Applying Belief Revision to Case-Based Reasoning

Julien Cojan, Jean Lieber

Applying belief revision to case-based reasoning

Julien Cojan[1] and Jean Lieber[2,3,4]

[1] INRIA Sophia-Antipolis, Wimmics Project, France
[2] Université de Lorraine, LORIA, UMR 7503—54506 Vandœuvre-lès-Nancy, France
[3] CNRS—54506 Vandœuvre-lès-Nancy, France
[4] Inria—54602 Villers-lès-Nancy, France
Julien.Cojan@inria.fr
Jean.Lieber@loria.fr

Abstract. Adaptation is a task of case-based reasoning (CBR) that aims at modifying a case to solve a new problem. Now, belief revision deals also about modifications. This chapter studies how some results about revision can be applied to formalize adaptation and, more widely, CBR. Revision operators based on distances are defined in formalisms frequently used in CBR and applied to define an adaptation operator that takes into account the domain knowledge and the adaptation knowledge. This approach to adaptation is shown to generalize some other approaches to adaptation, such as rule-based adaptation.

1 Introduction

Case-based reasoning and belief revision are two domains in which the notions of similarity and modification play an important role.

Case-based reasoning (CBR [1]) is a reasoning process using a case base, where a case is a representation of a problem-solving episode, in general, in the form of a problem-solution pair. CBR aims at solving a *target problem* and generally consists in a retrieval step (selection of one or several case(s) from the case base that is/are similar to the target problem), an adaptation step (modification of the retrieved case(s) to propose a solution to the target problem), and a possible storage of the case formed by the target problem and its solution.

Belief revision is the process of changing a belief base about a static world by incorporating new beliefs while keeping the belief base consistent. When the old beliefs are inconsistent with the new beliefs, the formers have to be modified in order to restore consistency with the latters. Usually, belief revision is based on the minimal change principle [2]: most of the old beliefs should be kept. One way to measure change (so that it is minimal) is to use a similarity metric (to be maximized) or a distance (to be minimized).

Thus, the question raised is whether the modification performed during CBR could be performed by a belief revision operator. This question has been addressed in several publications and this chapter gives a synthesis of some of them.

The chapter is organized as follows. Some preliminaries about CBR are given in section 2. Section 3 introduces belief revision. In CBR, the modifications are performed during the adaptation step, section 4 is the core of the chapter and describes

revision-based adaptation from a theoretical viewpoint and with a few examples. More globally, belief revision can be applied to CBR as a whole as section 5 shows. Several revision operators have been implemented for different formalisms, together with their revision-based adaptation functions. Some of them are gathered in REVISOR (`http://revisor.loria.fr`), a system that is briefly described in section 6. Finally section 7 concludes the chapter.

2 Preliminaries

2.1 Formalism

The approach to CBR presented in this chapter can be applied to a variety of representation languages. It is assumed that there exists a representation language $\mathcal{L}$: a formula is an element of $\mathcal{L}$. The semantics of $\mathcal{L}$ is given by a (possibly infinite) set $\mathcal{U}$ and by a function $\text{Mod} : \varphi \in \mathcal{L} \mapsto \text{Mod}(\varphi) \in 2^{\mathcal{U}}$, defining, in a model-theoretical manner, the semantics of $\mathcal{L}$: $\mathtt{a}$ is a model of φ if $\mathtt{a} \in \text{Mod}(\varphi)$; φ_1 entails φ_2 ($\varphi_1 \models \varphi_2$) if $\text{Mod}(\varphi_1) \subseteq \text{Mod}(\varphi_2)$; φ_1 and φ_2 are equivalent ($\varphi_1 \equiv \varphi_2$) if $\text{Mod}(\varphi_1) = \text{Mod}(\varphi_2)$. A subset A of $\mathcal{U}$ is *representable* in $\mathcal{L}$ if there exists a formula φ such that $\text{Mod}(\varphi) = A$.

It is assumed that $\mathcal{L}$ is stable under conjunction, which means that for every $\varphi_1, \varphi_2 \in \mathcal{L}$ there exists a formula denoted by $\varphi_1 \wedge \varphi_2$ such that $\text{Mod}(\varphi_1 \wedge \varphi_2) = \text{Mod}(\varphi_1) \cap \text{Mod}(\varphi_2)$.

Some formalisms are stable under negation (or complement), which means that for every $\varphi \in \mathcal{L}$, there exists a formula denoted by $\neg\varphi$ such that $\text{Mod}(\neg\varphi) = \mathcal{U} \setminus \text{Mod}(\varphi)$. For such formalisms, $\varphi_2 \vee \varphi_2$ is an abbreviation for $\neg(\neg\varphi_1 \wedge \neg\varphi_2)$, $\varphi_1 \Rightarrow \varphi_2$ is an abbreviation for $\neg\varphi_1 \vee \varphi_2$ and $\varphi_1 \Leftrightarrow \varphi_2$ is an abbreviation for $(\varphi_1 \Rightarrow \varphi_2) \wedge (\varphi_2 \Rightarrow \varphi_1)$.

Propositional logic with n variables is an example of such a formalism: $\mathcal{U}$ denotes the set of interpretations on the variables. Every $A \subseteq \mathcal{U}$ is representable in this logic.

2.2 Case-based reasoning: principles and notations

For CBR, $\mathcal{U}$ is called the *case universe*. A *case instance* $\mathtt{a}$ is, by definition, an element of $\mathcal{U}$: $\mathtt{a} \in \mathcal{U}$. A *case* $\mathtt{C}$ is a class of case instances: $\mathtt{C} \in 2^{\mathcal{U}}$ (in this chapter, a case represents a class of experiences, it is what is called an ossified case in [1] and a generalized case in [3]). For instance, when the formalism is propositional logic with n variables, $\mathcal{U}$ is the set of the 2^n interpretations and a case $\mathtt{C}$ is represented by a formula φ: $\mathtt{C} = \text{Mod}(\varphi)$.

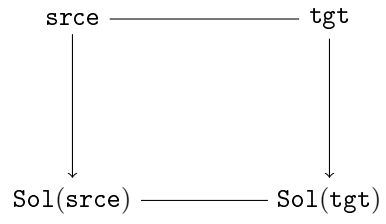
A *source case* is denoted by `Source`: it is a case of `CaseBase` (the case base). The *target case* is denoted by `Target`: it is the input of the CBR system. In many applications, the source cases `Source` are *specific*: each of them represents a single case instance $\mathtt{a}$ ($\mathtt{Source} = \{\mathtt{a}\}$). By contrast, the target case specifies only its "problem part" and needs to be completed by a "solution part". The aim of the CBR process is to perform this completion:

$$\begin{aligned} &\mathtt{CBR} : (\mathtt{CaseBase}, \mathtt{Target}) \mapsto \mathtt{ComplTarget} \\ &\qquad \text{with } \mathtt{ComplTarget} \subseteq \mathtt{Target} \end{aligned} \tag{1}$$

Usually, this inference is decomposed into two steps:

$$\texttt{Retrieval} : (\texttt{CaseBase}, \texttt{Target}) \mapsto \texttt{Source} \in \texttt{CaseBase}$$
$$\texttt{Adaptation} : (\texttt{Source}, \texttt{Target}) \mapsto \texttt{ComplTarget}$$

In many CBR applications, a case instance $\texttt{a}$ can be decomposed into a problem part x and a solution part y: $\texttt{a} = (x, y)$. Let $\mathcal{U}_{\texttt{pb}}$ and $\mathcal{U}_{\texttt{sol}}$ be the universes of problem and solution instances: $x \in \mathcal{U}_{\texttt{pb}}$, $y \in \mathcal{U}_{\texttt{sol}}$, $\mathcal{U} = \mathcal{U}_{\texttt{pb}} \times \mathcal{U}_{\texttt{sol}}$. A source case $\texttt{Source}$ is decomposed in a *source problem* $\texttt{srce} \in 2^{\mathcal{U}_{\texttt{pb}}}$ and its solution $\texttt{Sol}(\texttt{srce}) \in 2^{\mathcal{U}_{\texttt{sol}}}$, thus $\texttt{Source} = \texttt{srce} \times \texttt{Sol}(\texttt{srce})$ that is interpreted as: for all $x \in \texttt{srce}$ there exists $y \in \texttt{Sol}(\texttt{srce})$ such that $\texttt{a} = (x, y)$ is a licit case (i.e., y solves x). The solution part of the target problem is unknown, thus $\texttt{Target} = \texttt{tgt} \times \mathcal{U}_{\texttt{sol}}$, where $\texttt{tgt}$ is called the *target problem*. When cases are decomposed in problems and solutions, CBR aims at solving the target problem $\texttt{tgt}$, thus $\texttt{ComplTarget} = \texttt{tgt} \times \texttt{Sol}(\texttt{tgt})$ where $\texttt{Sol}(\texttt{tgt}) \in 2^{\mathcal{U}_{\texttt{sol}}}$. In general, a target problem is specific: it is a singleton, i.e., $\texttt{tgt} = \{x^t\}$ with $x^t \in \mathcal{U}_{\texttt{pb}}$. This problem-solution decomposition is not needed to present revision-based adaptation but it is a prerequisite for other approaches to adaptation mentioned in the chapter. When cases are decomposed in problem and solution parts, it is common to consider the adaptation problem as an analogical problem represented by the following diagram:



(2)

that can be read as "$\texttt{Sol}(\texttt{tgt})$ is to $\texttt{Sol}(\texttt{srce})$ as $\texttt{tgt}$ is to $\texttt{srce}$" (transformational analogy [4]) or "$\texttt{Sol}(\texttt{tgt})$ is to $\texttt{tgt}$ as $\texttt{Sol}(\texttt{srce})$ is to $\texttt{srce}$" (derivational analogy [5]). In other words, adaptation aims at solving an analogical problem.

The domain knowledge $\texttt{DK}$ is a knowledge base giving a necessary condition for a case instance to be licit. Thus, the domain knowledge can be represented by a subset $\texttt{DK}$ of $\mathcal{U}$ and for each $\texttt{a} \in \mathcal{U}$, $\texttt{a} \notin \texttt{DK}$ involves that $\texttt{a}$ is not licit. When the case universe is decomposed in $\mathcal{U}_{\texttt{pb}} \times \mathcal{U}_{\texttt{sol}}$, $\texttt{a} = (x, y) \notin \texttt{DK}$ means that y is not a solution of x or that x and/or y are meaningless (i.e., they are objects represented in the language that have no correspondence in the real world, e.g., in the domain of zoology, a cat that is not a mammal). Having no domain knowledge (or not taking it into account) amounts to $\texttt{DK} = \mathcal{U}$.

Each source case is assumed to be consistent with the domain knowledge, i.e.,

$$\texttt{DK} \cap \texttt{Source} \neq \emptyset \tag{3}$$

Similarly, if a target case is inconsistent with the domain knowledge, it has not to be considered for the CBR inference (the system has to reject it before the retrieval). Thus, if $\texttt{Target}$ is an input of the adaptation procedure, it is required that

$$\texttt{DK} \cap \texttt{Target} \neq \emptyset \tag{4}$$

The result of adaptation must also be consistent with DK, therefore:

$$\texttt{DK} \cap \texttt{ComplTarget} \neq \emptyset \tag{5}$$

(It can be noted that (4) is a consequence of (1) and (5).)

2.3 Distances and metric spaces

A *distance* on a set $\mathcal{U}$ is defined in this chapter as a function $d : \mathcal{U} \times \mathcal{U} \to [0; +\infty]$ such that $d(\texttt{a}, \texttt{b}) = 0$ iff $\texttt{a} = \texttt{b}$ (the properties of symmetry and triangle inequality are not required in this chapter). Let $\texttt{b} \in \mathcal{U}$ and A, $B \in 2^{\mathcal{U}}$. The usual abbreviations are used:

$$d(A, \texttt{b}) = \inf_{\texttt{a} \in A} d(\texttt{a}, \texttt{b}) \qquad d(\texttt{b}, A) = \inf_{\texttt{a} \in A} d(\texttt{b}, \texttt{a}) \qquad d(A, B) = \inf_{\texttt{a} \in A, \texttt{b} \in B} d(\texttt{a}, \texttt{b})$$

By convention, the infimum on the empty set is $+\infty$, e.g., $d(A, \emptyset) = +\infty$. A is said to be closed under d if

$$\{\texttt{b} \in \mathcal{U} \mid d(A, \texttt{b}) = 0\} = \{\texttt{b} \in \mathcal{U} \mid d(\texttt{b}, A) = 0\} = A \qquad \text{(closeness of } A\text{)}$$

A *metric space* is an ordered pair $(\mathcal{U}, d)$ where d is a distance on $\mathcal{U}$.[5]

If $\mathcal{U} = \mathbb{R}^n$ (where $\mathbb{R}$ is the set of the real numbers), a L_1-distance is a distance d parametrized by a base $\mathcal{B}$ of the vector space $\mathcal{U}$ such that

$$d(\texttt{a}, \texttt{b}) = \sum_{i=1}^{n} |v_i - u_i| \qquad (L_1 \text{ distance})$$

where $(u_1, u_2, \ldots, u_n)$ (resp., $(v_1, v_2, \ldots, v_n)$) is the representation of a (resp., of b) in $\mathcal{B}$. When $\mathcal{B}$ is the canonical base, $u_i = \texttt{a}_i$ and $v_i = \texttt{b}_i$ for each $i \in \{1, 2, \ldots n\}$. By extension, if $\mathcal{U}$ is a subset of $\mathbb{R}^n$, a distance d on $\mathcal{U}$ that is the restriction of a L_1 distance on $\mathbb{R}^n$ is also called a L_1 distance on $\mathcal{U}$ (for example, if $\mathcal{U} = \mathbb{Z}^n$ where $\mathbb{Z}$ is the set of integers).

If $\mathcal{U} = \mathbb{B}^n$ where $\mathbb{B} = \{\texttt{true}, \texttt{false}\}$, the weighted Hamming distance with weights $(w_1, w_2, \ldots, w_n)$ (where $w_i > 0$) is defined, for $\texttt{a} = (\texttt{a}_1, \texttt{a}_2, \ldots, \texttt{a}_n)$ and $\texttt{b} = (\texttt{b}_1, \texttt{b}_2, \ldots, \texttt{b}_n)$ by

$$d(\texttt{a}, \texttt{b}) = \sum_{i=1}^{n} \begin{cases} 0 & \text{if } \texttt{a}_i = \texttt{b}_i \\ w_i & \text{else} \end{cases}$$

The Hamming distance is the weighted Hamming distance with $w_i = 1$ for each $i \in \{1, 2, \ldots, n\}$. An interpretation in propositional logic with n variables is assimilated to an element of $\mathbb{B}^n$, hence the notion of weighted Hamming distance between such interpretations.

[5] Calling it a metric space is an abuse of the usual mathematical language, since d does not necessarily verifies all the postulates of a distance metric. However, the term metric space is used in this chapter with this generalized meaning, for the sake of simplicity.

3 Belief revision

3.1 Belief revision in propositional logic

In [2], postulates of belief revision are proposed in a general logical setting. These postulates are based on the idea of minimal change. They are applied to propositional logic in [6] which presents 6 postulates that a belief operation $\dot{+}$ has to verify in this formalism: given ψ and μ, two belief bases, $\psi \dot{+} \mu$ is a revision of ψ by μ. One of these postulates states that $\dot{+}$ is independent to syntax:

$$\text{if } \psi_1 \equiv \psi_2 \text{ and } \mu_1 \equiv \mu_2 \text{ then } \psi_1 \dot{+} \mu_1 \equiv \psi_2 \dot{+} \mu_2 \tag{6}$$

where ψ_1, ψ_2, μ_1, and μ_2 are formulas representing beliefs. As a consequence of (6), a formula φ can be assimilated to $\text{Mod}(\varphi)$, the set of its models: in the rest of the chapter, formulas and subsets of $\mathcal{U}$ are used indifferently. Then, the other 5 postulates can be rewritten as follows (using subsets of $\mathcal{U}$, the set of interpretations, instead of propositional formulas):

($\dot{+}1$) $A \dot{+} B \subseteq B$.
($\dot{+}2$) If $A \cap B \neq \emptyset$ then $A \dot{+} B = A \cap B$.
($\dot{+}3$) If $B \neq \emptyset$ then $A \dot{+} B \neq \emptyset$.
($\dot{+}4$) $(A \dot{+} B) \cap C \subseteq A \dot{+} (B \cap C)$.
($\dot{+}5$) If $(A \dot{+} B) \cap C \neq \emptyset$ then $A \dot{+} (B \cap C) \subseteq (A \dot{+} B) \cap C$.

(A, B, and C are subsets of $\mathcal{U}$.) The interpretation of these postulates is made further (in section 4.3), for their application to revision-based adaptation.

Intuitively, to revise A by B, the idea is to modify minimally A into A' so that $A' \cap B \neq \emptyset$, and then $A \dot{+} B = A' \cap B$. Now, there are many ways to model minimal modifications. Among them, there is the modification based on a distance d on $\mathcal{U}$ (figure 1 illustrates it): given $\lambda \in \mathbb{R}$ with $\lambda \geq 0$, $G^d_\lambda(A)$ is the generalization (a kind of modification) of $A \subseteq \mathcal{U}$ defined by

$$G^d_\lambda(A) = \{\mathtt{b} \in \mathcal{U} \mid d(A, \mathtt{b}) \leq \lambda\}$$

Then, the revision operator $\dot{+}^d$ is defined by

$$A \dot{+}^d B = G^d_\delta(A) \cap B \qquad \text{where } \delta = \inf\{\lambda \mid G^d_\lambda(A) \cap B \neq \emptyset\}$$

Note that the infima on $\mathcal{U}$ are always reached when $\mathcal{U}$ is finite, which is the case when $\mathcal{U}$ is the set of interpretations over n propositional variables. This kind of revision operators is a direct generalization of the Dalal revision operator [7], which is based on the Hamming distance between propositional interpretations. The following equivalent definition can be given:

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

(the δ's in the two definitions are equal).

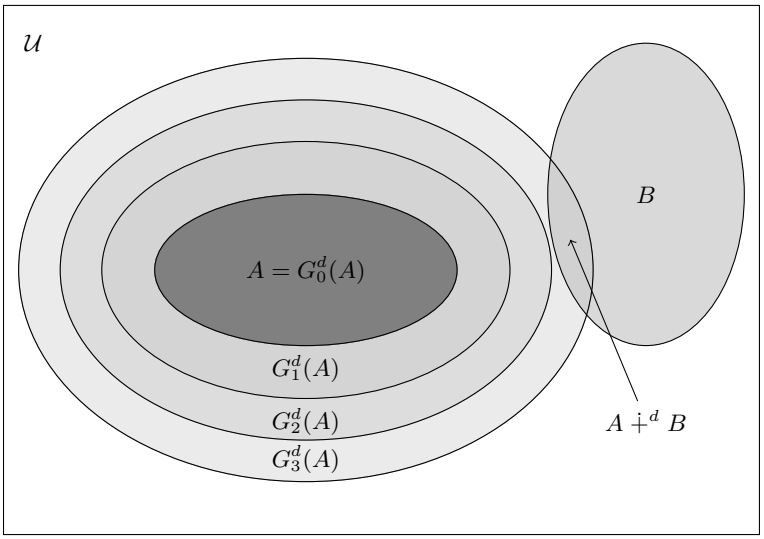


Fig. 1: Illustration of a distance-based revision operator in propositional logic. The set of interpretations is mapped to a finite subset of the plan and the distance between interpretations corresponds to a distance in the plan. A is generalized step by step ($G_1^d(A)$, $G_2^d(A)$, etc.) until the generalized step $A' = G_\lambda^d(A)$ intersects with B. The result is $A \dot{+}^d B = A' \cap B$.

3.2 Belief revision in a metric space

The ($\dot{+}$1-5) postulates can be straightforwardly generalized to other formalisms $\mathcal{L}$ where $\mathcal{U}$ is (a priori) any set and each formula φ of $\mathcal{L}$ is assimilated to a subset Mod(φ) of $\mathcal{U}$. In this generalization, A, B and C are sets assumed to be representable in $\mathcal{L}$.

Given a distance d on $\mathcal{U}$, $\dot{+}^d$ can be defined as above but it must be noticed that $\dot{+}^d$ may not satisfy the ($\dot{+}$1-5) postulates. This issue is considered further, in section 4.4.

The representability issue must also be addressed in this generalization, thus we propose the following postulate:

($\dot{+}$6) For any $\psi, \mu \in \mathcal{L}$, there exists $\rho \in \mathcal{L}$ such that Mod(ρ) = Mod(ψ) $\dot{+}$ Mod(μ).

In propositional logic, for any operator $\dot{+}$, this postulateis particularly holds, since every subset of the set $\mathcal{U}$ of the interpretations of propositional variables is representable in this logic.

3.3 Integrity constraint belief merging

Let ψ_1, ψ_2, …, ψ_k, and μ be $k+1$ belief bases. Merging ψ_1, ψ_2, …, ψ_k, given the integrity constraint μ consists in building a belief base φ such that $\varphi \models \mu$ and φ keeps "as much as possible" from the ψ_i's. A merging operator

$$\triangle : \left(\mu, \{\psi_i\}_{1 \leq i \leq k}\right) \mapsto \varphi = \triangle_\mu \left(\{\psi_i\}_{1 \leq i \leq k}\right)$$

is assumed to satisfy some postulates similar to the postulates for a revision operator (in [8], such postulates are defined in propositional logic but can be easily generalized to metric spaces). Actually, the notion of integrity constraint extends the notion of belief revision in the sense that if $\triangle$ is such a merging operator, then $\dot{+}$ defined by $\psi \dot{+} \mu = \triangle_\mu(\{\psi\})$ satisfies the ($\dot{+}$1-5) postulates.

4 Revision-based adaptation

This section defines revision-based adaptation (subsection 4.1), presents an example in propositional logic (subsection 4.2), studies its properties (subsection 4.3), describes with details revision-based adaptation in metric spaces (subsection 4.4), mentions briefly an extension to multiple case adaptation (subsection 4.5), relates this approach to adaptation with other approaches to adaptation (subsection 4.6) and gives pointers on other work related to revision-based adaptation (subsection 4.7).

4.1 Definition

Let $\mathcal{U}$ be the case universe and $\dot{+}$ be a revision operator on $\mathcal{U}$. The $\dot{+}$-adaptation is defined as follows [9]:

$$\texttt{ComplTarget} = (\texttt{DK} \cap \texttt{Source}) \dot{+} (\texttt{DK} \cap \texttt{Target}) \tag{7}$$

$(\texttt{DK} \cap \texttt{Source})$ (resp., $(\texttt{DK} \cap \texttt{Target})$) is the source (resp., target) case interpreted within the domain knowledge (i.e., case instances known to be not licit are removed). Thus (7) can be interpreted as a minimal modification of the source case to satisfy the target case, given the domain knowledge, knowing that the minimality of modification is the one associated with the operator $\dot{+}$.

4.2 Example in propositional logic

Let us consider the following story. Léon is about to invite Thècle and wants to prepare her an appropriate meal. His target problem can be specified by the characteristics of Thècle about food. Let us assume that Thècle is vegetarian (denoted by the propositional variable v) and that she has other characteristics (denoted by o) not detailed in this example:

$$\texttt{Target} = v \wedge o$$

From his experience as a host, Léon remembers that he had invited Simone some times ago and he thinks that Simone is very similar to Thècle according to food behavior, except that she is not a vegetarian ($\neg v \wedge o$). He had proposed to Simone a meal with salad (s), beef (b), and a dessert (d), and she was satisfied by the two formers but has not eaten the dessert, thus Léon has retained the source case

$$\texttt{Source} = (\neg v \wedge o) \wedge (s \wedge b \wedge \neg d)$$

Besides that, Léon has some general knowledge about food: he knows that beef is meat, that meat and tofu are protein foods, and that vegetarians do not eat meat. Moreover, the

only protein food that he is willing to cook, apart from meat, is tofu. Thus, his domain knowledge is

$$\texttt{DK} \quad = \quad b \Rightarrow m \quad \wedge \quad m \vee t \Leftrightarrow p \quad \wedge \quad v \Rightarrow \neg m$$

where b, m, t, and p are the propositional variables for "some beef/meat/tofu/protein food is appreciated by the current guest". According to $\dot{+}$-adaptation, what meal should be proposed to Thècle? If $\dot{+}$ is the Dalal revision operator, the $\dot{+}$-adaptation of the meal for Simone to a meal for Thècle is

$$\texttt{ComplTarget} \equiv \texttt{DK} \wedge \texttt{Target} \wedge (s \wedge t \wedge \neg d)$$

In [9], this adaptation is qualified as conservative: the salad and the absence of dessert is reused for the target case and, though the beef is not kept (to ensure a consistent result), the consequence of b that is consistent with DK, i.e., p, is kept, and thus, t is proposed instead of beef (since $v \wedge p \models_{\texttt{DK}} t$; in other words, some protein food is required, the only vegetarian protein that Léon is willing to cook is tofu, thus there will be tofu in the meal).

4.3 Properties

The ($\dot{+}$1-6) postulates entail some properties of revision-based adaptation.

($\dot{+}$1) applied to $\dot{+}$-adaptation gives $\texttt{ComplTarget} \subseteq \texttt{DK} \cap \texttt{Target}$, which entails the property (1) required for the adaptation process (cf. section 2.2) and the fact that $\texttt{ComplTarget} \subseteq \texttt{DK}$ (no instance case a known to be illicit—$\texttt{a} \in \mathcal{U} \setminus \texttt{DK}$—is in the result).

Let us assume that $\texttt{DK} \cap \texttt{Source} \cap \texttt{Target} \neq \emptyset$. Then, ($\dot{+}$2) entails that $\texttt{ComplTarget} = \texttt{DK} \cap \texttt{Source} \cap \texttt{Target}$. This means that if the target case is consistent with the source case, given the domain knowledge, then it can be inferred by $\dot{+}$-adaptation that Source solves Target. This is consistent with the principle of this kind of adaptation: ComplTarget is obtained by keeping from Source as much as possible, and if no modification is needed then no modification is applied.

($\dot{+}$3) gives: if $\texttt{DK} \cap \texttt{Target} \neq \emptyset$ then $\texttt{ComplTarget} \neq \emptyset$. Since $\texttt{ComplTarget} \subseteq \texttt{DK}$ (cf. ($\dot{+}$1)), $\texttt{DK} \cap \texttt{ComplTarget} \neq \emptyset$, which is a property required by an adaptation operator (cf. equation (5), section 2.2).

According to [6], ($\dot{+}$4) and ($\dot{+}$5) capture the minimality of modifications. Thus they express the minimality of modification made by a $\dot{+}$-adaptation. This can be interpreted as follows. The conjunction of ($\dot{+}$4) and ($\dot{+}$5) can be reformulated as:

$$\begin{cases} \text{Either } (A \dot{+} B) \cap C = \emptyset, \\ \text{Or } (A \dot{+} B) \cap C = A \dot{+} (B \cap C). \end{cases} \tag{8}$$

Let F represent some additional features about the target problem: the new target case is $\texttt{Target}_2 = \texttt{Target} \cap F$. If ComplTarget is consistent with F, then ($\dot{+}$4) and ($\dot{+}$5) entail that the adaptation of Source to $\texttt{Target}_2$ gives $\texttt{ComplTarget}_2 = \texttt{ComplTarget} \cap F$. In other words, if F does not involve needs on modifications (corresponding to an inconsistency) then the result of the $\dot{+}$-adaptation can be reused straightforwardly.

($\dot{+}$6) involves that ComplTarget is representable in $\mathcal{L}$.

4.4 Revision-based adaptation in metric spaces

In this section, $\dot{+}^d$-adaptation is considered on a metric space $(\mathcal{U}, d)$. This study is motivated by applications of CBR which are frequently based on attribute-value formalisms, where the attributes range frequently in simple domains (numeric domains, Boolean, etc.). Thus, the first idea was to study revision for such formalisms, but it appeared that the more general framework of metric space was a better level of study (particularities of attribute-value formalisms did not add much to the study). Therefore, this has involved the necessity to study revision in this framework. This section goes from general to specific. First, revision in $(\mathcal{U}, d)$ is considered according to the revision postulates. Then, it is applicated to attribute-constraint formalisms (that include attribute-value formalisms), which is further applicated to formalisms of linear constraints, leading to a practical algorithm for revision in this case. Finally, a practical application to the cooking domain is presented.

The ($\dot{+}$1-6) postulates in metric spaces

This section studies the revision postulates for $\dot{+}^d$. Some of these postulates are not satisfied by $\dot{+}^d$ and some additional assumptions on the representation language $\mathcal{L}$ and on d are proposed that are sufficient conditions for their satisfaction (recall that the subsets A, B, and C of $\mathcal{U}$ involved in postulates ($\dot{+}$1-5) are representable in $\mathcal{L}$).

($\dot{+}$1) is always satisfied by $\dot{+}^d$ (direct consequence from the definition of $\dot{+}^d$).

($\dot{+}$2) is not always satisfied by $\dot{+}^d$ as the following counterexample shows. Let $\mathcal{U} = \mathbb{R}$ and let $\mathcal{L}$ be the language of intervals of $\mathbb{R}$, e.g., $[0;1[= \{\mathtt{a} \in \mathcal{U} \mid 0 \leq \mathtt{a} < 1\}$. Let $d : (\mathtt{a}, \mathtt{b}) \mapsto |\mathtt{b} - \mathtt{a}|$. It can be shown that $[0;1[\dot{+}^d [0;1[= [0;1] \not\subseteq [0;1[$. Now, let us consider the following additional assumption about the formalism $\mathcal{L}$:

($\mathcal{L}$1) Every subset A of $\mathcal{U}$ that can be represented in $\mathcal{L}$ is closed under d.

Under this assumption, ($\dot{+}$2) is satisfied as proven hereafter. Let A and B be two subsets of $\mathcal{U}$ such that A is closed and $A \cap B \neq \emptyset$. Thus, $d(A, B) = 0$ and

$$\begin{aligned} A \dot{+}^d B &= \{\mathtt{b} \in B \mid d(A, \mathtt{b}) = 0\} \\ &= \{\mathtt{b} \in \mathcal{U} \mid d(A, \mathtt{b}) = 0\} \cap B \\ &= A \cap B \qquad \text{since } A \text{ is closed} \end{aligned}$$

Therefore $A \dot{+}^d B = A \cap B$ and ($\dot{+}$2) is satisfied.

($\dot{+}$3) is not always satisfied by $\dot{+}^d$ as the following counterexample shows. Let $\mathcal{U}$ and d be the same as in the counterexample of ($\dot{+}$2). Let $A = [0;1]$ and $B =]2;3]$. $B \neq \emptyset$ but $A \dot{+}^d B = \emptyset$. This suggests that A and B should be closed but even if the ($\mathcal{L}$1) assumption was made, ($\dot{+}$3) may be not satisfied. Figure 2 presents a counterexample of ($\dot{+}$3) with A and B, two closed sets. Now, let us consider the following assumption:

($\mathcal{L}$2) For every A and B non empty subsets of $\mathcal{U}$ representable in $\mathcal{L}$, the distance between A and B is always reached: there exist $\mathtt{a} \in A$ and $\mathtt{b} \in B$ such that $d(A, B) = d(\mathtt{a}, \mathtt{b})$.

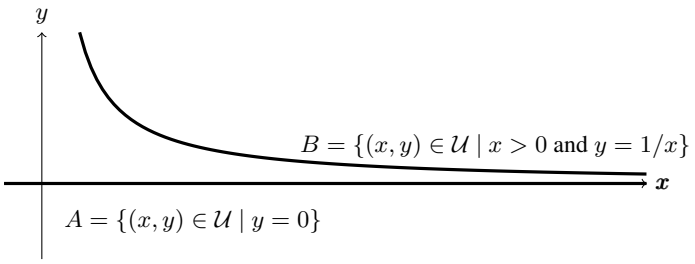


Fig. 2: A counterexample of ($\dot{+}3$). $\mathcal{U} = \mathbb{R}^2$, d is a L_1 distance on $\mathcal{U}$, $\mathcal{L}$ is chosen so that A, B and $\emptyset$ are representable in this formalism, which is consistent with assumption ($\mathcal{L}1$), since A and B are closed under d. Finally, $B \neq \emptyset$. Despite these facts, $A \dot{+} B = \emptyset$ (since $d(A, B) = 0$ and no element of B is at distance 0 from an element of A).

Under this assumption, ($\dot{+}3$) is satisfied as proven hereafter. Let $A, B \in 2^{\mathcal{U}}$ such that $B \neq \emptyset$. If $A = \emptyset$, $d(A, B) = d(A, \mathtt{b}) = +\infty$ for every $\mathtt{b} \in B$, thus $A \dot{+}^d B = B \neq \emptyset$. If $A \neq \emptyset$ and if A and B are representable in $\mathcal{L}$ then ($\mathcal{L}2$) entails that $d(A, B) = d(\mathtt{a}, \mathtt{b})$ for some $(\mathtt{a}, \mathtt{b}) \in A \times B$. Since $d(\mathtt{a}, \mathtt{b}) \geq d(A, \mathtt{b}) \geq d(A, B) = d(\mathtt{a}, \mathtt{b})$, $d(A, \mathtt{b}) = d(\mathtt{a}, \mathtt{b})$. Therefore, $\mathtt{b}$ is such that $d(A, \mathtt{b}) = d(A, B)$ thus $\mathtt{b} \in A \dot{+}^d B$ and so, $A \dot{+}^d B \neq \emptyset$. So ($\dot{+}3$) is satisfied.

If $\mathcal{U} = \mathbb{R}^n$, if d is a L_1 distance on $\mathcal{U}$, and if each A representable in $\mathcal{L}$ is closed and bounded, then ($\mathcal{L}2$) is satisfied. More generally, if d is a distance in the classical mathematical sense (it verifies separation, symmetry, triangle inequality, and $d(\mathtt{a}, \mathtt{b}) < +\infty$ for every $\mathtt{a}, \mathtt{b} \in \mathcal{U}$), and if every A representable in $\mathcal{L}$ is a compact space, then ($\mathcal{L}2$) is satisfied.

($\dot{+}4$) and ($\dot{+}5$) are always satisfied by $\dot{+}^d$ as proven hereafter. The conjunction of these postulates is equivalent to (8). Let $A, B, C \in 2^{\mathcal{U}}$. If $(A \dot{+}^d B) \cap C = \emptyset$ then (8) is verified. Now, assume that $(A \dot{+}^d B) \cap C \neq \emptyset$ and let $\mathtt{b} \in (A \dot{+}^d B) \cap C$. Then $\mathtt{b} \in (B \cap C)$ and $d(A, \mathtt{b}) = d(A, B)$. Thus, the following chain of relations can be established:

$$d(A, B) \leq d(A, B \cap C) \leq d(A, \mathtt{b}) = d(A, B)$$

(cf. the infimum appearing in the definition of $d(A, \cdot)$ and the fact that $\mathtt{b} \in B \cap C$). Therefore, these numbers are all equal and $d(A, B) = d(A, B \cap C)$. Hence

$$\begin{aligned} A \dot{+}^d (B \cap C) &= \{\mathtt{b} \in B \cap C \mid d(A, \mathtt{b}) = d(A, B \cap C)\} \\ &= \{\mathtt{b} \in B \cap C \mid d(A, \mathtt{b}) = d(A, B)\} \\ &= \{\mathtt{b} \in B \mid d(A, \mathtt{b}) = d(A, B)\} \cap C \\ &= (A \dot{+}^d B) \cap C \end{aligned}$$

($\dot{+}6$) is not always satisfied by $\dot{+}^d$ as the following counterexample shows. Let $\mathcal{U} = \mathbb{R}^2$ and let $\mathcal{L}$ be the language of *horizontal rectangles*, i.e., for every $\varphi \in \mathcal{L}$, there exists $[x_1; x_2]$ and $[y_1; y_2]$, two intervals of $\mathbb{R}$, such that $\text{Mod}(\varphi) = [x_1; x_2] \times [y_1; y_2]$. Now, let d be the distance on $\mathcal{U}$ defined by $d(\mathtt{a}, \mathtt{b})$ is the Euclidian distance rounded up

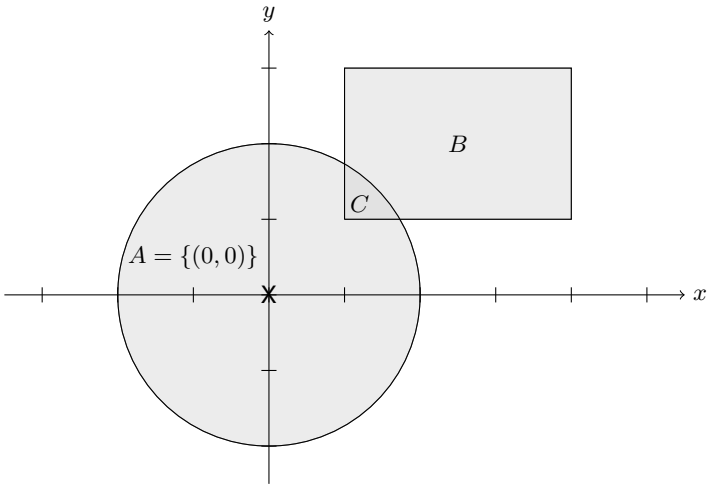


Fig. 3: A counterexample of ($\dot{+}6$). $A = [0;0] \times [0;0] = \{(0,0)\}$ and $B = [1;4] \times [1;3]$ are two subsets of $\mathcal{U}$ representable in the language $\mathcal{L}$ of horizontal rectangles, though $A \dot{+}^d B = C$ (where d is the Euclidian distance rounded up to integers) is not representable in $\mathcal{L}$. Indeed, C is the intersection of $G_2^d(A)$, the closed disk of radius 2 and centered on $(0,0)$, with B.

to integers ($d(\mathtt{a},\mathtt{b}) = \lceil d_2(\mathtt{a},\mathtt{b}) \rceil$ where d_2 is the canonical Euclidian distance on $\mathbb{R}^2$ and $\lceil x \rceil$ is the smallest integer n such that $x \leq n$). As illustrated by figure 3, $A \dot{+}^d B$ is a subset of $\mathcal{U}$ that is not representable in $\mathcal{L}$.

Now, let us consider the following assumption:

($\mathcal{L}3$) For each A, subset of $\mathcal{U}$ representable in $\mathcal{L}$, for each $\lambda \in \mathbb{R}$ with $\lambda \geq 0$, $G_\lambda^d(A)$ is representable in $\mathcal{L}$.

It is a sufficient condition for ($\dot{+}6$). Indeed, if A and B is representable in $\mathcal{L}$ then ($\mathcal{L}3$) entails that $G_\delta^d(A)$ is representable in $\mathcal{L}$, with $\delta = d(A,B)$. Therefore, $G_\delta^d(A) \cap B = A \dot{+}^d B$ is representable in $\mathcal{L}$, since this formalism is stable under conjunction. So ($\dot{+}6$) holds. Note that ($\mathcal{L}3$) is not a necessary condition for ($\dot{+}6$).

Table 1 summarizes these results.

Attribute-constraint formalisms

Definitions. In this section, it is assumed that $\mathcal{U} = V_1 \times V_2 \times \ldots \times V_n$ where each V_i is a "simple value" space, i.e. either $\mathbb{R}$ (the set of real numbers), $\mathbb{Z}$ (the set of integers), any interval of $\mathbb{R}$ or $\mathbb{Z}$, $\mathbb{B} = \{\mathtt{true}, \mathtt{false}\}$, or another set defined in extension. For $i \in \{1, 2, \ldots, n\}$, the attribute a_i is the i^{th} projection:

$$a_i : (\mathtt{a}_1, \mathtt{a}_2, \ldots, \mathtt{a}_n) \in \mathcal{U} \mapsto \mathtt{a}_i \in V_i$$

	($\dot{+}1$) is always satisfied.
Under assumption ($\mathcal{L}1$),	($\dot{+}2$) is satisfied.
Under assumption ($\mathcal{L}2$),	($\dot{+}3$) is satisfied.
	($\dot{+}4$) is always satisfied.
	($\dot{+}5$) is always satisfied.
Under assumption ($\mathcal{L}3$),	($\dot{+}6$) is satisfied.

Table 1: Conditions for having the revision postulates satisfied by a distance-based revision operator.

A formula φ of the representation language is a constraint, i.e., a Boolean expression based on the attributes a_i: $\varphi = P(a_1, a_2, \ldots, a_n)$. The semantics of φ is

$$\text{Mod}(\varphi) = \{\mathtt{a} \in \mathcal{U} \mid P(a_1(\mathtt{a}), a_2(\mathtt{a}), \ldots, a_n(\mathtt{a}))\}$$

These formalisms contain propositional logic with n variables: $V_i = \mathbb{B}$ (for each $i \in \{1, 2, \ldots, n\}$), knowing that the Boolean expressions are based on the Boolean operations `and`, `or`, and `not`. For example, if $n = 3$, and $a_1 = o$, $a_2 = t$ and $a_3 = v$:

$$\text{Mod}(\neg v \vee o) = \{(\mathtt{a}_1, \mathtt{a}_2, \mathtt{a}_3) \in \mathcal{U} \mid \mathtt{or}(\mathtt{not}(\mathtt{a}_3), \mathtt{a}_1)) = \mathtt{true}\}$$

These formalisms also contain the attribute-value formalisms often used for representing cases in CBR [10]: a specific case `C` is defined by $\mathtt{C} = (a_1 = v_1) \wedge (a_2 = v_2) \wedge \ldots \wedge (a_n = v_n)$ and thus $\mathtt{C} = \{(v_1, v_2, \ldots, v_n)\}$. When problem-solution decomposition is made, in general, the attributes are split in problem attributes (a_1, …, a_p) and solution attributes (a_{p+1}, …, a_n). Classically, the distance used on $\mathcal{U}$ for the retrieval is the weighted sum of distances on each problem attribute.

Application to the numerical case with linear constraints. Now, it is assumed that each V_i is either $\mathbb{R}$ or $\mathbb{Z}$ and each formula is a conjunction of linear constraints on the attributes. A linear constraint is an expression of the form $\sum_{i=1}^{n} \alpha_i \cdot a_i \leq \beta$ where $\alpha_1, \ldots, \alpha_n, \beta \in \mathbb{R}$.

Let d be the L_1 distance on $\mathcal{U}$ parametrized by a base $\mathcal{B}$. It can be shown that $\dot{+}^d$ satisfies all the ($\dot{+}$1-6) postulates (where A, B, and C are defined thanks to conjunctions of linear constraints). $\dot{+}^d$-adaptation amounts to solve the following optimization problem:

$$\mathtt{a} \in \mathtt{DK} \cap \mathtt{Source} \tag{9}$$

$$\mathtt{b} \in \mathtt{DK} \cap \mathtt{Target} \tag{10}$$

$$\text{minimize} \sum_{i=1}^{n} |v_i - u_i| \tag{11}$$

where $(u_1, \ldots, u_n)$ and $(v_1, \ldots, v_n)$ are the respective representations of `a` and `b` in the base $\mathcal{B}$. `ComplTarget` is the set of the `b` that solve this optimization problem.

In this optimization problem, (9) and (10) are linear constraints, but the function to be minimized in (11) is not linear. However, this optimization problem can be solved thanks to the solving of the following linear problem (introducing the new variables z_1, $\ldots$, z_n):

$$\begin{aligned}
&\mathtt{a} \in \mathtt{DK} \cap \mathtt{Source} \\
&\mathtt{b} \in \mathtt{DK} \cap \mathtt{Target} \\
&v_i - u_i \leq z_i && (1 \leq i \leq n) \\
&u_i - v_i \leq z_i && (1 \leq i \leq n) \\
&\text{minimize } \sum_{i=1}^{n} z_i
\end{aligned}$$

It can be shown that the optimal values of a and b in the two optimization problems are the same. Therefore, in this formalism, $\dot{+}^d$-adaptation amounts to a linear programming problem, which is NP-complete if some $V_i = \mathbb{Z}$ but is polynomial when all $V_i = \mathbb{R}$ [11].

More details about this process can be found in [12].

A cooking application. This principle has been applied to a CBR system called Taaable (`http://taaable.fr`) that has been a contestant of the CCC (*Computer Cooking Contest*, organized during the ICCBR conferences). The CCC provides a recipe base. A contestant of the CCC is a system that has to solve cooking problems using these recipes (a case of this application is a recipe). These problems are specified by a set of desired ingredients or dish types, and undesired ones (e.g., "I'd like a pear pie but I don't like cinnamon."). Taaable has won the main challenge and the adaptation challenge of this contest in 2010 [13]. The adaptation of ingredient quantities was made possible thanks to a reduction to linear programming as mentioned before. Details can be found in [13] but the idea, explained on a simplified example, is as follows. Suppose that the user wants a recipe of a pear pie and that Taaable retrieves an apple pie. The domain knowledge is expressed by linear constraints on these properties, such as:

$$\begin{aligned}
\mathtt{mass}_{\mathtt{fruit}} &= 120 \cdot \mathtt{nb}_{\mathtt{apple}} + 100 \cdot \mathtt{nb}_{\mathtt{pear}} \\
\mathtt{mass}_{\mathtt{sweet}} &= \mathtt{mass}_{\mathtt{sugar}} + 10 \cdot \mathtt{nb}_{\mathtt{apple}} + 15 \cdot \mathtt{nb}_{\mathtt{pear}}
\end{aligned}$$

(these knowledge can be found in a free nutritional database). Each $\mathtt{mass}_\bullet$ is an attribute on $\mathbb{R}$ and each $\mathtt{nb}_\bullet$ is an attribute on $\mathbb{N}$ (non negative integers). The source case is a singleton $\{\mathtt{a}\}$ such that $\mathtt{nb}_{\mathtt{apple}}(\mathtt{a}) = 4$ and $\mathtt{mass}_{\mathtt{sugar}}(\mathtt{a}) = 40$. The target case corresponds to the constraint $\mathtt{nb}_{\mathtt{apple}} = 0$ (the substitution of apples by pears is inferred by a previous step similar to a $\dot{+}$-adaptation in propositional logic). The $\dot{+}^d$-adaptation leads to a maximal preservation of the attributes $\mathtt{mass}_{\mathtt{fruit}}$ and $\mathtt{mass}_{\mathtt{sugar}}$ and since the pears contain more sweet than the apples, the mass of added sugar is lowered (there is a sort of "compensation effect"). More precisely, the $\dot{+}^d$-adaptation (at least for some base $\mathcal{B}$) gives $\mathtt{ComplTarget} = \{\mathtt{b}\}$ with $\mathtt{nb}_{\mathtt{pear}}(\mathtt{b}) = 5$ (the total fruit mass from Source to ComplTarget is modified from 480 to 500) and $\mathtt{mass}_{\mathtt{sugar}}(\mathtt{b}) = 5$ (the total sweet mass is unchanged).

4.5 Multiple case adaptation

Some CBR systems retrieve several cases and then adapt them in order to solve the target case:

$$\texttt{Retrieval}: (\texttt{CaseBase}, \texttt{Target}) \mapsto \{\texttt{Source}_i\}_{1 \leq i \leq k} \subseteq \texttt{CaseBase}$$

$$\texttt{Adaptation}: \left(\{\texttt{Source}_i\}_{1 \leq i \leq k}, \texttt{Target}\right) \mapsto \texttt{ComplTarget}$$

This adaptation is called multiple case adaptation and is also known as case combination. Multiple case adaptation extends single case adaptation (which is a case combination with $k = 1$) in the same way as integrity constraint belief merging extends belief revision (cf. section 3.3), hence the idea[6] to use a merging operator $\triangle$ on $\mathcal{U}$ to define a multiple case adaptation process:

$$\texttt{ComplTarget} = \triangle_{\texttt{DK} \cap \texttt{Target}} \left(\{\texttt{DK} \cap \texttt{Source}_i\}_{1 \leq i \leq k}\right)$$

which generalizes (7).

This approach to multiple case adaptation is studied in [12].

4.6 Revision-based adaptation and other approaches to adaptation

Other approaches to adaptation have been defined in the CBR literature. This section compares revision-based adaptation to some of them.

Adaptation by generalization and specialization

The principle of this adaptation is to generalize the source case and then to specialize it to the target case. Since

- the source and target cases must be considered w.r.t. the domain knowledge and
- this generalization of the source case must be minimal so that it meets the constraints of the target case,

this kind of adaptation amounts to generalize minimally $A = \texttt{DK} \cap \texttt{Source}$ into A' such that $A' \cap B \neq \emptyset$ with $B = \texttt{DK} \cap \texttt{Target}$. Therefore, it is a $\dot{+}$-adaptation for which the modification operation $A \mapsto A'$ is a generalization ($A \subseteq A'$).

Conversely, if $\dot{+}$ is a revision operator and $A, B \in 2^{\mathcal{U}}$, $A \dot{+} B = A' \cap B$ such that A has been modified minimally in A'. This modification is not necessarily a generalization but if $\widehat{A} = A \cup A'$, then $A \subseteq \widehat{A}$ and $A \dot{+} B = \widehat{A} \cap B$. Indeed, either $A \cap B \neq \emptyset$ so $A' = A$ and then $\widehat{A} = A$, or $A \cap B = \emptyset$ so $\widehat{A} \cap B = (A \cup A') \cap B = (A \cap B) \cup (A' \cap B) = \emptyset \cup (A' \cap B) = A \dot{+} B$. So, without loss of generality, it can be considered that the modification of A done when revising A by any revision operator on $\mathcal{U}$ is a generalization. Hence, $\dot{+}$-adaptation can be considered as a formalization of the general model of adaptation by generalization and specialization.

When the revision operator is based on a distance d, $\dot{+}^d$-adaptation can be read as the composition of

[6] Once suggested by Pierre Marquis. Thanks Pierre!

a generalization $\texttt{DK} \cap \texttt{Source} \mapsto G^d_\delta(\texttt{DK} \cap \texttt{Source})$ and
a specialization $G^d_\delta(\texttt{DK} \cap \texttt{Source}) \mapsto G^d_\delta(\texttt{DK} \cap \texttt{Source}) \cap \texttt{DK} \cap \texttt{Target}$.

To illustrate this idea, it can be noticed that the example of section 4.2 (when Léon invites Thècle) of revision-based adaptation in propositional logic, can be redescribed as an adaptation by generalization and specialization. First, the meal is generalized by substituting beef by protein food (generalization motivated by the fact that Thècle is vegetarian). Then, the generalized meal is specialized by substituting protein food by tofu.

For the other approaches to adaptation considered below, the assumptions of problem-solving decomposition ($\mathcal{U} = \mathcal{U}_{\texttt{pb}} \times \mathcal{U}_{\texttt{sol}}$, $\texttt{Source} = \texttt{srce} \times \texttt{Sol}(\texttt{srce})$, $\texttt{Target} = \texttt{tgt} \times \mathcal{U}_{\texttt{sol}}$, $\texttt{ComplTarget} = \texttt{tgt} \times \texttt{Sol}(\texttt{tgt})$) and of specificities of the source case and the target problem ($\texttt{Source} = \{\texttt{a}\} = \{(x^s, y^s)\}$, $\texttt{tgt} = \{x^t\}$) are made.

Null adaptation, orthogonal revision operators and conservative adaptation

In [1], null adaptation is justified by the assertion "People often do little adaptation." This adaptation is defined by $\texttt{Sol}(\texttt{tgt}) = \texttt{Sol}(\texttt{srce})$. It is often used when the solution space representation is very simple (e.g., a set of predefined categories). It is also used when CBR is assimilated to a simple approximate reasoning method: if $\texttt{Sol}(\texttt{srce})$ solves $\texttt{srce}$ and $\texttt{srce}$ is similar to $\texttt{tgt}$ then it is likely that $\texttt{Sol}(\texttt{tgt}) = \texttt{Sol}(\texttt{srce})$ approximately solves $\texttt{tgt}$. Under the following assumptions, $\dot{+}$-adaptation coincides with null adaptation:

(A1) No domain knowledge is considered (i.e., $\texttt{DK} = \mathcal{U}$);
(A2) $\dot{+}$ is *orthogonal*,[7] i.e. there exists two revision operators, $\dot{+}_{\texttt{pb}}$ on $\mathcal{U}_{\texttt{pb}}$ and $\dot{+}_{\texttt{sol}}$ on $\mathcal{U}_{\texttt{sol}}$, such that

$$(A_{\texttt{pb}} \times A_{\texttt{sol}}) \dot{+} (B_{\texttt{pb}} \times B_{\texttt{sol}}) = (A_{\texttt{pb}} \dot{+}_{\texttt{pb}} B_{\texttt{pb}}) \times (A_{\texttt{sol}} \dot{+}_{\texttt{sol}} B_{\texttt{sol}}) \quad (12)$$

for any $A_{\texttt{pb}}, B_{\texttt{pb}} \in 2^{\mathcal{U}_{\texttt{pb}}}$ and $A_{\texttt{sol}}, B_{\texttt{sol}} \in 2^{\mathcal{U}_{\texttt{sol}}}$.

Indeed, under these assumptions:

$$\begin{aligned}\texttt{ComplTarget} &= (\texttt{srce} \times \texttt{Sol}(\texttt{srce})) \dot{+} (\texttt{tgt} \times \mathcal{U}_{\texttt{sol}}) \\ &= (\texttt{srce} \dot{+}_{\texttt{pb}} \texttt{tgt}) \times (\texttt{Sol}(\texttt{srce}) \dot{+}_{\texttt{sol}} \mathcal{U}_{\texttt{sol}})\end{aligned}$$

Therefore:

$$\begin{aligned}\texttt{Sol}(\texttt{tgt}) &= \texttt{Sol}(\texttt{srce}) \dot{+}_{\texttt{sol}} \mathcal{U}_{\texttt{sol}} \\ &= \texttt{Sol}(\texttt{srce}) \cap \mathcal{U}_{\texttt{sol}} \qquad \text{(according to postulate } (\dot{+}2)) \\ &= \texttt{Sol}(\texttt{srce})\end{aligned}$$

[7] This adjective is justified further in the chapter.

The intuition behind this notion of orthogonality of $\dot{+}$ is that the modifications are made independently in the problem space and in the solution space. For example, if d is a distance on $\mathcal{U}$ such that there exists a distance $d_{\texttt{pb}}$ on $\mathcal{U}_{\texttt{pb}}$ and a distance $d_{\texttt{sol}}$ on $\mathcal{U}_{\texttt{sol}}$ with

$$d((x^1, y^1), (x^2, y^2)) = d_{\texttt{pb}}(x^1, x^2) + d_{\texttt{sol}}(y^1, y^2) \tag{13}$$

for each $(x^1, y^1), (x^2, y^2) \in \mathcal{U}$, then $\dot{+}^d$ is orthogonal. Note that the reverse is not true. For example, if $d((x^1, y^1), (x^2, y^2)) = \sqrt{d_{\texttt{pb}}(x^1, x^2)^2 + d_{\texttt{sol}}(y^1, y^2)^2}$ then $\dot{+}^d$ is orthogonal, though d cannot be written as in (13) in general.

Now, let us consider the situation when the assumption (A2) is made but the assumption (A1) is not. $\texttt{Source}$ is a singleton $\{(x^s, x^t)\}$ consistent with $\texttt{DK}$, thus $\texttt{DK} \cap \texttt{Source} = \{(x^s, x^t)\}$. $\texttt{Target} = \{x^t\} \times \mathcal{U}_{\texttt{sol}}$ so:

$$\begin{aligned} \texttt{DK} \cap \texttt{Target} &= \{(x, y) \in \texttt{DK} \mid x = x^t\} = \{x^t\} \times \texttt{DK}_{\texttt{sol}}(x^t) \\ \text{where } \texttt{DK}_{\texttt{sol}}(x^t) &= \{y \in \mathcal{U}_{\texttt{sol}} \mid (x^t, y) \in \texttt{DK}\} \end{aligned}$$

Therefore:

$$\begin{aligned} \texttt{ComplTarget} &= \{(x^s, y^s)\} \dot{+} (\{x^t\} \times \texttt{DK}_{\texttt{sol}}(x^t)) \\ &= (\{x^s\} \dot{+}_{\texttt{pb}} \{x^t\}) \times (\{y^s\} \dot{+}_{\texttt{sol}} \texttt{DK}_{\texttt{sol}}(x^t)) \qquad \text{(cf. (12))} \\ &= \{x^t\} \times (\{y^s\} \dot{+}_{\texttt{sol}} \texttt{DK}_{\texttt{sol}}(x^t)) \end{aligned}$$

($\{x^s\} \dot{+}_{\texttt{pb}} \{x^t\} = \{x^t\}$ is a consequence of ($\dot{+}1$) and ($\dot{+}3$)). So, the $\dot{+}$-adaptation consists in "repairing" the solution $\texttt{Sol}(\texttt{srce}) = \{y^s\}$ according to the constraint $\texttt{DK}_{\texttt{sol}}(x^t)$, the repair being done by $\dot{+}_{\texttt{sol}}$. Note that, it this process, neither $\dot{+}_{\texttt{pb}}$ nor x^s are used.

In the following, a $\dot{+}$-adaptation with an orthogonal revision operator $\dot{+}$ is called a *conservative adaptation*.[8]

Rule-based adaptation

Rule-based adaptation is the adaptation based on a set of *adaptation rules*. Following the formalization of [14], an adaptation rule is an ordered pair $(\texttt{r}, \mathcal{A}_{\texttt{r}})$ where $\texttt{r}$ is a binary relation on $\mathcal{U}_{\texttt{pb}}$ and $\mathcal{A}_{\texttt{r}}$ is such that, for $x^s, x^t \in \mathcal{U}_{\texttt{pb}}$ and $y^s \in \mathcal{U}_{\texttt{sol}}$ ($\texttt{Source} = \{(x^s, y^s)\}$, $\texttt{Target} = \{x^t\} \times \mathcal{U}_{\texttt{sol}}$):

$$\textbf{if } x^s \ \texttt{r} \ x^t \textbf{ then } \mathcal{A}_{\texttt{r}}(x^s, y^s, x^t) = y^t \textit{ probably} \text{ solves } x^t$$

The rule is not certain (hence the "probably").

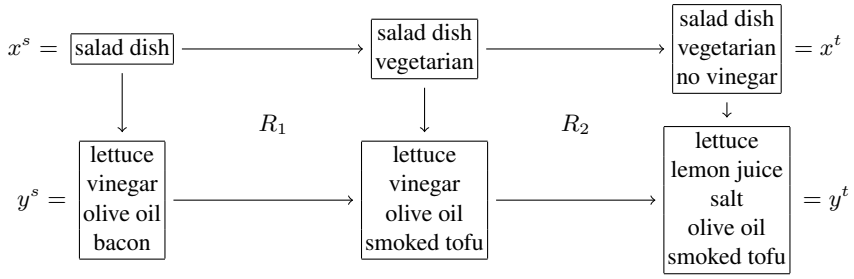
The adaptation rules can be composed as explained hereafter. Let $\texttt{AK}$ be the finite set of adaptation rules that are available to the CBR system. Let $\texttt{AK}_{\texttt{pb}} = \{\texttt{r} \mid (\texttt{r}, \mathcal{A}_{\texttt{r}}) \in \texttt{AK}\}$.

[8] In the first publications about $\dot{+}$-adaptation, the term "conservative adaptation" was used for any $\dot{+}$-adaptation. Then, it has appeared that some $\dot{+}$-adaptation could hardly be qualified as conservative (see below), thus, from now on, the term of conservative adaptation is only used for revision-based adaptation using an orthogonal revision operator.

$R_1 =$ Bacon can be sustituted by smoked tofu

$R_2 =$ In a salad dish, vinegar can be sustituted by lemon juice and salt

(a) Examples of rules.



(b) An example of rule-based adaptation.

$$\begin{aligned}
\texttt{DK} &= \text{lettuce} \Rightarrow \text{green salad} \quad \wedge \quad \text{escarole} \Rightarrow \text{green salad} \\
&\wedge \quad \text{vegetarian} \Rightarrow \neg\text{meat} \quad \wedge \quad \text{bacon} \Rightarrow \text{meat} \\
\texttt{Source} &= \text{salad dish} \wedge \text{lettuce} \wedge \text{vinegar} \wedge \text{olive oil} \wedge \text{bacon} \\
\texttt{Target} &= \text{salad dish} \wedge \neg\text{vinegar} \wedge \text{vegetarian} \\
\texttt{AK} &= \{R_1, R_2\} \\
\text{with } R_1 &= \text{bacon} \rightsquigarrow \text{smoked tofu} \\
\text{and } R_2 &= \text{salad dish} \wedge \text{vinegar} \rightsquigarrow \text{salad dish} \wedge \text{lemon juice} \wedge \text{salt} \\
\texttt{ComplTarget} &\equiv \texttt{DK} \wedge \texttt{Target} \wedge \text{lettuce} \wedge \text{lemon juice} \wedge \text{salt} \wedge \text{olive oil} \wedge \text{smoked tofu}
\end{aligned}$$

(c) Formalization of this example as a $\dot{+}^{d_{\texttt{AK}}}$-adaptation.

$$\texttt{Target}_2 = \texttt{Target} \wedge \neg\text{lettuce}$$

(d) A new target case, slightly different from `Target` but which cannot be solved by rule-based adaptation of `Source` given `DK` and `AK` (i.e., by $\dot{+}^{d_{\texttt{AK}}}$-adaptation).

$$\begin{aligned}
\texttt{ComplTarget}_2 &= (\texttt{DK} \wedge \texttt{Source}) \dot{+}^{d} (\texttt{DK} \wedge \texttt{Target}_2) \\
&\equiv \texttt{DK} \wedge \texttt{Target}_2 \wedge \text{green salad} \wedge \text{lemon juice} \wedge \text{salt} \wedge \text{olive oil} \wedge \text{smoked tofu}
\end{aligned}$$

(e) Solving $\texttt{Target}_2$ by $\dot{+}^{d}$-adaptation, where d is defined by equation (14), with d_0 the Hamming distance. This adaptation consists in applying R_1, R_2 and in removing lettuce. $\texttt{ComplTarget}_2 \models$ green salad $\wedge \neg$lettuce: the lettuce can be replaced by any other kind of green salad, e.g., escarole.

Fig. 4: Rule-based adaptation and revision-based adaptation on an example.

$\mathtt{AK}_{\mathtt{pb}}$ provides a structure on $\mathcal{U}_{\mathtt{pb}}$. A similarity path from $x^s \in \mathcal{U}_{\mathtt{pb}}$ to $x^t \in \mathcal{U}_{\mathtt{pb}}$ is a path in $(\mathcal{U}_{\mathtt{pb}}, \mathtt{AK}_{\mathtt{pb}})$: it is a sequence of relations $\mathtt{r}^i \in \mathtt{AK}_{\mathtt{pb}}$ such that there exist $x^0, x^1, \ldots, x^q \in \mathcal{U}_{\mathtt{pb}}$ with $x^0 = x^s$, $x^q = x^t$, and $x^{i-1}\ \mathtt{r}^i\ x^i$ $(1 \leq i \leq q)$. Given such a similarity path, $y^t \in \mathcal{U}$ that probably solves x^t can be computed by applying successively the rules $(\mathtt{r}^1, \mathcal{A}_{\mathtt{r}^1})$, …, $(\mathtt{r}^q, \mathcal{A}_{\mathtt{r}^q})$: $y^i = \mathcal{A}_{\mathtt{r}^i}(x^{i-1}, y^{i-1}, x^i)$ for i taking the successive values 1, 2, …, q. Finally, $y^t = y^q$ probably solves x^t. This can be graphically represented by the following diagram, composed of q diagrams like the one of (2), section 2.2:

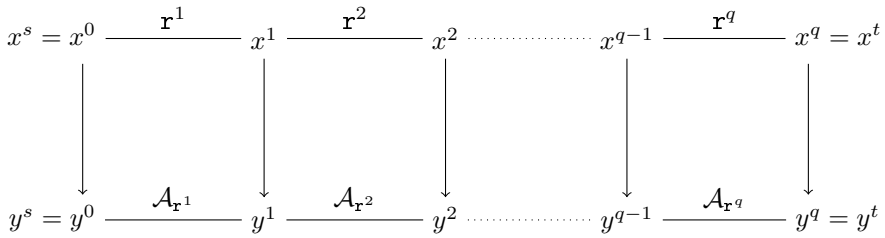


Figure 4(b) illustrates such a rule-based adaptation with a similarity path in two steps, using rules of figure 4(a).

There may be several similarity paths from x^s to x^t. The choice between them is usually based on a cost function such that if SP_1 and SP_2 are two similarity paths from x^s to x^t and $\mathtt{cost}(SP_1) < \mathtt{cost}(SP_2)$ then SP_1 is preferred to SP_2, which is interpreted as "SP_1 is more likely to lead to an appropriate solution to x^t than SP_2." The function cost is usually assumed to be additive, that is $\mathtt{cost}(SP)$ is the sum of $\mathtt{cost}(\mathtt{r})$ for r a relation of SP. To each $(\mathtt{r}, \mathcal{A}_{\mathtt{r}}) \in \mathtt{AK}$, $\mathtt{cost}(\mathtt{r}) > 0$ is an information associated with this adaptation rule.[9]

Let $d_{\mathtt{AK}}$ be the distance on $\mathcal{U}$ defined by

$$d_{\mathtt{AK}}((x^s, y^s), (x^t, y^t)) = \min \left\{ \mathtt{cost}(SP) \;\middle|\; \begin{array}{l} SP\text{: similarity path from } x^s \text{ to } x^t \\ \text{such that the application of } SP \\ \text{on } \{(x^s, y^s)\} \text{ gives } y^t \end{array} \right\}$$

with the convention $\min \emptyset = +\infty$. Let $\mathtt{ComplTarget} = \mathtt{tgt} \times \mathtt{Sol}(\mathtt{tgt})$ be the result of $\dot{+}^{d_{\mathtt{AK}}}$-adaptation without domain knowledge ($\mathcal{U} = \mathtt{DK}$). If there is no similarity path from x^s to x^t, then $\mathtt{ComplTarget} = \mathtt{Target}$ (the adaptation fails: it does not add any information to the target case). Else, $\mathtt{b} = (x^t, y^t) \in \mathtt{ComplTarget}$ iff y^t is obtained by application of a similarity path of minimal cost. Therefore, revision-based adaptation includes rule-based adaptation. Moreover, DK can be taken into account in $\dot{+}^{d_{\mathtt{AK}}}$-adaptation, thus this enables to specify a rule-based adaptation taking into account the domain knowledge. Conversely, if some adaptation knowledge AK in the form of rules has been acquired (e.g., by means of knowledge discovery and data-mining techniques [15, 16]), this can be useful to specify a relevant revision operator. Indeed, there

[9] A coarse modeling of this cost is $\mathtt{cost}(\mathtt{r}) = -\log P$ where P is the probability that y^t is a licit solution of x^s. Thus, the additivity of the cost corresponds to an independence assumption of the q adaptation steps.

are many possible revision operators and the adaptation knowledge enables to make some choices among them.

Figure 4(c) illustrates how the rule-based adaptation of figure 4(b) can be formalized by a revision-based adaptation using $\dot{+}^{d_{\mathtt{AK}}}$.

A limitation of rule-based reasoning is that it can fail, meaning that there is no similarity path from x^s to x^t ($d(x^s, x^t) = +\infty$), which involves that $\mathtt{ComplTarget} = \mathtt{Target}$. This is particularly true when there are few adaptation rules. Indeed, if $\mathtt{AK} \subseteq \mathtt{AK}'$ then $d_{\mathtt{AK}}(\mathtt{a}, \mathtt{b}) \geq d_{\mathtt{AK}'}(\mathtt{a}, \mathtt{b})$ so if $\dot{+}^{d_{\mathtt{AK}'}}$-adaptation fails then $\dot{+}^{d_{\mathtt{AK}}}$-adaptation fails. One way to overcome this limitation is to combine this kind of adaptation with another approach to adaptation and the principle of revision-based adaptation can be used to formalize this combination. This idea is formalized as follows. Let us assume that the other approach to adaptation that has to be combined with rule-based adaptation can be formalized as a revision-based adaptation and let d_0 be a distance on $\mathcal{U}$ such that this adaptation coincides with the $\dot{+}^{d_0}$-adaptation (intuitively, this adaptation is a "novice" adaptation, hence the 0 in d_0). The $\dot{+}^{d}$-adaptation with d defined below combines rule-based adaptation with $\dot{+}^{d_0}$-adaptation (for $\mathtt{a}, \mathtt{b} \in \mathcal{U}$):

$$d(\mathtt{a}, \mathtt{b}) = \inf_{\mathtt{c} \in \mathcal{U}} \left(W_{\mathtt{AK}} \cdot d_{\mathtt{AK}}(\mathtt{a}, \mathtt{c}) + W_0 \cdot d_0(\mathtt{c}, \mathtt{b}) \right) \tag{14}$$

where $W_{\mathtt{AK}}$ and W_0 are two positive constants. When $\mathtt{AK} = \emptyset$, $d = d_0$ (intuitively: with no adaptation knowledge, the adaptation process is a novice). If the infimum above is reached on $\mathtt{c} = (x^c, y^c)$, the $\dot{+}^{d}$-adaptation consists in a rule-based adaptation of $\mathtt{Source} = \{\mathtt{a}\}$ to solve $\{x^c\}$ and then a $\dot{+}^{d_0}$-adaptation of $\mathtt{c}$ to solve the target case.

Figure 4(d) presents an example of adaptation that cannot be solved by rule-based adaptation using the two rules of figure 4(a) but can be solved by $\dot{+}^{d}$-adaptation according to equation (14).

Differential calculus as a $\dot{+}$-adaptation

First-order differential calculus rely on the equation

$$dy_j = \sum_i \frac{\partial y_j}{\partial x_i} dx_i$$

where $x \mapsto y$ is a differentiable function from $\mathbb{R}^p$ to $\mathbb{R}^q$. dx_i and dy_j can be interpreted as small differences and substituted respectively by $x_i^t - x_i^s$ and $y_j^t - y_j^s$ when x^s and x^t are similar and thus can be seen as a way to adapt $\mathtt{Source} = \{(x^s, y^s)\}$ in order to solve $\mathtt{Target} = \{x^t\} \times \mathcal{U}_{\mathtt{sol}}$. In this adaptation process, $\left\{ \frac{\partial y_j}{\partial x_i} \right\}_{ij}$ constitutes the adaptation knowledge: it represents how a solution descriptor is modified when a problem descriptor is modified.

In the following, we restrict ourselves to a differentiable function $f : \mathbb{R} \to \mathbb{R}$, though the principle presented below can be generalized to $f : \mathbb{R}^p \to \mathbb{R}^q$. Thus $\mathcal{U}_{\mathtt{pb}} = \mathcal{U}_{\mathtt{sol}} = \mathbb{R}$.

A first-order approximation approximation of f in x^t is

$$y^t = x^s + \varrho^s (x^t - x^s) \tag{15}$$

The best choice for ϱ^s is $\varrho^s = f'(x^s)$ (where f' is the differential of f): this choice is the best in the sense that $f'(x^s)$ is the only ϱ^s such that

$$y^t - (y^s + \varrho^s(x^t - x^s)) \underset{x^t \longrightarrow x^s}{=} o(x^t - x^s)$$

(this is a direct consequence of the definition of differentials). The adaptation knowledge of this first-order approximation is ϱ^s (either defined for each source case or more globally).

The adaptation defined in (15) coincides with the $\dot{+}^d$-adaptation with $\mathtt{DK} = \mathcal{U}$ and d defined by

$$d((x^s, y^s), (x^t, y^t)) = |x^t - x^s| + |y^t - y^s - \varrho^s(x^t - x^s)| \tag{16}$$

(the value of ϱ^s when $\{(x^s, y^s)\}$ is not a source case does not matter here so it can be chosen arbitrarily).

From this, conclusions similar to the conclusions about rule-based adaptation can be drawn. First, $\dot{+}$-adaptation makes it possible to incorporate domain knowledge in this first-order approximation. Then, the knowledge of the adaptation knowledge (ϱ^s) can be used to parametrize the revision operator.

Let $(\overrightarrow{\varepsilon_1}, \overrightarrow{\varepsilon_2})$ be the canonical base of $\mathbb{R}^2$: $\overrightarrow{\varepsilon_1} = (1, 0)$, $\overrightarrow{\varepsilon_2} = (0, 1)$. Let $\{(x^s, y^s)\}$ be a fixed source case. The distance defined by (16) is the L_1 distance parametrized by the base $(\overrightarrow{e_1}, \overrightarrow{e_2})$ with $\overrightarrow{e_1} = \overrightarrow{\varepsilon_1} + \varrho^s\overrightarrow{\varepsilon_2}$ and $\overrightarrow{e_2} = \overrightarrow{\varepsilon_2}$ (ϱ^s is a constant here, since $\{(x^s, y^s)\}$ is fixed). The base $(\overrightarrow{e_1}, \overrightarrow{e_2})$ is orthogonal w.r.t. $(\overrightarrow{\varepsilon_1}, \overrightarrow{\varepsilon_2})$ iff $\varrho^s = 0$. Now, when $\varrho^s = 0$, d can be written as in (13) thus $\dot{+}^d$ is an orthogonal revision operator (which justifies a posteriori the adjective "orthogonal": cf. note 7). So, if the base $\mathcal{B}$ is orthogonal, $\dot{+}^d$-adaptation is a conservative adaptation.

Other approaches to adaptation

There are other adaptation approaches described in the literature. Most of them are domain-specific and defined by their algorithms. However, there is at least one other general approach to adaptation: the case-based adaptation [17, 18] (also known as *recursive* CBR [19]). The idea is that the adaptation of a CBR system can be a CBR system itself where cases are *adaptation cases*. This approach to adaptation still remains to be formalized in order to be compared to revision-based adaptation.

4.7 Other studies related to revision-based adaptation

Some other studies related to revision-based adaptation have been carried out. Two of them are explained below.

Adaptation in the description logic $\mathcal{ALC}$

Description logics (DLs) form a family of formalisms that are equivalent to fragments of first-order logic [20].[10] The most used semantics of DLs is based on the theory of models, as this is the case for, e.g., propositional logic. However, given the collection $\mathcal{U}$ of all the interpretations of a DL, it is uneasy (if not impossible) to define a distance on $\mathcal{U}$. Therefore, the authors have not considered any more the idea of defining a $\dot{+}^d$ revision operator for a DL and have searched in another direction.

In [21], the existence of a belief contraction operator satisfying the postulates of contraction is studied for various DLs (belief contraction is an operator on belief bases that has been related to belief revision through the so-called Harper and Levi indentities). There are also some research and implementations on the related issue of repairing inconsistent DL knowledge bases [22].

In [23], an approach to adaptation in the DL $\mathcal{ALC}$ (the simplest propositionally closed DL) has been proposed that is inspired by the principle of revision-based adaptation (though it has not defined a revision operator dealing with any knowledge bases of this DL). The principle of this adaptation in $\mathcal{ALC}$ is based on the semantic tableau method and on inconsistency repairing. The semantic tableau method is used to check the consistency of a DL knowledge base. Roughly said, it consists in applying a set of deductive rules whenever it is possible and then, to detect the clashes (a clash is a contradiction of an easy to detect predefined form). If there is a clash in each reasoning branch, then the knowledge base is inconsistent, otherwise it is consistent (provided that the set of rules have some completeness property). The adaptation in $\mathcal{ALC}$ consists in (1) "pretending" that the source case solves the target problem, (2) applying the semantic tableau method which leads to clashes, and (3) applying a clash repairing strategy that minimizes a repair cost.

The following example illustrates this approach to adaptation in $\mathcal{ALC}$, using the first-order logic syntax. This example is in the cooking domain and cases represent recipes. The domain knowledge is expressed by the following formula:

$$\begin{aligned}\mathtt{DK} = {} & \forall x\ \mathtt{gratedRawCarrot}(x) \Leftrightarrow \mathtt{carrot}(x) \wedge \mathtt{raw}(x) \wedge \mathtt{grated}(x) \\ & \wedge \forall x\ \mathtt{root}(x) \Leftrightarrow \mathtt{carrot}(x) \vee \mathtt{parsnip}(x) \vee \mathtt{celery}(x) \\ & \wedge \forall x\ \mathtt{parsnip}(x) \Rightarrow \neg \mathtt{raw}(x)\end{aligned}$$

(the first line expresses what grated raw carrots are, the second line means that the only roots that are considered in this example are carrots, parsnips and celeries, the third line means that a parsnip in a recipe must not be raw). The source case is represented by a formula about a constant σ representing a carrot salad, which is a starter with grated raw carrot and vinaigrette as ingredients:

$$\begin{aligned}\mathtt{Source} = {} & \mathtt{starter}(\sigma) \wedge \exists x\ \mathtt{ingredient}(\sigma, x) \wedge \mathtt{gratedRawCarrot}(x) \\ & \wedge \exists x\ \mathtt{ingredient}(\sigma, x) \wedge \mathtt{vinaigrette}(x)\end{aligned}$$

The target case expresses the fact that the user wants the recipe of a starter without carrot in it, by the following formula about the constant θ:

$$\mathtt{Target} = \mathtt{starter}(\theta) \wedge \neg(\exists x\ \mathtt{ingredient}(\theta, x) \wedge \mathtt{carrot}(x))$$

[10] Some DLs use constructs that go beyond standard first-order logic, such as concrete domains, but this is not the case for $\mathcal{ALC}$.

The three steps of the adaptation are as follows. (1) To pretend that the source case solves the target case amounts to identify σ and θ ($\sigma = \theta$). (2) $\texttt{DK} \wedge \texttt{Source} \wedge \texttt{Target} \wedge (\sigma = \theta)$ is inconsistent; the tableaux method leads to the clash

$$\text{between } \exists x, \texttt{ingredient}(\sigma, x) \wedge \texttt{carrot}(x) \tag{17}$$

$$\text{and } \neg\exists x, \texttt{ingredient}(\theta, x) \wedge \texttt{carrot}(x) \tag{18}$$

(3) In order to repair this clash, the piece of knowledge of equation (17) is removed but its consequence (given DK)

$$\exists x, \texttt{ingredient}(\theta, x) \wedge \texttt{root}(x) \wedge \texttt{raw}(x) \wedge \texttt{grated}(x) \tag{19}$$

is kept. Thus, from (18), (19) and the domain knowledge, it can be deduced that

$$\exists x, \texttt{ingredient}(\theta, x) \wedge \texttt{parsnip}(x) \wedge \texttt{raw}(x) \wedge \texttt{grated}(x) \tag{20}$$

$$\text{or } \exists x, \texttt{ingredient}(\theta, x) \wedge \texttt{celery}(x) \wedge \texttt{raw}(x) \wedge \texttt{grated}(x) \tag{21}$$

Now (20) leads to another clash, since the parsnip must not be raw. By contrast, the celery can be eaten raw and does not lead to another clash. Therefore, the piece of knowledge (20) is disgarded whereas (21) is kept and, finally:

$$\begin{aligned}\texttt{ComplTarget} \equiv\ & \texttt{DK} \wedge \texttt{starter}(\theta) \\ & \wedge \exists x\ \texttt{ingredient}(\sigma, x) \wedge \texttt{celery}(x) \wedge \texttt{raw}(x) \wedge \texttt{grated}(x) \\ & \wedge \exists x\ \texttt{ingredient}(\sigma, x) \wedge \texttt{vinaigrette}(x)\end{aligned}$$

Adaptation of qualitative constraint networks

Qualitative algebras (QAs) constitute a family of knowledge representation formalisms. For example, Allen algebra is a well-known QA dedicated to temporal representation [24] and RCC8 is a well-known QA dedicated to spatial representation [25]. A qualitative constraint network (QCN) is a knowledge base of a QA; it is given by a set of constraints between variables. Belief revision has been studied for QAs in [26]: a revision operator $\dot{+}$ associating to a QCN ψ and a QCN μ a QCN $\psi \dot{+} \mu$ is defined, following principles of distance-based revision operators.[11]

Therefore, revision-based adaptation can be applied to a CBR system in which cases are represented by QCNs. This has been studied in [27]. In a first application, revision-based adaptation has been applied to adapt the preparation parts of cooking recipes, for the system Taaable. The QA used was INDU [28], an extension of the Allen algebra. In a second application, revision-based adaptation has been applied to adapt a crop allocation in a farmland. The QA used was RCC8. This work has raised some implementation issues that are addressed in the paper.

[11] $\mathcal{U}$ corresponds to the set of scenarios, i.e., fully constrained QCNs. Mod(φ) is the set of the scenarios that are more specific than the QCN φ. A distance d between scenarios is defined. One problem is that if $A = \text{Mod}(\psi)$ and $B = \text{Mod}(\mu)$, $A \dot{+}^d B$ is not necessarily representable by a QCN (i.e., postulate ($\dot{+}$6) does not hold). In this case, to write it in a simplified manner, a most specific QCN χ is chosen such that $\text{Mod}(\chi) \supseteq A \dot{+}^d B$, and $\psi \dot{+} \mu = \chi$.

The figure 5 illustrates revision-based adaptation in the Allen algebra (the figure represents the different pieces of knowledge in this formalism—using the qualitative constraint network notations for the most part—and the captions describe them in natural language, for reader non familiar with the Allen algebra). The story underlying this example is an attribution of schedule to teachers. The source case corresponds to the previous year. The target case corresponds to the fact that, for some reason, the French teacher and the Math teacher want to avoid each other. The distance used in this example is based on the distance between the Allen basic relations which are the distances in the relation neighbourhood graph of this formalism (see [29]).

5 Revision-based CBR

Let $\texttt{SOURCE}$ be the union of all the cases from the case base:

$$\texttt{SOURCE} = \bigcup_i \texttt{Source}_i \text{ where } \texttt{CaseBase} = \{\texttt{Source}_i\}_i$$

The following question can be raised: according to what conditions can the CBR process with $\texttt{SOURCE}$ as only source case be equivalent to the CBR process with $\texttt{CaseBase}$? This question is addressed below with a $\dot{+}^d$-adaptation.

Let A_1, A_2, …, A_n, and B be $n+1$ subsets of $\mathcal{U}$. Let $\delta_i = d(A_i, B)$ and $\Delta = \min_i \delta_i$. The following equation holds:

$$\left(\bigcup_i A_i\right) \dot{+}^d B = \bigcup_{i,\delta_i=\Delta} (A_i \dot{+}^d B)$$

Indeed $d(\bigcup_i A_i, \texttt{b}) = \min_i d(A_i, \texttt{b})$ for any $\texttt{b} \in B$, and so $(\bigcup_i A_i) \dot{+} B = \{\texttt{b} \in \mathcal{U} \mid \min_i d(A_i, b) = \Delta\} = \bigcup_{i,\delta_i=\Delta}(A_i \dot{+} B)$.

From this equation applied to $A_i = \texttt{DK} \cap \texttt{Source}_i$ and $B = \texttt{DK} \cap \texttt{Target}$, it comes that the $\dot{+}^d$-adaptation of $\texttt{SOURCE}$ to solve $\texttt{Target}$ gives $\texttt{COMPL\_TARGET}$ such that

$$\texttt{COMPL\_TARGET} = \bigcup_{i,\delta_i=\Delta} \texttt{ComplTarget}_i$$

where $\texttt{ComplTarget}_i$ is the result of the $\dot{+}^d$-adaptation of $\texttt{Source}_i$ to solve $\texttt{Target}$.

First, let us consider that there is only one i such that $\delta_i = \Delta$. Then $\texttt{COMPL\_TARGET} = \texttt{ComplTarget}_i$. Therefore if the retrieval process aims at selecting the source case $\texttt{Source}_i \in \texttt{CaseBase}$ that minimizes $d(\texttt{DK} \cap \texttt{Source}_i, \texttt{DK} \cap \texttt{Target})$ then

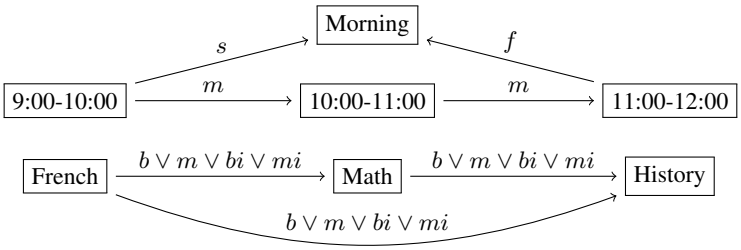
$$\texttt{CBR}(\{\texttt{SOURCE}\}, \texttt{Target}) = \texttt{CBR}(\texttt{CaseBase}, \texttt{Target}) \tag{22}$$

Now, let us consider that there are ex aequo source cases for such a retrieval process: there are several source cases $\texttt{Source}$ such that $d(\texttt{DK} \cap \texttt{Source}, \texttt{DK} \cap \texttt{Target}) = \Delta$. Then the equation (22) still holds if the two following modifications are made:

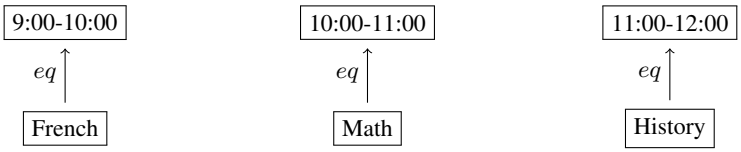
- Retrieval returns the set S of source cases $\texttt{Source}$ minimizing $d(\texttt{DK} \cap \texttt{Source}, \texttt{DK} \cap \texttt{Target})$;

for each $x \in \{\text{French}, \text{History}, \text{Math}\}$ and each $y \in \{\text{9:00-10:00}, \text{10:00-11:00}, \text{11:00-12:00}\}$

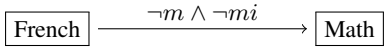
$x\ (eq \vee m \vee mi \vee b \vee bi)\ y$ $\qquad$ $x\ (s \vee d \vee f)$ Morning



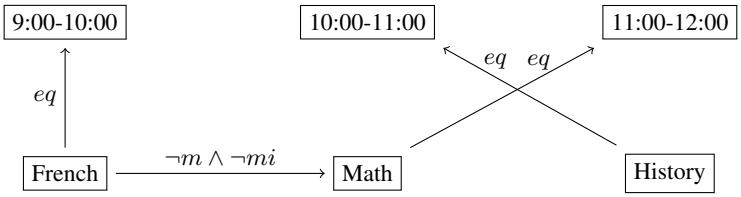
(a) `DK`. For each of the three courses and each of the three slots, either the course coincides with the slot or it has an intersection of length 0 with it, moreover, the course takes place during the morning. The slot 10:00-11:00 (resp., 11:00-12:00) immediatly follows the slot 9:00-10:00 (resp., 10:00-11:00). There is no intersection of length non null between two different courses.



(b) `Source`. The previous year, the slot 9:00-10:00 (resp., 10:00-11:00, 11:00-12:00) was attributed to French (resp., Math, resp., History).



(c) `Target`. The French course and the Math course must not meet any more. (Technically, $\neg m \wedge \neg mi$ is an abbreviation for $eq \vee b \vee bi \vee d \vee di \vee f \vee fi \vee o \vee oi \vee s \vee si$.)



(d) `ComplTarget` (without the pieces of knowledge given by `DK`). The adaptation has consisted in the exchange of slots between the Math teacher and the History teacher.

Fig. 5: Example of revision-based adaptation in the Allen algebra.

– Adaptation first performs a $\dot{+}^d$-adaptation of each $\texttt{Source} \in S$ and then takes the union of the results.

Therefore, if the distance d used for $\dot{+}^d$-adaptation is also used for the retrieval process as it is described above, the whole CBR inference can be specified by $\dot{+}^d$ and DK: DK is the "static" knowledge (stating that some case instances are not appropriate) and d is the "dynamic" knowledge about the modification from a case instance to another one. This can be linked to the general principle of "adaptation-guided retrieval" [30] stating that the adaptation knowledge should be used during retrieval: a source case must be preferred to another one if the former requires less "adaptation effort" (this adaptation effort being measured thanks to d for $\dot{+}^d$-based CBR).

6 REVISOR: a set of revision-based adaptation engines

REVISOR gathers several adaptation engines under GPL license and is available at `http://revisor.loria.fr`. In the current version, several engines have been developped.

REVISOR/PL implements $\dot{+}^d$ and the corresponding revision-based adaptation, where the formalism is propositional logic with a finite number of variables and where d is the weighted Hamming distance between interpretations.

REVISOR/PLAK implements $\dot{+}^d$ and the corresponding revision-based adaptation, where the formalism is propositional logic with a finite number of variables and where d is defined by equation (14) with d_0 a weighted Hamming distance and AK a set of adaptation rules defined by the user, together with their costs.

REVISOR/CLC implements $\dot{+}^d$ and the corresponding revision-based adaptation, where formulas are conjunctions of linear constraints with variables on $\mathbb{Z}$ and on $\mathbb{R}$ and where d is a L_1 distance.

REVISOR/QA implements $\dot{+}^d$ and the corresponding revision-based adaptation, in one of the following qualitative algebras: the Allen algebra, INDU and RCC8.

REVISOR/CLC and REVISOR/QA have been used for the system Taaable.

7 Conclusion

Case-based reasoning systems use similarity (usually in the form of a similarity measure or a distance). This is obvious for the retrieval of a case similar to the target case but this chapter shows how it can be used for adaptation: an important class of revision operators is based on distances. Indeed, $\dot{+}^d$-based adaptation can be reformulated as the process of selecting the case instances that are the closest ones to the source case, in the metric space $(\mathcal{U}, d)$, with constraints given by DK. Since adaptation aims at solving a certain type of analogical problem (in which two of the four elements in the analogy are problems, the other ones—including the unknown—are solutions), this approach concretely relates analogical reasoning with belief revision.

This approach to adaptation has a good level of generality since it captures some other approaches to adaptation. This is useful for at least two reasons. First, it proposes

a general framework for specifying adaptation, covering many approaches. Second, it enables to modify an existing adaptation process by taking into account the domain knowledge in this approach. For instance, rule-based adaptation does not consider the domain knowledge (unless it is considered in the adaptation rules): it works on the case universe $\mathcal{U}$. Revision-based adaptation enables to re-specify it and then to have it working in the case universe $\mathcal{U} \cap \mathtt{DK}$ (i.e., the case universe without the case instances known to be illicit).

Finally, from our own experience, this way to consider adaptation, has appeared as a useful guide to specify and to realize adaptation procedures.

However, even if revision-based adaptation would capture all the approaches to adaptation (and we do not claim that it is the case), it would not close the investigations about adaptation in CBR. Indeed, a revision operator is parametrized by a topology (usually a distance) and the issue of the choice of an appropriate topology, is far from being completely addressed. In fact, the choice of this topology is an adaptation knowledge acquisition issue and an important aspect of the research on revision-based adaptation is to have related this topology of the case universe with the adaptation knowledge.

References

1. Riesbeck, C.K., Schank, R.C.: Inside Case-Based Reasoning. Lawrence Erlbaum Associates, Inc., Hillsdale, New Jersey (1989)
2. Alchourrón, C.E., Gärdenfors, P., Makinson, D.: On the Logic of Theory Change: partial meet functions for contraction and revision. Journal of Symbolic Logic **50** (1985) 510–530
3. Maximini, K., Maximini, R., Bergmann, R.: An investigation of generalized cases. In Ashley, K.D., Bridge, D., eds.: Proceedings of the 5th International Conference on Case Base Reasoning (ICCBR'03). Volume 2689 of LNAI., Trondheim, Norway, Springer (June 2003) 261–275
4. Carbonell, J.G.: Learning by analogy: Formulating and generalizing plans from past experience. In R. S. Michalski and J. G. Carbonell and T. M. Mitchell, ed.: Machine Learning, An Artificial Intelligence Approach. Morgan Kaufmann, Inc. (1983) 137–161
5. Carbonell, J.G.: Derivational analogy: A Theory of Reconstructive Problem Solving and Expertise Acquisition. In: Machine Learning. Volume 2. Morgan Kaufmann, Inc. (1986) 371–392
6. Katsuno, H., Mendelzon, A.: Propositional knowledge base revision and minimal change. Artificial Intelligence **52**(3) (1991) 263–294
7. Dalal, M.: Investigations into a theory of knowledge base revision: Preliminary report. In: AAAI. (1988) 475–479
8. Konieczny, S., Lang, J., Marquis, P.: DA^2 merging operators. Artificial Intelligence **157**(1-2) (2004) 49–79
9. Lieber, J.: Application of the Revision Theory to Adaptation in Case-Based Reasoning: the Conservative Adaptation. In: Proceedings of the 7th International Conference on Case-Based Reasoning (ICCBR-07). Lecture Notes in Artificial Intelligence 4626. Springer, Belfast (2007) 239–253
10. Kolodner, J.: Case-Based Reasoning. Morgan Kaufmann, Inc. (1993)
11. Karmarkar, N.: A new polynomial-time algorithm for linear programming. Combinatorica **4**(4) (1984) 373–396
12. Cojan, J., Lieber, J.: Belief Merging-based Case Combination. In: Case-Based Reasoning Research and Development (ICCBR 2009). (2009) 105–119

13. Blansché, A., Cojan, J., Dufour Lussier, V., Lieber, J., Molli, P., Nauer, E., Skaf Molli, H., Toussaint, Y.: TAAABLE 3: Adaptation of ingredient quantities and of textual preparations. In: 18h International Conference on Case-Based Reasoning - ICCBR 2010, "Computer Cooking Contest" Workshop Proceedings. (2010)
14. Lieber, J., Napoli, A.: Correct and Complete Retrieval for Case-Based Problem-Solving. In Prade, H., ed.: Proceedings of the 13th European Conference on Artificial Intelligence (ECAI-98), Brighton, United Kingdom. (1998) 68–72
15. Craw, S., Wiratunga, N., Rowe, R.C.: Learning adaptation knowledge to improve case-based reasoning. Artificial Intelligence **170**(16-17) (2006) 1175–1192
16. d'Aquin, M., Badra, F., Lafrogne, S., Lieber, J., Napoli, A., Szathmary, L.: Case Base Mining for Adaptation Knowledge Acquisition. In Veloso, M.M., ed.: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07), Morgan Kaufmann, Inc. (2007) 750–755
17. Jarmulak, J., Craw, S., Rowe, R.: Using Case-Base Data to Learn Adaptation Knowledge for Design. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI'01), Morgan Kaufmann, Inc. (2001) 1011–1016
18. Leake, D.B., Kinley, A., Wilson, D.C.: Acquiring Case Adaptation Knowledge: A Hybrid Approach. In: AAAI/IAAI. Volume 1. (1996) 684–689
19. Stahl, A., Bergmann, R.: Applying Recursive CBR for the Customization of Structure Products in an Electronic Shop. In Blanzieri, E., Portinale, L., eds.: Advances in Case-Based Reasoning — Proceedings of the fifth European Workshop on Case-Based Reasoning (EWCBR-2k). Lecture Notes in Artificial Intelligence 1898. Springer (2000) 297–308
20. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P., eds.: The Description Logic Handbook. Cambridge University Press, cambridge, UK (2003)
21. Flouris, G., Plexousakis, D., Antoniou, G.: On Applying the AGM theory to DLs and OWL. In Gil, Y., Motta, E., eds.: Proceedings of the 4th International Semantic Web Conference (ISWC 2005). LNCS 3729, Springer (November 2005) 216–231
22. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.: Debugging unsatisfiable classes in OWL ontologies. Web Semantics: Science, Services and Agents on the World Wide Web **3**(4) (2005) 268–293
23. Cojan, J., Lieber, J.: An Algorithm for Adapting Cases Represented in $\mathcal{ALC}$. In: 22th Internationational Joint Conference on Artificial Intelligence, Barcelone Espagne (07 2011)
24. Allen, J.F.: An interval-based representation of temporal knowledge. In: Proceedings 7th International Joint Conference on Artificial Intelligence (IJCAI 1981). (1981) 221–226
25. Randell, D.A., Cui, Z., Cohn, A.G.: A spatial logic based on regions and connection. In: Knowledge Representation. (1992) 165–176
26. Condotta, J.F., Kaci, S., Marquis, P., Schwind, N.: A Syntactical Approach to Qualitative Constraint Networks Merging. In: Proc. of the 17th LPAR (Logic for Programming, Artificial Intelligence and Reasoning). (2010) 233–247
27. Dufour-Lussier, V., Le Ber, F., Lieber, J., Martin, L.: Adapting Spatial and Temporal Cases. In Ian Watson, B.D.A., ed.: International Conference for Case-Based Reasoning. Volume 7466 of Lecture Notes in Artificial Intelligence., Lyon, France, Amélie Cordier, Marie Lefevre, Springer (September 2012) 77–91
28. Pujari, A.K., Kumari, G.V., Sattar, A.: INDU: An Interval & Duration Network. Advanced Topics in Artificial Intelligence (1999) 291–303
29. Ligozat, G.: On generalized interval calculi. In: Proceedings of the 9th National Conference of the American Association for Artificial Intelligence (AAAI), Anaheim, CA, AAAI Press/MIT Press (1991) 234–240
30. Smyth, B., Keane, M.T.: Using adaptation knowledge to retrieve and adapt design cases. Knowledge-Based Systems **9**(2) (1996) 127–135