoted as $\text{NC}(P)$) is the least fixpoint of the operator $T^b_P$.

Result of applying a change $C$ to a database $B_I$ is defined as

$$B_R = (B_I \otimes -C') \oplus C.$$
Definition of justified revision

**Definition 2** The reduct $P_{BR}|B_I$ is obtained from $P$ by

1. removing every rule whose body contains an annotated atom that is not satisfied in $B_R$,

2. replacing each annotated atom $(l;\beta)$ from the body of each remaining rule by the annotated atom $(l;\gamma)$, where $\gamma = pcomp((\theta^{-1}(B_I))(l), \beta)$.

**Definition 3** $B_R$ is a $P$-justified revision of $B_I$ if $B_R = (B_I \otimes -C) \oplus C$, where $C = NC(P_{BR}|B_I)$ is the necessary change for $P_{BR}|B_I$. 

38
Properties of justified revisions

**Theorem.** Let $P$ be an annotated revision program and let $B_I$ and $B_R$ be $T^2$-valuations. If $B_R$ is a $P$-justified revision of $B_I$ then $B_R$ is a c-model of $P$ (and, hence, also a model of $P$).

**Theorem.** Let a $T^2$-valuation $B_I$ be a model of an annotated revision program $P$. Then, $B_I$ is a $P$-justified revision of itself if and only if $B_I$ is a c-model of $P$.  

More properties of justified revisions

**Theorem.** Let $B_I$ be a model of an annotated revision program $P$. Let $B_R$ be a $P$-justified revision of $B_I$. Then, $B_R \leq_k B_I$.

**Theorem.** Let $B_I$ be a consistent model of an annotated revision program $P$. Then, $B_I$ is the only $P$-justified revision of itself.
A database scenario

- RP models integrity constraints
- Two tasks: how to fix, how to reason (without fixing)

Database $D$ satisfies constraints ($RP$) → Database $D'$ no longer satisfies $RP$ → Revision of $D'$ satisfies $RP$

Use Justified Revisions to answer queries ↔ Holds in all revisions? ↔ Holds in no revisions?
Current and future work

- Well-founded semantics.
- Database connection (database repairs).
- Iterated revisions.
- ‘Industrial grade’ implementation?