Belief revision-based case-based reasoning

Julien Cojan[1] and Jean Lieber[234]

[1] INRIA Sophia-Antipolis, Edelweiss Project, julien.cojan@inria.fr
[2] Université de Lorraine, LORIA, UMR 7503 — Vandœuvre-lès-Nancy, F-54506, France, jean.lieber@loria.fr
[3] CNRS, LORIA, UMR 7503 — Vandœuvre-lès-Nancy, F-54506, France
[4] Inria — Villers-lès-Nancy, F-54602, France

Remark on the difference between the paper and the talk

The paper gathers several results on revision-based CBR and is rather technical

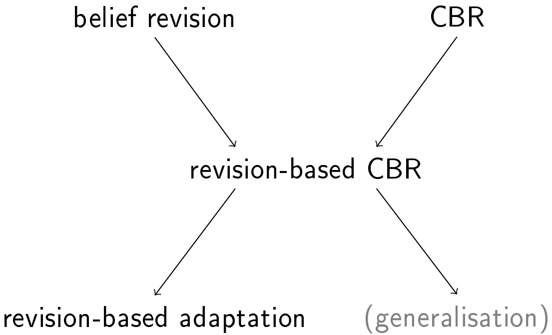
Remark on the difference between the paper and the talk

The paper gathers several results on revision-based CBR and is rather technical

The talk will make some simplifying assumptions and will try to be more intuitive

Outline of the talk

preliminaries



Preliminaries

Formalism

- Generalisation from propositional logic with n variables:
  - $\mathcal{U} = \{\texttt{false}, \texttt{true}\}^n$: set of the interpretations
  - A formula φ represents a subset $\texttt{Mod}(\varphi)$ of $\mathcal{U}$

Formalism

- Generalisation from propositional logic with n variables:
  - $\mathcal{U} = \{\mathtt{false}, \mathtt{true}\}^n$: set of the interpretations
  - A formula φ represents a subset $\mathtt{Mod}(\varphi)$ of $\mathcal{U}$
- Generalisation:
  - $\mathcal{U}$: a given set ("the universe of instances/interpretations")
  - $\mathcal{L}$: the representation language
  - Semantics: $\varphi \in \mathcal{L} \mapsto \mathtt{Mod}(\varphi) \in 2^{\mathcal{U}}$
  - With $A = \mathtt{Mod}(\varphi)$ and $B = \mathtt{Mod}(\chi)$
    - $\varphi \models \chi$ if $A \subseteq B$
    - φ is satisfiable if $A \neq \emptyset$

Distance

- $d : \mathcal{U} \times \mathcal{U} \to [0; +\infty]$ is a distance if

$$d(x, y) = 0 \qquad \text{iff} \qquad x = y$$

(no other condition required in this talk)

Belief Revision

Belief revision (introduction)

- Belief base = knowledge base (no difference in this talk)

Belief revision (introduction)

- Belief base = knowledge base (no difference in this talk)
- Given
  - ψ a revisable consistent knowledge base
  - μ a non revisable consistent knowledge base

Belief revision (introduction)

- Belief base = knowledge base (no difference in this talk)
- Given
  - ψ a revisable consistent knowledge base
  - μ a non revisable consistent knowledge base
- A revision $\boxed{\psi \dot{+} \mu}$ of ψ by μ is a knowledge base obtained by *minimal* modification of ψ so that it is consistent with μ:
  - $\psi \mapsto \psi'$
  - $\psi \dot{+} \mu = \psi' \cap \mu$

Belief revision (introduction)

- Belief base = knowledge base (no difference in this talk)
- Given
  - ψ a revisable consistent knowledge base
  - μ a non revisable consistent knowledge base
- A revision $\boxed{\psi \dot{+} \mu}$ of ψ by μ is a knowledge base obtained by *minimal* modification of ψ so that it is consistent with μ:
  - $\psi \mapsto \psi'$
  - $\psi \dot{+} \mu = \psi' \cap \mu$
- Different ways to model *minimal modification* hence different revision operators

Belief revision (introduction)

- Belief base = knowledge base (no difference in this talk)
- Given
  - ψ a revisable consistent knowledge base
  - μ a non revisable consistent knowledge base
- A revision $\boxed{\psi \dot{+} \mu}$ of ψ by μ is a knowledge base obtained by *minimal* modification of ψ so that it is consistent with μ:
  - $\psi \mapsto \psi'$
  - $\psi \dot{+} \mu = \psi' \cap \mu$
- Different ways to model *minimal modification* hence different revision operators
- In the following, $\dot{+}$ is defined on $\mathcal{L} \times \mathcal{L}$ or on $2^{\mathcal{U}} \times 2^{\mathcal{U}}$:

$$\mathtt{Mod}(\psi \dot{+} \mu) = \mathtt{Mod}(\psi) \dot{+} \mathtt{Mod}(\mu)$$

Belief revision (postulates)

- AGM postulates (1985)

Belief revision (postulates)

- AGM postulates (1985)
- Katsuno & Mendelzon (1992):
  reformulation in propositional logic

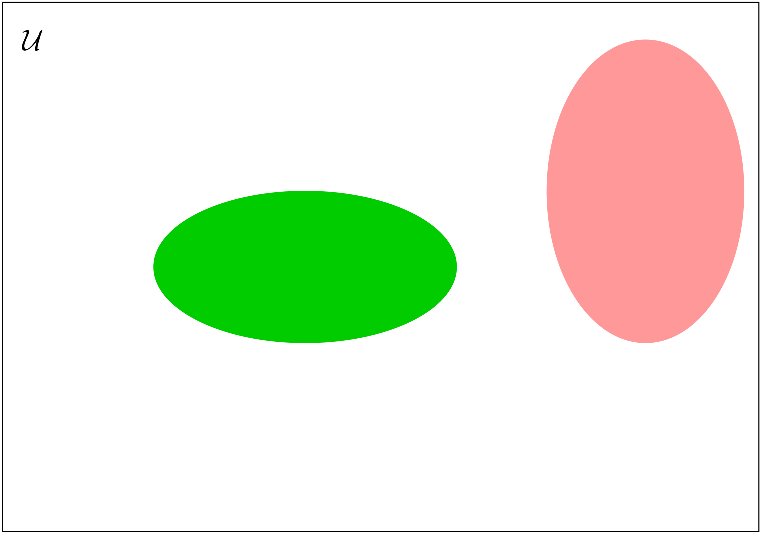
Belief revision (postulates)

- AGM postulates (1985)
- Katsuno & Mendelzon (1992):
  reformulation in propositional logic
- Easy to generalise in our formalism (see further)

Belief revision (distance-based revision operators)

With d, a distance on $\mathcal{U}$, $A \subseteq \mathcal{U}$, $B \subseteq \mathcal{U}$:

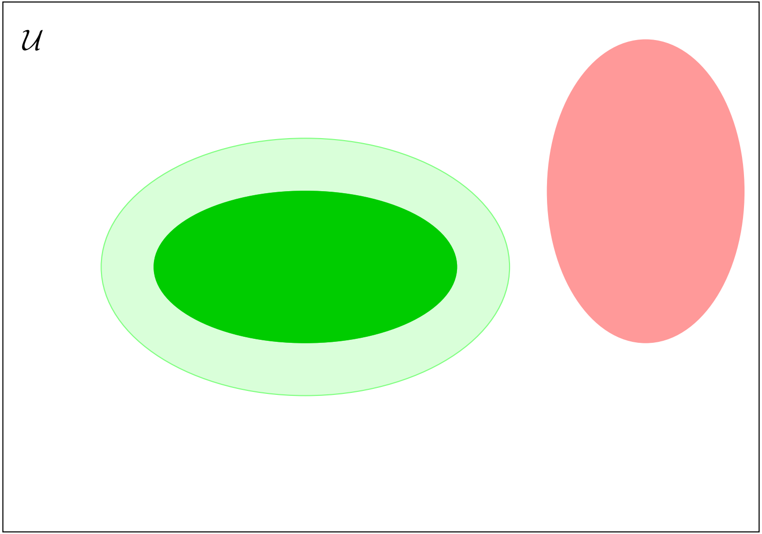
$$A \dotplus^d B = \{\mathtt{b} \in B \mid d(A, \mathtt{b}) = d(A, B)\}$$



Belief revision (distance-based revision operators)

With d, a distance on $\mathcal{U}$, $A \subseteq \mathcal{U}$, $B \subseteq \mathcal{U}$:

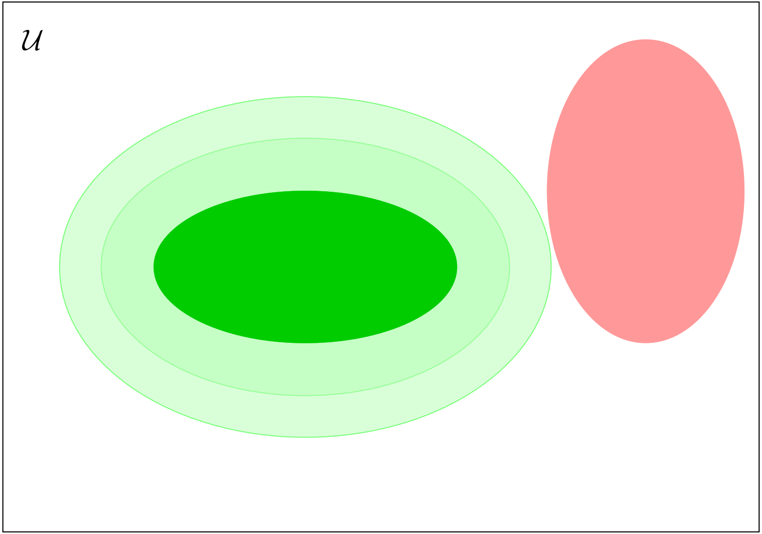
$$A \dot{+}^d B = \{\mathtt{b} \in B \mid d(A, \mathtt{b}) = d(A, B)\}$$



Belief revision (distance-based revision operators)

With d, a distance on $\mathcal{U}$, $A \subseteq \mathcal{U}$, $B \subseteq \mathcal{U}$:

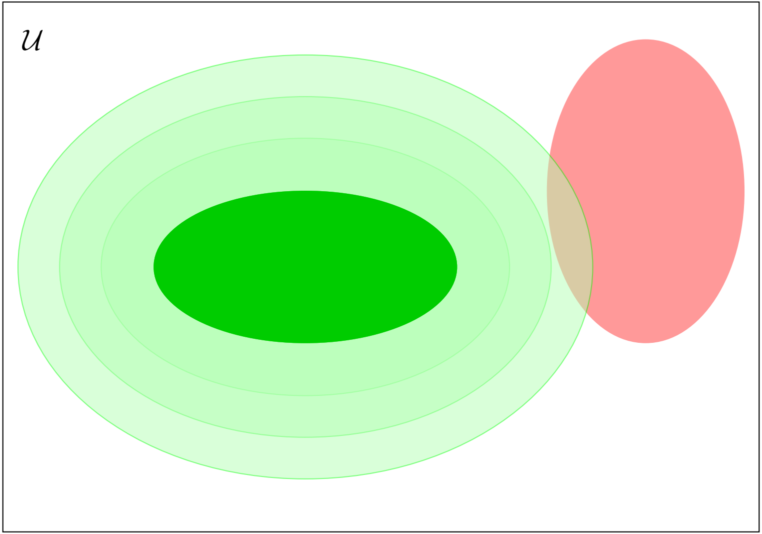
$$A \dot{+}^d B = \{\mathtt{b} \in B \mid d(A, \mathtt{b}) = d(A, B)\}$$



Belief revision (distance-based revision operators)

With d, a distance on $\mathcal{U}$, $A \subseteq \mathcal{U}$, $B \subseteq \mathcal{U}$:

$$A \dot{+}^d B = \{\mathtt{b} \in B \mid d(A, \mathtt{b}) = d(A, B)\}$$



Case-Based Reasoning

CBR: problems, solutions and cases

Assumption for the talk: problem-solution decomposition of cases

CBR: problems, solutions and cases

Assumption for the talk: problem-solution decomposition of cases

- A problem instance x: an element of $\mathcal{U}_{\mathtt{pb}}$

CBR: problems, solutions and cases

Assumption for the talk: problem-solution decomposition of cases

- A problem instance x: an element of $\mathcal{U}_{\mathtt{pb}}$
- A problem $\mathtt{pb}$: a class of problem instances, $\mathtt{pb} \in 2^{\mathcal{U}_{\mathtt{pb}}}$

CBR: problems, solutions and cases

Assumption for the talk: problem-solution decomposition of cases

- A problem instance x: an element of $\mathcal{U}_{\texttt{pb}}$
- A problem $\texttt{pb}$: a class of problem instances, $\texttt{pb} \in 2^{\mathcal{U}_{\texttt{pb}}}$
- A solution instance y: an element of $\mathcal{U}_{\texttt{sol}}$

CBR: problems, solutions and cases

Assumption for the talk: problem-solution decomposition of cases

- A problem instance x: an element of $\mathcal{U}_{\texttt{pb}}$
- A problem `pb`: a class of problem instances, $\texttt{pb} \in 2^{\mathcal{U}_{\texttt{pb}}}$
- A solution instance y: an element of $\mathcal{U}_{\texttt{sol}}$
- A solution `sol`: a class of solution instances, $\texttt{sol} \in 2^{\mathcal{U}_{\texttt{sol}}}$

CBR: problems, solutions and cases

Assumption for the talk: problem-solution decomposition of cases

- A problem instance x: an element of $\mathcal{U}_{\mathtt{pb}}$
- A problem $\mathtt{pb}$: a class of problem instances, $\mathtt{pb} \in 2^{\mathcal{U}_{\mathtt{pb}}}$
- A solution instance y: an element of $\mathcal{U}_{\mathtt{sol}}$
- A solution $\mathtt{sol}$: a class of solution instances, $\mathtt{sol} \in 2^{\mathcal{U}_{\mathtt{sol}}}$
- There exists a relation "x has for solution y" but it is incompletely known.

CBR: problems, solutions and cases

Assumption for the talk: problem-solution decomposition of cases

- A problem instance x: an element of $\mathcal{U}_{\mathtt{pb}}$
- A problem $\mathtt{pb}$: a class of problem instances, $\mathtt{pb} \in 2^{\mathcal{U}_{\mathtt{pb}}}$
- A solution instance y: an element of $\mathcal{U}_{\mathtt{sol}}$
- A solution $\mathtt{sol}$: a class of solution instances, $\mathtt{sol} \in 2^{\mathcal{U}_{\mathtt{sol}}}$
- There exists a relation "x has for solution y" but it is incompletely known.
- A case $\mathtt{C}$ is given by a problem $\mathtt{pb}$ and a solution $\mathtt{sol}(\mathtt{pb})$ of $\mathtt{pb}$: $\mathtt{C} = \mathtt{pb} \times \mathtt{sol}(\mathtt{pb})$.

CBR: problems, solutions and cases

Assumption for the talk: problem-solution decomposition of cases

- A problem instance x: an element of $\mathcal{U}_{\texttt{pb}}$
- A problem $\texttt{pb}$: a class of problem instances, $\texttt{pb} \in 2^{\mathcal{U}_{\texttt{pb}}}$
- A solution instance y: an element of $\mathcal{U}_{\texttt{sol}}$
- A solution $\texttt{sol}$: a class of solution instances, $\texttt{sol} \in 2^{\mathcal{U}_{\texttt{sol}}}$
- There exists a relation "x has for solution y" but it is incompletely known.
- A case $\texttt{C}$ is given by a problem $\texttt{pb}$ and a solution $\texttt{sol}(\texttt{pb})$ of $\texttt{pb}$: $\texttt{C} = \texttt{pb} \times \texttt{sol}(\texttt{pb})$.
- Thus, $\texttt{C} \in 2^{\mathcal{U}}$ where $\mathcal{U} = \mathcal{U}_{\texttt{pb}} \times \mathcal{U}_{\texttt{sol}}$.

CBR (input/output)

Input

Output

CBR (input/output)

Input

- The target case $\texttt{Target} = \texttt{tgt} \times \mathcal{U}_{\texttt{sol}}$ (unknown solution) (often $\texttt{tgt} = \{x^t\}$)

Output

CBR (input/output)

Input

- The target case $\texttt{Target} = \texttt{tgt} \times \mathcal{U}_{\texttt{sol}}$ (unknown solution) (often $\texttt{tgt} = \{x^t\}$)
- The case base $\texttt{CaseBase}$: set of the cases available to the system: $\texttt{Source} \in \texttt{CaseBase}$, $\texttt{Source} = \texttt{srce} \times \texttt{sol}(\texttt{srce})$ (often $\texttt{Source} = \{(x^s, y^s)\}$)

Output

CBR (input/output)

Input

- The target case $\mathtt{Target} = \mathtt{tgt} \times \mathcal{U}_{\mathtt{sol}}$ (unknown solution)
  (often $\mathtt{tgt} = \{x^t\}$)
- The case base $\mathtt{CaseBase}$: set of the cases available to the system: $\mathtt{Source} \in \mathtt{CaseBase}$, $\mathtt{Source} = \mathtt{srce} \times \mathtt{sol}(\mathtt{srce})$
  (often $\mathtt{Source} = \{(x^s, y^s)\}$)
- The domain knowledge $\mathtt{DK} \subseteq \mathcal{U}$
  a necessary condition for a case instance $\mathsf{a} = (x, y) \in \mathcal{U}$ to be licit

$$\text{if } \mathsf{a} \notin \mathtt{DK} \text{ then } \mathsf{a} \text{ is not licit}$$

Output

CBR (input/output)

Input

- The target case $\texttt{Target} = \texttt{tgt} \times \mathcal{U}_{\texttt{sol}}$ (unknown solution)
  (often $\texttt{tgt} = \{x^t\}$)
- The case base $\texttt{CaseBase}$: set of the cases available to the system: $\texttt{Source} \in \texttt{CaseBase}$, $\texttt{Source} = \texttt{srce} \times \texttt{sol}(\texttt{srce})$
  (often $\texttt{Source} = \{(x^s, y^s)\}$)
- The domain knowledge $\texttt{DK} \subseteq \mathcal{U}$
  a necessary condition for a case instance $\mathsf{a} = (x, y) \in \mathcal{U}$ to be licit

$$\text{if } \mathsf{a} \notin \texttt{DK} \text{ then } \mathsf{a} \text{ is not licit}$$

- Other knowledge containers (similarity, $\texttt{AK}$)

Output

CBR (input/output)

Input

- The target case $\mathtt{Target} = \mathtt{tgt} \times \mathcal{U}_{\mathtt{sol}}$ (unknown solution) (often $\mathtt{tgt} = \{x^t\}$)
- The case base $\mathtt{CaseBase}$: set of the cases available to the system: $\mathtt{Source} \in \mathtt{CaseBase}$, $\mathtt{Source} = \mathtt{srce} \times \mathtt{sol}(\mathtt{srce})$ (often $\mathtt{Source} = \{(x^s, y^s)\}$)
- The domain knowledge $\mathtt{DK} \subseteq \mathcal{U}$ a necessary condition for a case instance $\mathsf{a} = (x, y) \in \mathcal{U}$ to be licit

$$\text{if a} \notin \mathtt{DK} \text{ then a is not licit}$$

- Other knowledge containers (similarity, $\mathtt{AK}$)

Output

$\mathtt{sol}(\mathtt{tgt}) \in 2^{\mathcal{U}_{\mathtt{sol}}}$ such that the hypothesis "$\mathtt{sol}(\mathtt{tgt})$ solves $\mathtt{tgt}$" is likely (...). (often $\mathtt{sol}(\mathtt{tgt}) = \{y^t\}$)

CBR (retrieval and adaptation)

tgt

CBR (retrieval and adaptation)

$$\texttt{srce} \xleftarrow{\textit{retrieval}} \texttt{tgt}$$

CBR (retrieval and adaptation)

$$\begin{tikzcd}
\texttt{srce} \arrow[d] & & \texttt{tgt} \arrow[ll, "\textit{retrieval}"'] \\
\texttt{sol(srce)} & &
\end{tikzcd}$$

CBR (retrieval and adaptation)

$$
\begin{tikzcd}
\mathtt{srce} \arrow[d] & & \mathtt{tgt} \arrow[ll, "\textit{retrieval}"'] \\
\mathtt{sol(srce)} \arrow[rr, "\textit{adaptation}"'] & & \mathtt{sol(tgt)}
\end{tikzcd}
$$

CBR (retrieval and adaptation)

$$
\begin{tikzcd}
\texttt{srce} \arrow[d] & & \texttt{tgt} \arrow[ll, "\textit{retrieval}"'] \arrow[d] \\
\texttt{sol(srce)} \arrow[rr, "\textit{adaptation}"'] & & \texttt{sol(tgt)}
\end{tikzcd}
$$

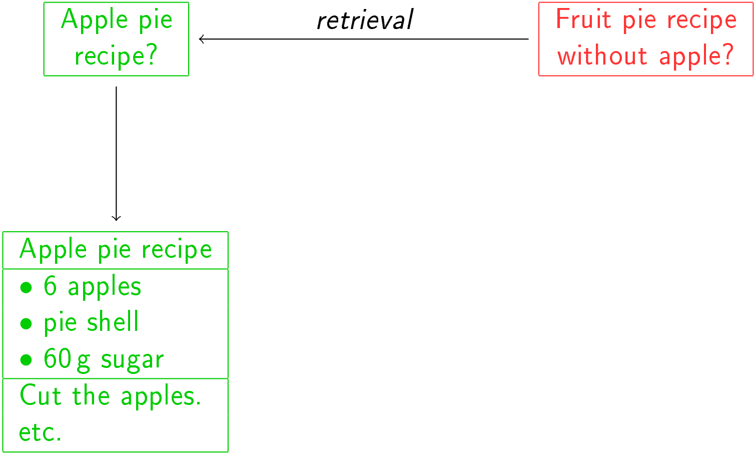
CBR (example) http://taaable.fr

Fruit pie recipe without apple?

CBR (example) http://taaable.fr

Apple pie recipe? $\xleftarrow{\textit{retrieval}}$ Fruit pie recipe without apple?

CBR (example) http://taaable.fr



CBR (example) http://taaable.fr

Apple pie recipe?

retrieval

Fruit pie recipe without apple?

Apple pie recipe

- 6 apples
- pie shell
- 60 g sugar

Cut the apples.
etc.

adaptation

$\sigma(\texttt{sol}(\texttt{srce}))$ with

CBR (example) http://taaable.fr

Apple pie recipe?

retrieval

Fruit pie recipe without apple?

Apple pie recipe

- 6 apples
- pie shell
- 60 g sugar

Cut the apples. etc.

adaptation

$\sigma(\texttt{sol}(\texttt{srce}))$ with

1. σ = apples $\rightsquigarrow$ pears

CBR (example) `http://taaable.fr`

Apple pie recipe? $\xleftarrow{\textit{retrieval}}$ Fruit pie recipe without apple?

Apple pie recipe
• 6 apples • pie shell • 60 g sugar
Cut the apples. etc.

$\xrightarrow{\textit{adaptation}}$

$\sigma(\texttt{sol}(\texttt{srce}))$ with

1. σ = apples $\rightsquigarrow$ pears
2. σ = apples $\rightsquigarrow$ pears and almond powder

CBR (example) `http://taaable.fr`

Fruit pie recipe without apple? $\xrightarrow{\textit{retrieval}}$ Apple pie recipe?

Apple pie recipe

- 6 apples
- pie shell
- 60 g sugar

Cut the apples.
etc.

$\xrightarrow{\textit{adaptation}}$

$\sigma(\texttt{sol}(\texttt{srce}))$ with

1. $\sigma =$ apples $\rightsquigarrow$ pears
2. $\sigma =$ apples $\rightsquigarrow$ pears and almond powder
3. $\sigma = \begin{matrix}\text{6 apples} \\ \text{60 g sugar}\end{matrix} \rightsquigarrow \begin{matrix}\text{9 pears} \\ \text{24 g sugar}\end{matrix}$

CBR (example) http://taaable.fr

Apple pie recipe?

retrieval

Fruit pie recipe without apple?

Apple pie recipe

- 6 apples
- pie shell
- 60 g sugar

Cut the apples.
etc.

adaptation

$\sigma(\mathtt{sol}(\mathtt{srce}))$ with

1. $\sigma =$ apples $\rightsquigarrow$ pears
2. $\sigma =$ apples $\rightsquigarrow$ pears and almond powder
3. $\sigma = \begin{matrix}\text{6 apples} \\ \text{60 g sugar}\end{matrix} \rightsquigarrow \begin{matrix}\text{9 pears} \\ \text{24 g sugar}\end{matrix}$
4. σ: adaptation of the preparation

Revision-Based CBR

Revision-based adaptation (intuition)

- Adaptation: modification of $\texttt{srce} \times \texttt{sol(srce)}$
  so that it is consistent with $\texttt{tgt} \times \mathcal{U}_{\texttt{sol}}$

$$//$$

- Revision: modification of ψ
  so that it is consistent with μ

Revision-based adaptation (definition)

$$\mathtt{tgt} \times \mathtt{sol}(\mathtt{tgt}) = (\qquad \mathtt{srce} \times \mathtt{sol}(\mathtt{srce})) \qquad (\qquad \mathtt{tgt} \times \mathcal{U}_{\mathtt{sol}})$$

Revision-based adaptation (definition)

$$\mathtt{tgt} \times \mathtt{sol(tgt)} = (\mathtt{DK} \cap \mathtt{srce} \times \mathtt{sol(srce)}) \qquad (\mathtt{DK} \cap \mathtt{tgt} \times \mathcal{U}_{\mathtt{sol}})$$

Revision-based adaptation (definition)

$$\mathtt{tgt} \times \mathtt{sol}(\mathtt{tgt}) = (\mathtt{DK} \cap \mathtt{srce} \times \mathtt{sol}(\mathtt{srce})) \dotplus \quad (\mathtt{DK} \cap \mathtt{tgt} \times \mathcal{U}_{\mathtt{sol}})$$

Revision-based adaptation (definition)

$$\mathtt{tgt} \times \mathtt{sol}(\mathtt{tgt}) = (\mathtt{DK} \cap \mathtt{srce} \times \mathtt{sol}(\mathtt{srce})) \dot{+}_{\mathtt{AK}} (\mathtt{DK} \cap \mathtt{tgt} \times \mathcal{U}_{\mathtt{sol}})$$

Revision-based adaptation (properties)

	revision postulates (reformulated)	$\dot{+}$-adaptation properties
$\dot{+}\,1$	$A \dot{+} B \subseteq B$	The result of the adaptation is consistent with DK.
$\dot{+}\,2$	if $A \cap B \neq \emptyset$ then $A \dot{+} B = A \cap B$	If the target case is consistent with the source case then this latter is deductively reused for solving the target case.
$\dot{+}\,3$	if $B \neq \emptyset$ then $A \dot{+} B \neq \emptyset$	Unless the target case is DK-inconsistent, the adaptation gives a consistent result.
$\dot{+}\,4$ $\dot{+}\,5$	if $(A \dot{+} B) \cap C \neq \emptyset$ then $A \dot{+} (B \cap C) = (A \dot{+} B) \cap C$	Adaptation by *minimal* modification (according to $\dot{+}$).
$\dot{+}\,6$	The result of the revision is representable in $\mathcal{L}$.	The result of the adaptation is representable in $\mathcal{L}$.

Example 1: in prop. logic, with Dalal revision operator

Example 1: in prop. logic, with Dalal revision operator

- $\texttt{tgt} = \texttt{pie} \wedge \texttt{fruit} \wedge \neg\texttt{apple}$

  (pie: recipe of pies, apple: recipes with apples, etc.)

Example 1: in prop. logic, with Dalal revision operator

- $\texttt{tgt} = \texttt{pie} \wedge \texttt{fruit} \wedge \neg\texttt{apple}$

  (pie: recipe of pies, apple: recipes with apples, etc.)
- $\texttt{srce} = \texttt{pie} \wedge \texttt{apple}$

  $\texttt{sol}(\texttt{srce}) = \texttt{pie\_shell} \wedge \texttt{sugar}$

Example 1: in prop. logic, with Dalal revision operator

- $\mathtt{tgt} = \mathtt{pie} \land \mathtt{fruit} \land \neg\mathtt{apple}$
  (pie: recipe of pies, apple: recipes with apples, etc.)
- $\mathtt{srce} = \mathtt{pie} \land \mathtt{apple}$
  $\mathtt{sol}(\mathtt{srce}) = \mathtt{pie\_shell} \land \mathtt{sugar}$
- $\mathtt{DK} = \left\{\begin{matrix} \mathtt{apple} \lor \mathtt{pear} \Leftrightarrow \mathtt{pome\_fruit}, & \\ \mathtt{pome\_fruit} \Rightarrow \mathtt{fruit}, & \text{etc.} \end{matrix}\right\}$

Example 1: in prop. logic, with Dalal revision operator

- $\mathtt{tgt} = \mathtt{pie} \wedge \mathtt{fruit} \wedge \neg\mathtt{apple}$

  (pie: recipe of pies, apple: recipes with apples, etc.)
- $\mathtt{srce} = \mathtt{pie} \wedge \mathtt{apple}$

  $\mathtt{sol}(\mathtt{srce}) = \mathtt{pie\_shell} \wedge \mathtt{sugar}$
- $\mathtt{DK} = \left\{\begin{matrix} \mathtt{apple} \vee \mathtt{pear} \Leftrightarrow \mathtt{pome\_fruit}, & \\ \mathtt{pome\_fruit} \Rightarrow \mathtt{fruit}, & \text{etc.} \end{matrix}\right\}$
- $\dot{+}$ = Dalal revision operator $= \dot{+}^{d_H}$ with d_H: Hamming distance

$$\begin{aligned} d_H(\mathcal{I}, \mathcal{J}) &= \text{number of flips of variables from } \mathcal{I} \text{ to } \mathcal{J} \\ &= \mathrm{card}\left(\{\text{variable } a \mid \mathcal{I}(a) \neq \mathcal{J}(a)\}\right) \end{aligned}$$

Example 1: in prop. logic, with Dalal revision operator

- $\texttt{tgt} = \texttt{pie} \wedge \texttt{fruit} \wedge \neg\texttt{apple}$
  (pie: recipe of pies, apple: recipes with apples, etc.)
- $\texttt{srce} = \texttt{pie} \wedge \texttt{apple}$
  $\texttt{sol}(\texttt{srce}) = \texttt{pie\_shell} \wedge \texttt{sugar}$
- $\texttt{DK} = \begin{Bmatrix} \texttt{apple} \vee \texttt{pear} \Leftrightarrow \texttt{pome\_fruit}, & \\ \texttt{pome\_fruit} \Rightarrow \texttt{fruit}, & \text{etc.} \end{Bmatrix}$
- $\dot{+} =$ Dalal revision operator $= \dot{+}^{d_H}$ with d_H: Hamming distance

$$\begin{aligned} d_H(\mathcal{I}, \mathcal{J}) &= \text{number of flips of variables from } \mathcal{I} \text{ to } \mathcal{J} \\ &= \text{card}\left(\{\text{variable } a \mid \mathcal{I}(a) \neq \mathcal{J}(a)\}\right) \end{aligned}$$

$$\texttt{tgt} \wedge \texttt{sol}(\texttt{tgt}) = \texttt{pie} \wedge \boxed{\texttt{pear}} \wedge \texttt{pie\_shell} \wedge \texttt{sugar}$$

Example 2: in prop. logic, using adaptation rules

Example 2: in prop. logic, using adaptation rules

- Same `tgt`, `srce`, `sol(srce)` and `DK`

Example 2: in prop. logic, using adaptation rules

- Same tgt, srce, sol(srce) and DK
- Use of adaptation knowledge given by adaptation rules

$$\mathtt{AK} = \left\{\begin{matrix} \mathtt{salad} \land \mathtt{vinegar} \rightsquigarrow \mathtt{salad} \land \mathtt{lemon} \land \mathtt{salt}, \\ \mathtt{apple} \rightsquigarrow \mathtt{pear} \land \mathtt{almond\_powder}, \qquad \text{etc.} \end{matrix}\right\}$$

Example 2: in prop. logic, using adaptation rules

- Same $\mathtt{tgt}$, $\mathtt{srce}$, $\mathtt{sol(srce)}$ and $\mathtt{DK}$
- Use of adaptation knowledge given by adaptation rules
  $$\mathtt{AK} = \left\{ \begin{array}{l} \mathtt{salad} \wedge \mathtt{vinegar} \leadsto \mathtt{salad} \wedge \mathtt{lemon} \wedge \mathtt{salt}, \\ \mathtt{apple} \leadsto \mathtt{pear} \wedge \mathtt{almond\_powder}, \qquad \text{etc.} \end{array} \right\}$$
- $\dot{+} = \dot{+}^{d_{\mathtt{AK}}}$ with $d_{\mathtt{AK}}$ defined by
  - Version 1:

$$d_{\mathtt{AK}}(\mathcal{I}, \mathcal{J}) = \text{ length of the shortest } \mathtt{AK}\text{-path from } \mathcal{I} \text{ to } \mathcal{J}$$

Example 2: in prop. logic, using adaptation rules

- Same $\mathtt{tgt}$, $\mathtt{srce}$, $\mathtt{sol(srce)}$ and $\mathtt{DK}$
- Use of adaptation knowledge given by adaptation rules
  $$\mathtt{AK} = \left\{ \begin{matrix} \mathtt{salad} \land \mathtt{vinegar} \rightsquigarrow \mathtt{salad} \land \mathtt{lemon} \land \mathtt{salt}, \\ \mathtt{apple} \rightsquigarrow \mathtt{pear} \land \mathtt{almond\_powder}, \qquad \text{etc.} \end{matrix} \right\}$$
- $\dot{+} = \dot{+}^{d_{\mathtt{AK}}}$ with $d_{\mathtt{AK}}$ defined by
  - Version 1:

$$d_{\mathtt{AK}}(\mathcal{I}, \mathcal{J}) = \text{ length of the shortest } \mathtt{AK}\text{-path from } \mathcal{I} \text{ to } \mathcal{J}$$

Limitation: $d_{\mathtt{AK}}(\mathcal{I}, \mathcal{J}) = +\infty$ if no $\mathtt{AK}$-path from $\mathcal{I}$ to $\mathcal{J}$ exist

Example 2: in prop. logic, using adaptation rules

- Same $\mathtt{tgt}$, $\mathtt{srce}$, $\mathtt{sol(srce)}$ and $\mathtt{DK}$
- Use of adaptation knowledge given by adaptation rules
$$\mathtt{AK} = \left\{ \begin{matrix} \mathtt{salad} \wedge \mathtt{vinegar} \rightsquigarrow \mathtt{salad} \wedge \mathtt{lemon} \wedge \mathtt{salt}, \\ \mathtt{apple} \rightsquigarrow \mathtt{pear} \wedge \mathtt{almond\_powder}, \qquad \text{etc.} \end{matrix} \right\}$$
- $\dot{+} = \dot{+}^{d_{\mathtt{AK}}}$ with $d_{\mathtt{AK}}$ defined by
  - Version 1:
$$d_{\mathtt{AK}}(\mathcal{I}, \mathcal{J}) = \text{ length of the shortest } \mathtt{AK}\text{-path from } \mathcal{I} \text{ to } \mathcal{J}$$
Limitation: $d_{\mathtt{AK}}(\mathcal{I}, \mathcal{J}) = +\infty$ if no $\mathtt{AK}$-path from $\mathcal{I}$ to $\mathcal{J}$ exist
  - Version 2:
$$d_{\mathtt{AK}}(\mathcal{I}, \mathcal{J}) = \begin{array}{l} \text{length of the shortest path from } \mathcal{I} \text{ to } \mathcal{J} \\ \text{using adaptation rules and flips of variables} \end{array}$$
(when $\mathtt{AK} = \emptyset$, $d_{\mathtt{AK}} = d_H$)

Example 2: in prop. logic, using adaptation rules

- Same $\mathtt{tgt}$, $\mathtt{srce}$, $\mathtt{sol}(\mathtt{srce})$ and $\mathtt{DK}$
- Use of adaptation knowledge given by adaptation rules
$$\mathtt{AK} = \left\{\begin{matrix} \mathtt{salad} \land \mathtt{vinegar} \rightsquigarrow \mathtt{salad} \land \mathtt{lemon} \land \mathtt{salt}, \\ \mathtt{apple} \rightsquigarrow \mathtt{pear} \land \mathtt{almond\_powder}, \qquad \text{etc.} \end{matrix}\right\}$$
- $\dot{+} = \dot{+}^{d_{\mathtt{AK}}}$ with $d_{\mathtt{AK}}$ defined by
  - Version 1:
  $$d_{\mathtt{AK}}(\mathcal{I}, \mathcal{J}) = \quad \text{length of the shortest } \mathtt{AK}\text{-path from } \mathcal{I} \text{ to } \mathcal{J}$$
  Limitation: $d_{\mathtt{AK}}(\mathcal{I}, \mathcal{J}) = +\infty$ if no $\mathtt{AK}$-path from $\mathcal{I}$ to $\mathcal{J}$ exist
  - Version 2:
  $$d_{\mathtt{AK}}(\mathcal{I}, \mathcal{J}) = \quad \begin{array}{l} \text{length of the shortest path from } \mathcal{I} \text{ to } \mathcal{J} \\ \text{using adaptation rules and flips of variables} \end{array}$$
  (when $\mathtt{AK} = \emptyset$, $d_{\mathtt{AK}} = d_H$)

$$\mathtt{tgt} \land \mathtt{sol}(\mathtt{tgt}) = \mathtt{pie} \land \boxed{\mathtt{pear} \land \mathtt{almond\_powder}} \land \mathtt{pie\_shell} \land \mathtt{sugar}$$

Example 3: adaptation of quantities (1/2)

Example 3: adaptation of quantities (1/2)

- $\mathcal{L}$: conjunction of linear constraints
  (on integers and real numbers)

Example 3: adaptation of quantities (1/2)

- $\mathcal{L}$: conjunction of linear constraints
  (on integers and real numbers)
- $\texttt{tgt} = (\texttt{nb}_{\texttt{apple}} = 0)$

Example 3: adaptation of quantities (1/2)

- $\mathcal{L}$: conjunction of linear constraints (on integers and real numbers)
- $\mathtt{tgt} = (\mathtt{nb}_{\mathtt{apple}} = 0)$
- $$\mathtt{srce} \wedge \mathtt{sol}(\mathtt{srce}) = \begin{array}{l} (\mathtt{nb}_{\mathtt{pie\_shell}} = 1) \wedge (\mathtt{nb}_{\mathtt{apple}} = 6) \wedge \\ (\mathtt{mass}_{\mathtt{sugar}} = 60) \wedge (\mathtt{nb}_{\mathtt{pear}} = 0) \wedge \\ (\mathtt{nb}_{\mathtt{tomato}} = 0) \wedge \ldots \end{array}$$

Example 3: adaptation of quantities (1/2)

- $\mathcal{L}$: conjunction of linear constraints (on integers and real numbers)
- $\texttt{tgt} = (\texttt{nb}_{\texttt{apple}} = 0)$
- $$\texttt{srce} \wedge \texttt{sol}(\texttt{srce}) = \begin{array}{l} (\texttt{nb}_{\texttt{pie\_shell}} = 1) \wedge (\texttt{nb}_{\texttt{apple}} = 6) \wedge \\ (\texttt{mass}_{\texttt{sugar}} = 60) \wedge (\texttt{nb}_{\texttt{pear}} = 0) \wedge \\ (\texttt{nb}_{\texttt{tomato}} = 0) \wedge \ldots \end{array}$$
- $$\texttt{DK} = \begin{array}{l} (\texttt{mass}_{\texttt{apple}} = 242 \times \texttt{nb}_{\texttt{apple}}) \wedge \\ (\texttt{mass}_{\texttt{pear}} = 166 \times \texttt{nb}_{\texttt{pear}}) \wedge \ldots \\ (\texttt{mass}_{\texttt{fruit}} = \texttt{mass}_{\texttt{apple}} + \texttt{mass}_{\texttt{pear}} + \ldots) \wedge \\ (\texttt{mass}_{\texttt{carbohydrates}} = \texttt{mass}_{\texttt{sugar}} + 33 \times \texttt{nb}_{\texttt{apple}} \\ \qquad\qquad + 26 \times \texttt{nb}_{\texttt{pear}} + \ldots) \wedge \ldots \end{array}$$

Example 3: adaptation of quantities (1/2)

- $\mathcal{L}$: conjunction of linear constraints (on integers and real numbers)
- $\mathtt{tgt} = (\mathtt{nb}_{\mathtt{apple}} = 0)$
- $\mathtt{srce} \wedge \mathtt{sol}(\mathtt{srce}) = \begin{array}{l} (\mathtt{nb}_{\mathtt{pie\_shell}} = 1) \wedge (\mathtt{nb}_{\mathtt{apple}} = 6) \wedge \\ (\mathtt{mass}_{\mathtt{sugar}} = 60) \wedge (\mathtt{nb}_{\mathtt{pear}} = 0) \wedge \\ (\mathtt{nb}_{\mathtt{tomato}} = 0) \wedge \ldots \end{array}$
- $\mathtt{DK} = \begin{array}{l} (\mathtt{mass}_{\mathtt{apple}} = 242 \times \mathtt{nb}_{\mathtt{apple}}) \wedge \\ (\mathtt{mass}_{\mathtt{pear}} = 166 \times \mathtt{nb}_{\mathtt{pear}}) \wedge \ldots \\ (\mathtt{mass}_{\mathtt{fruit}} = \mathtt{mass}_{\mathtt{apple}} + \mathtt{mass}_{\mathtt{pear}} + \ldots) \wedge \\ (\mathtt{mass}_{\mathtt{carbohydrates}} = \mathtt{mass}_{\mathtt{sugar}} + 33 \times \mathtt{nb}_{\mathtt{apple}} \\ \qquad\qquad + 26 \times \mathtt{nb}_{\mathtt{pear}} + \ldots) \wedge \ldots \end{array}$
- $x = (x_1, x_2, \ldots, x_n) \in \mathcal{U}$, $x_k \in \mathbb{R}_+$ or $x_k \in \mathbb{N}$

Example 3: adaptation of quantities (1/2)

- $\mathcal{L}$: conjunction of linear constraints (on integers and real numbers)
- $\mathtt{tgt} = (\mathtt{nb}_{\mathtt{apple}} = 0)$
- $\mathtt{srce} \wedge \mathtt{sol(srce)} = \begin{array}{l} (\mathtt{nb}_{\mathtt{pie\_shell}} = 1) \wedge (\mathtt{nb}_{\mathtt{apple}} = 6) \wedge \\ (\mathtt{mass}_{\mathtt{sugar}} = 60) \wedge (\mathtt{nb}_{\mathtt{pear}} = 0) \wedge \\ (\mathtt{nb}_{\mathtt{tomato}} = 0) \wedge \ldots \end{array}$
- $\mathrm{DK} = \begin{array}{l} (\mathtt{mass}_{\mathtt{apple}} = 242 \times \mathtt{nb}_{\mathtt{apple}}) \wedge \\ (\mathtt{mass}_{\mathtt{pear}} = 166 \times \mathtt{nb}_{\mathtt{pear}}) \wedge \ldots \\ (\mathtt{mass}_{\mathtt{fruit}} = \mathtt{mass}_{\mathtt{apple}} + \mathtt{mass}_{\mathtt{pear}} + \ldots) \wedge \\ (\mathtt{mass}_{\mathtt{carbohydrates}} = \mathtt{mass}_{\mathtt{sugar}} + 33 \times \mathtt{nb}_{\mathtt{apple}} \\ \qquad\qquad\qquad + 26 \times \mathtt{nb}_{\mathtt{pear}} + \ldots) \wedge \ldots \end{array}$
- $x = (x_1, x_2, \ldots, x_n) \in \mathcal{U}$, $x_k \in \mathbb{R}_+$ or $x_k \in \mathbb{N}$
- $\dot{+} = \dot{+}^d$ with $d(x, y) = \sum_k w_k \cdot |y_k - x_k|$

Example 3: adaptation of quantities (2/2)

- For some sets of weights:

$$\begin{aligned}\texttt{tgt} \land \texttt{sol}(\texttt{tgt}) = \texttt{DK} \land (\texttt{nb}_{\texttt{apple}} = 0) \\ \land \boxed{(\texttt{nb}_{\texttt{pear}} = 9) \land (\texttt{mass}_{\texttt{sugar}} = 24)}\end{aligned}$$

- Algorithm: transformation of the problem into a linear programming problem

Example 4: adaptation of preparations

Example 4: adaptation of preparations

- Representation of recipes using a temporal qualitative algebra

Example 4: adaptation of preparations

- Representation of recipes using a temporal qualitative algebra
- Using a revision operator on qualitative constraint networks

Conclusion and Future Work

Conclusion: Belief revision offers a useful framework (or a viewpoint) for specifying adaptation in CBR

Conclusion and Future Work

Conclusion: Belief revision offers a useful framework (or a viewpoint) for specifying adaptation in CBR

Future work

Conclusion and Future Work

Conclusion: Belief revision offers a useful framework (or a viewpoint) for specifying adaptation in CBR

Future work:

- What part of the adaptation processes does it cover?

Conclusion and Future Work

Conclusion Belief revision offers a useful framework (or a viewpoint) for specifying adaptation in CBR

Future work

- What part of the adaptation processes does it cover?
- What revision operator choosing? (adaptation knowledge acquisition issue)

Conclusion and Future Work

Conclusion Belief revision offers a useful framework (or a viewpoint) for specifying adaptation in CBR

Future work

- What part of the adaptation processes does it cover?
- What revision operator choosing?
  (adaptation knowledge acquisition issue)

There is other work not mentioned there
(but mentioned in the paper)