The French Correction: When Retrieval is Harder to Specify than Adaptation

Yves Lepage, Jean Lieber, Isabelle Mornard, Emmanuel Nauer, Julien Romary, Reynault Sies

The French Correction: When Retrieval is Harder to Specify than Adaptation*

Yves Lepage[1], Jean Lieber[2], Isabelle Mornard[2,3], Emmanuel Nauer[2], Julien Romary[2], and Reynault Sies[2]

[1] Waseda University, IPS, 2-7 Hibikino, 808-0135 Kitakyushu, Japan
yves.lepage@waseda.jp
[2] Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
{firstname.secondname}@loria.fr
[3] Université Jean Monnet, Université de Lyon, France,
isabelle.mornard@etu.univ-st-etienne.fr
(The authors are listed alphabetically according to their second names.)

Abstract. A common idea in the field of case-based reasoning is that the retrieval step can be specified by the use of some similarity measure: the retrieved cases maximize the similarity to the target problem and, then, the adaptation step has to take into account the mismatches between the retrieved cases and the target problem in order to solve this latter. The use of this methodological schema for the application described in this paper has proven to be non efficient. Indeed, designing a retrieval procedure without the precise knowledge of the adaptation procedure has not been possible. The domain of this application is the correction of French sentences: a problem is an incorrect sentence and a valid solution is a correction of this problem. Adaptation consists in solving an analogical equation that enables to execute the correction of the retrieved case on the target problem. Thus, retrieval has to ensure that this application is feasible. The first version of such a retrieval procedure is described and evaluated: it is a knowledge-light procedure that does not use linguistic knowledge about French.

Keywords: case-based reasoning, retrieval, analogy, sentence correction

1 Introduction

Case-based reasoning (CBR [8]) aims at solving a problem with the help of a case base, where a case is the representation of a problem-solving episode. It is often decomposed in several steps including its inference steps, retrieval and adaptation. Retrieval consists in selecting one or several case(s) from the case base that is/are similar to the target problem (i.e., the problem to be

* The authors wish to thank Bruno Guillaume who has given us some valuable remarks for this project and Nicolas Lasolle who has helped us for its evaluation.

solved). Adaptation consists in modifying this/these retrieved case(s) in order to obtain a plausible solution to the target problem. For many CBR applications, the specification of retrieval is quite simple and amounts to choose a similarity metric or a distance function on the problem space. Then, the main difficulty of retrieval is algorithmic: how to design a program that efficiently implements this specification. By contrast, adaptation is often considered as more difficult to specify within a given application: the issue of its efficient implementation comes only in a second time.

The CBR application presented in this paper contrasts with this viewpoint: the adaptation has been rather simple to specify, whereas the first version of retrieval giving some relevant This CBR application aims at correcting linguistic errors in French sentences: its input is an incorrect sentence, its output is a correction of this sentence. For the sake of readability, the examples in this paper are in English. It is noteworthy that the correction is only at the grammatical level: the corrected sentence is expected to be orthographically and syntactically correct but there are no expected correction at the semantic level. For example, consider the following example:

Input:	*Tomatoes grows outdoors in winter.*
Output:	*Tomatoes grow outdoors in winter.*

The output sentence is orthographically and syntactically correct, but no correction is made at the semantic level (that would consist, for example, in substituting *winter* with *summer*).

The system presented in this paper is called *The French Correction* (abbreviated in TFC) and has several features that are worth mentioning. First, this first version of TFC is intentionally knowledge-light: almost all its knowledge lies in the case base (very little domain knowledge). Therefore, the system should give similar results in another alphabetic language using spaces for separating words. Indeed, it works at the character level (letters and punctuation marks). Second, TFC is *not* meant to be competitive with other correcting systems that are currently used in, e.g., word processing systems. By contrast, TFC's main goal is to provide a playground for CBR research.

This paper is organized as follows. Section 2 presents some preliminaries: the main assumptions and notations on CBR that are considered in this paper and some notions related to strings and to analogies. Section 3 informally specifies the TFC system. Building a CBR system requires the acquisition of a case base: case authoring is described in Section 4. The case-based inference is described in Sections 5 and 6: adaptation first and then retrieval. Indeed, the TFC retrieval module must be adaptation-guided for this application, hence this unusual order in the presentation. Section 7 presents the evaluation of TFC. Section 8 discusses the design of this system and, in particular, its originality with respect to the respective design of the retrieval and adaptation phases. Finally, Section 9 concludes this article with some research directions around *The French Correction*.

Main objective of this paper. This paper presents a problem that is easy to understand but not so easy to solve, together with a first baseline solution. Such a problem could be a challenge for the CBR community, or even a benchmark. The authors agree to distribute the case base and the test base for this purpose.

2 Preliminaries

This section recalls some notions related to CBR, strings and analogies. These notions are used in particular to define the adaptation step of TFC which relies on analogies on strings.

2.1 Preliminaries: assumptions and notations about CBR

Let $\mathcal{P}$ and $\mathcal{S}$ be two sets respectively called the *problem space* and *the solution space*. A *problem* $\mathtt{x}$ (resp., a solution $\mathtt{y}$) is by definition an element of $\mathcal{P}$ (resp., $\mathcal{S}$). Let $\rightsquigarrow$ be a relation on $\mathcal{P} \times \mathcal{S}$. For $(\mathtt{x}, \mathtt{y}) \in \mathcal{P} \times \mathcal{S}$, $\mathtt{x} \rightsquigarrow \mathtt{y}$ is read "$\mathtt{x}$ has for solution $\mathtt{y}$" or "$\mathtt{y}$ solves $\mathtt{x}$". The relation $\rightsquigarrow$ is in general incompletely known, though it is known to hold for a finite set of pairs $(\mathtt{x}^s, \mathtt{y}^s)$. This finite set is called the *case base*, denoted by $\mathtt{CB}$, and every $(\mathtt{x}^s, \mathtt{y}^s) \in \mathtt{CB}$ is called a *source case*.

CBR aims at solving a new problem, called the *target problem* and denoted by $\mathtt{x}^{\mathtt{tgt}}$, with the help of $\mathtt{CB}$.

The process model of CBR consists (1) in selecting k source cases similar to the target problem, (2) in inferring from these k source cases a candidate solution $\mathtt{y}^{\mathtt{tgt}}$ of $\mathtt{x}^{\mathtt{tgt}}$, (3) in confronting the hypothetical case $(\mathtt{x}^{\mathtt{tgt}}, \mathtt{y}^{\mathtt{tgt}})$ to, e.g., a human that validate it as a case if $\mathtt{x}^{\mathtt{tgt}} \rightsquigarrow \mathtt{y}^{\mathtt{tgt}}$ or correct $\mathtt{y}^{\mathtt{tgt}}$ otherwise, (4) in storing the validated and potentially corrected case $(\mathtt{x}^{\mathtt{tgt}}, \mathtt{y}^{\mathtt{tgt}})$ in $\mathtt{CB}$ if this storage is deemed useful. These steps are called (1) retrieval, (2) adaptation, (3) validation and repair, and (4) storage (aka as retrieve, reuse, revise and retain in the 4 R's model of [1]). In many applications, as the one described in this paper, $k = 1$: only one source case is retrieved and adapted to solve the target problem. For some of these applications, adaptation consists in reusing as such the solution of the retrieved case (i.e., $\mathtt{y}^{\mathtt{tgt}} = \mathtt{y}^s$): this is called *adaptation by copy*.

Case retrieval is often performed thanks to a distance function $\mathtt{dist}$ on $\mathcal{P}$: the selected case(s) $(\mathtt{x}^s, \mathtt{y}^s)$ being the one(s) that minimize(s) $\mathtt{dist}(\mathtt{x}^s, \mathtt{x}^{\mathtt{tgt}})$. [1] Thus, $\mathtt{dist}$ induces a ranking $\prec^{\mathtt{dist}}_{\mathtt{x}^{\mathtt{tgt}}}$ between problems defined by $\mathtt{x}^s \prec^{\mathtt{dist}}_{\mathtt{x}^{\mathtt{tgt}}} \mathtt{x}^u$ ("$\mathtt{x}^s$ is more similar to $\mathtt{x}^{\mathtt{tgt}}$ than $\mathtt{x}^u$ according to $\mathtt{dist}$") if $\mathtt{dist}(\mathtt{x}^s, \mathtt{x}^{\mathtt{tgt}}) < \mathtt{dist}(\mathtt{x}^u, \mathtt{x}^{\mathtt{tgt}})$.

The knowledge model of CBR consists in four knowledge containers: the case base $\mathtt{CB}$, the domain knowledge $\mathtt{DK}$, the retrieval knowledge $\mathtt{RK}$ and the adaptation knowledge $\mathtt{AK}$ [7]. $\mathtt{DK}$ is also known as the domain ontology and serves two purposes: giving a vocabulary for describing the cases and some integrity constraints, i.e., some *necessary* conditions for a pair $(\mathtt{x}, \mathtt{y})$ to be a case (i.e., $\mathtt{x} \rightsquigarrow \mathtt{y}$). $\mathtt{RK}$ and $\mathtt{AK}$ contain the application-dependent knowledge for, respectively,

[1] This can be equivalently defined by the maximization of the similarity measure $\mathtt{sim}$ defined by $\mathtt{sim}(\mathtt{x}^1, \mathtt{x}^2) = \frac{1}{1+\mathtt{dist}(\mathtt{x}^1, \mathtt{x}^2)}$.

performing retrieval and adaptation. A CBR system is qualified as *knowledge light* if most of the knowledge lies in $\texttt{CB}$.

2.2 Strings

Let $\mathcal{A}$ be a finite set; a *character* c is an element of $\mathcal{A}$. Let $\mathcal{A}^*$ be the set of strings on $\mathcal{A}$. The empty string is denoted by ϵ. The concatenation of two strings S and T is denoted by the juxtaposition ST. For $S, T \in \mathcal{A}^*$, S is a *substring* of T if there exist $X, Y \in \mathcal{A}^*$ such that $T = XSY$.

The length of a string $S \in \mathcal{A}^*$ is denoted by $|S|$. For instance, $|\epsilon| = 0$. For $c \in \mathcal{A}$ and $S \in \mathcal{A}^*$, $\texttt{\#occ}(c, S)$ is the number of occurrences of c in S, e.g., $\texttt{\#occ}(\textit{'t'}, \textit{tomato}) = 2$.

Given $S \in \mathcal{A}^*$, a *subsequence* of S is a string that can be obtained by removing 0 to $|S|$ characters from S. For example, *toto* is a subsequence of *tomato*. Given two strings S and T, an LCS (longest common subsequence) of S and T is a string L that is a subsequence of both S and T of maximum length (it exists, but it is not necessarily unique, though all LCSs of S and T have the same length). For example, an LCS of *tomato* and *toad* is *toa*.

For $S, T \in \mathcal{A}^*$, $\texttt{dist}_{\mathrm{LCS}}(S, T)$ is the LCS distance from S to T defined by

$$\texttt{dist}_{\mathrm{LCS}}(S, T) = |S| + |T| - 2\,|L|$$

where L is an LCS of S and T. It can be equivalently defined as the edit distance with the "delete a character" and "add a character" edit operations with the same cost of 1. For example, $\texttt{dist}_{\mathrm{LCS}}(\textit{tomato}, \textit{toad}) = 6 + 4 - 2 \times 3 = 4$.

2.3 Analogies

An *analogy* on a set $\mathcal{U}$ is a quaternary relation on $\mathcal{U}$ denoted, for $(A, B, C, D) \in \mathcal{U}^4$, by $A\!:\!B\!::\!C\!:\!D$, and read "A is to B as C is to D" that satisfies the following postulates (for any $A, B, C, D \in \mathcal{U}$): (1) $A\!:\!B\!::\!A\!:\!B$, (2) if $A\!:\!B\!::\!C\!:\!D$ then $C\!:\!D\!::\!A\!:\!B$, and (3) if $A\!:\!B\!::\!C\!:\!D$ then $A\!:\!C\!::\!B\!:\!D$.

An *analogical equation* is an expression of the form $A\!:\!B\!::\!C\!:\!x$ where $A, B, C \in \mathcal{U}$ and x is a symbol called the unknown of the analogical equation. Solving $A\!:\!B\!::\!C\!:\!x$ aims at finding the set of $D \in \mathcal{U}$ such $A\!:\!B\!::\!C\!:\!D$. An analogical equation may have 0, 1 or several solutions, depending on the analogy.

For analogies on sentences, an analogy can be built at the string level (i.e., without taking into account linguistic knowledge neither on the lexical level nor on the syntactic level) and it has been introduced for the purpose of machine translation [4]. It is defined as follows, for $A, B, C, D \in \mathcal{A}^*$:

$$A\!:\!B\!::\!C\!:\!D \quad \text{if} \quad \texttt{\#occ}(c, B) - \texttt{\#occ}(c, A) = \texttt{\#occ}(c, D) - \texttt{\#occ}(c, C), \qquad \text{(for any } c \in \mathcal{A})$$

$$\texttt{dist}_{\mathrm{LCS}}(A, B) = \texttt{dist}_{\mathrm{LCS}}(C, D) \quad \text{and} \quad \texttt{dist}_{\mathrm{LCS}}(A, C) = \texttt{dist}_{\mathrm{LCS}}(B, D)$$

For example:

You don't say! : *He does not say it.* :: *You don't know!* : *He does not know it.*

Now, a quaternary relation $\mathcal{R}$ can be defined on $\mathcal{A}^*$ that constitutes a subrelation of this analogical relation: if $\mathcal{R}(A, B, C, D)$ then $A\!:\!B\!::\!C\!:\!D$, for $A, B, C, D \in \mathcal{A}^*$. This relation is useful for the purpose of the presentation of most examples in the paper as it appears to be at the same time simpler to apprehend and sufficient for many examples. It is also used in the retrieval procedure to make it more efficient. For $A, B, C, D \in \mathcal{A}^*$, $\mathcal{R}(A, B, C, D)$ holds if there exists a substring S common to A and C and a substring T common to B and D such that B (resp., D) is obtained by a string replacement of S with T in A (resp., in C). Formally, $\mathcal{R}(A, B, C, D)$ if there exist $S, T, X, Y, X', Y' \in \mathcal{A}^*$ such that $A = XSY$, $B = XTY$, $C = X'SY'$ and $D = X'TY'$. For example (with the occurrences of S and T underlined),

$$\text{if}\left|\begin{array}{ll} A = \textit{Do you want some } \underline{\textit{coffee}}\textit{?} & C = \textit{This } \underline{\textit{coffee}} \textit{ is hot!} \\ B = \textit{Do you want some } \underline{\textit{tea}}\textit{?} & D = \textit{This } \underline{\textit{tea}} \textit{ is hot!} \end{array}\right.$$

then $\mathcal{R}(A, B, C, D)$ and, thus, $A\!:\!B\!::\!C\!:\!D$.

It can be noted that $\mathcal{R}$ is not an analogical relation (it satisfies the first and second postulates, but not the third one).

3 Goals of the TFC system

TFC is a CBR system that takes as input a sentence $\mathtt{x}^{\mathtt{tgt}}$ that is supposed to be incorrect and gives as output a sentence $\mathtt{y}^{\mathtt{tgt}}$ with the following objective: $\mathtt{y}^{\mathtt{tgt}}$ is a correction of $\mathtt{x}^{\mathtt{tgt}}$ at the language level. For a source case $(\mathtt{x}^s, \mathtt{y}^s)$, $\mathtt{x}^s$ is an incorrect sentence and $\mathtt{y}^s$ is a sentence obtained by correcting $\mathtt{x}^s$.

It is practical, in particular for further explanations, to consider special types of cases, called SR cases (SR stands for String Replacement). An SR case $(\mathtt{x}, \mathtt{y})$ is such that $\mathtt{y}$ is obtained by a single string replacement of a substring S of $\mathtt{x}$ by a string T where S and T contain no space (i.e., the modification lies within a single word). For example, $(\mathtt{x}^s, \mathtt{y}^s)$ defined below is an SR source case (with $S = es$ and $T = \epsilon$):

$$\mathtt{x}^s = \textit{They goes to the beach.} \qquad \mathtt{y}^s = \textit{They go to the beach}$$

The TFC case base in its current version contains only French sentences, though most of the ideas developed in this paper can be considered in another alphabetical language. Let $\mathcal{A}$ be the set of characters in such a language (letters, letters with diacritics, punctuation marks, space, etc.). Therefore, every problem $\mathtt{x}$ and every solution $\mathtt{y}$ belongs to $\mathcal{A}^*$, therefore $\mathcal{P} = \mathcal{S} = \mathcal{A}^*$. It is noteworthy that, in TFC, the problems and solutions belong to a common space (i.e., from an algorithmic viewpoint, they have the same type), which is not the general case in CBR.

4 Case authoring

Since TFC, at least in the version presented in this paper, is a knowledge-light CBR system, the acquisition of its case base is crucial. For this purpose, two approaches have been considered: a manual one and a semi-automatic one.

The manual case authoring approach has consisted in searching for documents about French grammar, frequent mistakes, etc., and in defining cases reflecting such errors. For example, in English, the case $(\mathtt{x}^s, \mathtt{y}^s)$ can be found with:[2]

$$\mathtt{x}^s = \textit{You like dance with me?} \qquad \mathtt{y}^s = \textit{Would you like to dance with me?}$$

The irregular forms in the language can be used to define cases. For example, the verb *to meet* is irregular: its preterit is *met* (and not *meeted*), hence the following case:

$$\mathtt{x}^s = \textit{He meeted her yesterday.} \qquad \mathtt{y}^s = \textit{He met her yesterday.}$$

Then, other cases were added by various contributors. The case base built that way is rather small (300 cases at the time of submission of this article), and the source cases are chosen to cover frequent common mistakes. Of course, when TFC fails to correctly solve a problem, the last steps of the CBR process (repair and storage) leads to an enrichment of the case base. This aspect of the system has not been studied in depth yet, but it is operational.

The semi-automatic case authoring approach uses WiCoPaCo [3] which is a collection of sentence (or text) pairs $(\mathtt{x}, \mathtt{y}) \in \mathcal{A}^{*2}$ taken from the Wikipedia French pages, where $\mathtt{x}$ is a sentence written by an editor and $\mathtt{y}$ is a sentence replacing it (in a next edition of the same article). WiCoPaCo comes with some markups explaining some of the changes.

However, using WiCoPaCo as such for a TFC case base has appeared to be inefficient. Indeed, a WiCoPaCo pair $(\mathtt{x}, \mathtt{y})$ is not necessarily a valid case, for example, $\mathtt{y}$ may contain an error, or $\mathtt{y}$ corrects $\mathtt{x}$ at a semantic level, or corresponds to an information update. Since WiCoPaCo contains hundreds of thousands pairs, a manual selection of such pairs that could be used as TFC cases would be too tedious for the project. Some automatic filters have been defined for deleting some irrelevant pairs, but they are not currently sufficient to obtain a TFC case base with a low level of noise. That is why, for this work, it has been decided to use a small case base containing 300 well-formed cases collected from several persons whose native language is French.

[2] `https://www.engvid.com/english-resource/50-common-grammar-mistakes-in-english/`

5 Adaptation

A single case adaptation is used in TFC. Consider first an example of adaptation problem, with $(\mathtt{x}^s, \mathtt{y}^s)$ the retrieved case and $\mathtt{x}^{\mathtt{tgt}}$, the problem to be solved:

$$\mathtt{x}^s = \textit{David would not eating his soup.} \qquad \mathtt{y}^s = \textit{David would not eat his soup.}$$
$$\mathtt{x}^{\mathtt{tgt}} = \textit{Cindy will going to Nancy.}$$

In this example, the error corrected in $(\mathtt{x}^s, \mathtt{y}^s)$ corresponds to the inappropriate *-ing* form at the end of the verb. The same error occurs in $\mathtt{x}^{\mathtt{tgt}}$, thus the transformation from $\mathtt{x}^s$ to $\mathtt{y}^s$ can be suggested as correction:

$$\mathtt{y}^{\mathtt{tgt}} = \textit{Cindy will go to Nancy.}$$

Therefore, with the analogy on strings defined in Section 2.3, $\mathtt{y}^{\mathtt{tgt}}$ is a solution of the following analogical equation with unknown y:

$$\mathtt{x}^s : \mathtt{y}^s :: \mathtt{x}^{\mathtt{tgt}} : y \tag{1}$$

More generally, the adaptation of TFC consists in solving the analogical equation (1). When this equation has no solution, adaptation fails. When it has several solutions, TFC's adaptation proposes all of them if there is no way to make a preference among them: this issue is considered again in the next section.

In practice, if $(\mathtt{x}^s, \mathtt{y}^s)$ is an SR source case, solving $\mathtt{x}^s : \mathtt{y}^s :: \mathtt{x}^{\mathtt{tgt}} : y$ consists, in such a situation, in solving $\mathcal{R}(\mathtt{x}^s, \mathtt{y}^s, \mathtt{x}^{\mathtt{tgt}}, y)$. This occurs for the above example: $\mathtt{y}^{\mathtt{tgt}}$ is obtained by substituting *ing* with ϵ in $\mathtt{x}^{\mathtt{tgt}}$.

Sometimes, there are several retrieved cases, when the retrieval procedure cannot distinguish them. When this occurs, the adaptation is performed on all these cases and this provides a multiset of solutions. The final result is an element of this multiset with the highest multiplicity.

What makes this adaptation rather simple to define is first that it is based on a previous work on analogy on strings [4] and second the fact that the problem and solution spaces coincide for this application. Moreover, although it is defined at a character level (since it only relies on the analogical relation defined on strings, without any linguistic knowledge), it gives results that are quite convincing for the correction of sentences, provided that an appropriate correction case exists in the case base, which is the role of the case authoring process, and provided that such an appropriate case has been selected, which is the role of case retrieval.

6 Retrieval

Retrieval aims at selecting a source case to be adapted. Given a target problem $\mathtt{x}^{\mathtt{tgt}}$ and two source cases $(\mathtt{x}^1, \mathtt{y}^1)$ and $(\mathtt{x}^2, \mathtt{y}^2)$, which one, if any, should be preferred? Following the principle of adaptation-guided retrieval (AGR [9]), it should be a case that is adaptable (i.e., the adaptation function returns a candidate solution to $\mathtt{x}^{\mathtt{tgt}}$, given this case and $\mathtt{x}^{\mathtt{tgt}}$). For comparing two cases $(\mathtt{x}^1, \mathtt{y}^1)$

and $(\mathtt{x}^2, \mathtt{y}^2)$ that both can be adapted in candidate solutions $\mathtt{y}^{1,\mathtt{tgt}}$ and $\mathtt{y}^{2,\mathtt{tgt}}$ to the target problem $\mathtt{x}^{\mathtt{tgt}}$, an ideal retrieval function will choose $(\mathtt{x}^1, \mathtt{y}^1)$ if the candidate solution $\mathtt{y}^{1,\mathtt{tgt}}$ is better than $\mathtt{y}^{2,\mathtt{tgt}}$ (e.g., $\mathtt{y}^{1,\mathtt{tgt}}$ is a correct solution of $\mathtt{x}^{\mathtt{tgt}}$ while $\mathtt{y}^{2,\mathtt{tgt}}$ is not).

This principle of AGR adapted to the retrieval problem of TFC is considered via an example. Then, the description of the knowledge-light retrieval approach of the first version of TFC is presented.

6.1 Example

Consider the following target problem:

$$\mathtt{x}^{\mathtt{tgt}} = \textit{George has read this books.}$$

and the three source cases $(\mathtt{x}^s, \mathtt{y}^s)$ $(s \in \{1, 2, 3\})$:

$$\begin{array}{ll} \mathtt{x}^1 = \textit{George have read this book.} & \mathtt{y}^1 = \textit{George has read this book.} \\ \mathtt{x}^2 = \textit{You has read this book.} & \mathtt{y}^2 = \textit{You have read this book.} \\ \mathtt{x}^3 = \textit{Put it on the tables, please.} & \mathtt{y}^3 = \textit{Put it on the table, please.} \end{array}$$

Now, consider someone who is agnostic to the task to be performed by the TFC system and who does not know the solutions $\mathtt{y}^1$, $\mathtt{y}^2$ and $\mathtt{y}^3$. This person is asked to rank $\mathtt{x}^1$, $\mathtt{x}^2$ and $\mathtt{x}^3$ according to their similarity to $\mathtt{x}^{\mathtt{tgt}}$, without any precision on what "similar" means. It is likely that he/she would give the ranking $\mathtt{x}^1 \prec_{\mathtt{x}^{\mathtt{tgt}}} \mathtt{x}^2 \prec_{\mathtt{x}^{\mathtt{tgt}}} \mathtt{x}^3$ where $\prec_{\mathtt{x}^{\mathtt{tgt}}}$ is read "is strictly more similar to $\mathtt{x}^{\mathtt{tgt}}$ than". This ranking is consistent with the one that is induced by, e.g., the LCS distance function:

$$\mathtt{dist}_{\mathrm{LCS}}(\mathtt{x}^1, \mathtt{x}^{\mathtt{tgt}}) \quad < \quad \mathtt{dist}_{\mathrm{LCS}}(\mathtt{x}^2, \mathtt{x}^{\mathtt{tgt}}) \quad < \quad \mathtt{dist}_{\mathrm{LCS}}(\mathtt{x}^3, \mathtt{x}^{\mathtt{tgt}})$$

Now, it is argued that the ranking of these three cases with respect to the target problem should be the reverse order, according to the defined adaptation process and to the English language correctness.

First, the source case $(\mathtt{x}^1, \mathtt{y}^1)$ is simply *not* adaptable to solve $\mathtt{x}^{\mathtt{tgt}}$: the analogical equation $\mathtt{x}^1 : \mathtt{y}^1 :: \mathtt{x}^{\mathtt{tgt}} : y$ has no solution. Indeed, if y was a solution, $\mathtt{\#occ}(\text{'}v\text{'}, \mathtt{y}^1) - \mathtt{\#occ}(\text{'}v\text{'}, \mathtt{x}^1) = \mathtt{\#occ}(\text{'}v\text{'}, y) - \mathtt{\#occ}(\text{'}v\text{'}, \mathtt{x}^{\mathtt{tgt}})$, thus $\mathtt{\#occ}(\text{'}v\text{'}, y) = 0 - 1 + 0 = -1$, which is not possible.

The source case $(\mathtt{x}^2, \mathtt{y}^2)$ is adaptable to solve $\mathtt{x}^{\mathtt{tgt}}$: the analogical equation $\mathtt{x}^2 : \mathtt{y}^2 :: \mathtt{x}^{\mathtt{tgt}} : y$ is solvable and its solutions are:

$$\begin{array}{lll} & y = \textit{George ha}\underline{\textit{ve}}\textit{ read this books.} & y = \textit{George has read thi}\underline{\textit{ve}}\textit{ books.} \\ \text{and} & y = \textit{George has read this book}\underline{\textit{ve}}\textit{.} & \end{array}$$

Unfortunately, both solutions are incorrect solutions of $\mathtt{x}^{\mathtt{tgt}}$, since these two sentences violate the English language.

The source case $(\mathtt{x}^3, \mathtt{y}^3)$ is also adaptable to solve $\mathtt{x}^{\mathtt{tgt}}$, thus, according to the adaptation-guided principle, both $(\mathtt{x}^2, \mathtt{y}^2)$ and $(\mathtt{x}^3, \mathtt{y}^3)$ are preferred to $(\mathtt{x}^1, \mathtt{y}^1)$, given the target problem $\mathtt{x}^{\mathtt{tgt}}$. The solutions of $\mathtt{x}^3 \mathtt{:} \mathtt{y}^3 \mathtt{::} \mathtt{x}^{\mathtt{tgt}} \mathtt{:} y$ are:

$$y = \textit{George ha read this books.} \qquad y = \textit{George has read thi books.}$$
$$\text{and} \qquad y = \textit{George has read this book.}$$

The third solution is a correct correction of $\mathtt{x}^{\mathtt{tgt}}$, therefore $(\mathtt{x}^3, \mathtt{y}^3)$ is preferred to $(\mathtt{x}^2, \mathtt{y}^2)$ according to an *a posteriori* help from a domain expert (i.e., someone who can say which sentence is correct in English and which is not). The design of retrieval aims at finding a way of *predicting* which cases are the most likely to provide a correct solution to $\mathtt{x}^{\mathtt{tgt}}$.

What this third example shows is that, even with a retrieved case such that adaptation gives a correct solution, it may give other solutions that are not: among the 3 values for y proposed above, only the third one is correct. Therefore, an interesting byproduct of retrieval would be to have some relevant information in order to discriminate among these solutions.

With a sufficient level in English linguistic knowledge, retrieval could consist in finding the error in $\mathtt{x}^{\mathtt{tgt}}$ and then in finding a source case that represents the correction of the same error (at the character level). Now, a challenge is to design a retrieval process that uses no linguistic knowledge. The retrieval process presented in the next section is a first attempt to meet this challenge.

6.2 Proposed retrieval procedure

The retrieval procedure described below is knowledge-light. In particular, it uses no linguistic knowledge about French, except for the fact that sentences can be split in words. This procedure consists in two phases, a filter phase and a ranking phase.

The filter phase aims at discarding cases that are not adaptable to solve the target problem $\mathtt{x}^{\mathtt{tgt}}$ such as the case $(\mathtt{x}^1, \mathtt{y}^1)$ in the example. More generally, a source case $(\mathtt{x}^s, \mathtt{y}^s)$ is filtered if the analogical equation $\mathtt{x}^s \mathtt{:} \mathtt{y}^s \mathtt{::} \mathtt{x}^{\mathtt{tgt}} \mathtt{:} y$ has no solution, which can be easily tested.

This filter can be efficiently implemented by considering necessary conditions and sufficient conditions for $(\mathtt{x}^s, \mathtt{y}^s)$ to be adaptable to solve $\mathtt{x}^{\mathtt{tgt}}$ (i.e., the analogical equation $\mathtt{x}^s \mathtt{:} \mathtt{y}^s \mathtt{::} \mathtt{x}^{\mathtt{tgt}} \mathtt{:} y$ has at least one solution). If a necessary condition does not hold, then the case is not adaptable and must be filtered. If a sufficient condition holds, then the case is adaptable (no more testing is needed at this phase of retrieval for this case). Examples of such conditions are presented below.

A *necessary condition* based on the definition of the analogy on strings is related to the character count. Indeed, if y is a solution of $\mathtt{x}^s \mathtt{:} \mathtt{y}^s \mathtt{::} \mathtt{x}^{\mathtt{tgt}} \mathtt{:} y$, then, for every character c,

$$\mathtt{\#occ}(c, y) = \underbrace{\mathtt{\#occ}(c, \mathtt{x}^{\mathtt{tgt}})}_{①} + \underbrace{\mathtt{\#occ}(c, \mathtt{y}^s) - \mathtt{\#occ}(c, \mathtt{x}^s)}_{②}$$

So, $\texttt{\#occ}(c, y)$ can be computed fast, since ① depends only on the target problem and ② depends only on the source case and, thus, can be computed offline. Now, the number of occurrences of a character in a string has to be nonnegative, thus, if the value computed for $\texttt{\#occ}(c, y)$ is negative, then $(\mathtt{x}^s, \mathtt{y}^s)$ is not adaptable to solve $\mathtt{x}^{\mathtt{tgt}}$ and can be filtered. It is noteworthy that only the characters c occurring in $\mathtt{x}^s$, $\mathtt{y}^s$ and/or $\mathtt{x}^{\mathtt{tgt}}$ need to be taken into account.

A *sufficient condition* is related to $\mathcal{R}$, the subrelation of the analogical relation introduced in the preliminaries. Given a source case $(\mathtt{x}^s, \mathtt{y}^s)$, there exist ordered pairs of strings (S, T) such that $\mathtt{y}^s$ is obtained by substring replacement of S with T in $\mathtt{x}^{\mathtt{tgt}}$ (since $(S, T) = (\mathtt{x}^s, \mathtt{y}^s)$ is such a pair, the existence of such pairs is ensured). Now, a pair of strings (S, T) with S of minimal length is associated to $(\mathtt{x}^s, \mathtt{y}^s)$ in an offline process. The three cases $(\mathtt{x}^s, \mathtt{y}^s)$ $(s \in \{1, 2, 3\})$ introduced in the previous section are used below to illustrate the procedure. With the cases $(\mathtt{x}^s, \mathtt{y}^s)$ of the example developed in the previous section $(s \in \{1, 2, 3\})$:

$$\text{for } s = 1, (S, T) = (ve, s) \quad \text{for } s = 2, (S, T) = (s, ve) \quad \text{for } s = 3, (S, T) = (s, \epsilon)$$

Since $S = s$ is a substring of $\mathtt{x}^{\mathtt{tgt}}$, both $(\mathtt{x}^2, \mathtt{y}^2)$ and $(\mathtt{x}^3, \mathtt{y}^3)$ are adaptable to solve $\mathtt{x}^{\mathtt{tgt}}$. This does not hold for $(\mathtt{x}^1, \mathtt{y}^1)$: $S = ve$ is not a substring of $\mathtt{x}^{\mathtt{tgt}}$.

The ranking phase is based on a preference relation between two source cases that are adaptable in a candidate solution of $\mathtt{x}^{\mathtt{tgt}}$. Let $(\mathtt{x}^s, \mathtt{y}^s)$ be a source case that has not been filtered. This involves that the replacements on $\mathtt{x}^s$ to obtain $\mathtt{y}^s$ can be applied on $\mathtt{x}^{\mathtt{tgt}}$ (at one or several place(s)).

For the sake of simplicity of the explanations, let us assume that the transformation from $\mathtt{x}^s$ to $\mathtt{y}^s$ corresponds to a substring replacement of a string S by a string T (e.g., $(\mathtt{x}^s, \mathtt{y}^s)$ is an SR source case) and that S is also a substring of $\mathtt{x}^{\mathtt{tgt}}$ (that is $\mathcal{R}(\mathtt{x}^s, \mathtt{y}^s, \mathtt{x}^{\mathtt{tgt}}, y)$ is solvable). Now, let i^s be the position of the substring S in $\mathtt{x}^s$ such that the replacement S with T in $\mathtt{x}^s$ is made at position i^s and let $i^{\mathtt{tgt}}$ be a position of the substring S in $\mathtt{x}^{\mathtt{tgt}}$ (S may occur as a substring in several positions in $\mathtt{x}^{\mathtt{tgt}}$, so there are potentially several $i^{\mathtt{tgt}}$'s). Case ranking is based on a value $\mathtt{score}(S, \mathtt{x}^s, i^s, \mathtt{x}^{\mathtt{tgt}}, i^{\mathtt{tgt}}) \geq 0$, the higher this value is, the more preferred is the source case $(\mathtt{x}^s, \mathtt{y}^s)$ for adapting $\mathtt{x}^{\mathtt{tgt}}$ by substituting S with T at position i. The definition of this score is based on a general assumption using the notion of *context*.

In the following, this notion is explained, the assumption is presented, the way the score is computed based on this assumption and in the knowledge-light framework is presented and, finally, some future studies are discussed on how it can be defined using linguistic knowledge about French. But first, an example is introduced to illustrate these notions:

$$\mathtt{x}^s = \textit{You do not liked to eat beans.} \qquad \mathtt{y}^s = \textit{You do not like to eat beans.}$$
$$\mathtt{x}^{\mathtt{tgt}} = \textit{We do wanted to go!}$$
$$\text{thus } S = d, \qquad T = \epsilon, \qquad i^s = 15 \qquad \text{and} \qquad i^{\mathtt{tgt}} \in \{3, 11\}$$

(a position i in a string $\mathtt{x}$ being represented by a value $i \in \{0, 1, \ldots, |\mathtt{x}| - 1\}$).

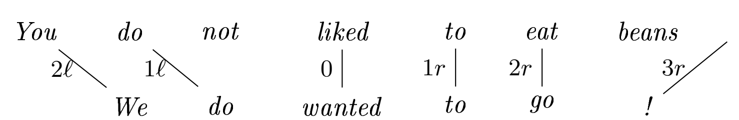
For a sentence $\mathtt{x}$ having a substring S at position i, the *context* of (S, i) in $\mathtt{x}$ gathers pieces of information (characters, words, etc.) "around" S ("the" S at position i). With this vague definition, the whole sentence $\mathtt{x}$ participates to the context of (S, i), but the idea is that pieces of information "close" to S have a greater importance in the context. Following this idea, $\mathtt{score}(S, \mathtt{x}^s, i^s, \mathtt{x}^{\mathtt{tgt}}, i^{\mathtt{tgt}})$ measures the matching between the context of (S, i^s) in $\mathtt{x}^s$ and the context of $(S, i^{\mathtt{tgt}})$ in $\mathtt{x}^{\mathtt{tgt}}$, hence the following general assumption (the underlined terms are the ones that have to be instantiated in an implementation):

> The <u>closer</u> a <u>linguistic entity</u> of $\mathtt{x}^{\mathtt{tgt}}$ is to the substring S at position i, the more its <u>similarity</u> to a matching linguistic entity of $\mathtt{x}^s$ contributes to $\mathtt{score}(S, \mathtt{x}^s, i^s, \mathtt{x}^{\mathtt{tgt}}, i^{\mathtt{tgt}})$.

This assumption is applied as follows for our current knowledge-light approach to retrieval:

- The linguistic entities that are considered are words.
- The closeness between such entities is defined by the number of words that separate the word in which S occurs (0 for this word, 1 for its neighbors in the sentence, 2 for the neighbors of the neighbors, etc.). For example, for $\mathtt{x}^{\mathtt{tgt}}$, the closest word is *wanted*, the second closest words are *do* and *to*, etc.
- The similarity between two linguistic entities is binary: if the two words are equal, their similarity measure is 1, otherwise, it is 0.

Only a short description of the score computing is given here. It is computed on the basis on a best match between the sentences $\mathtt{x}^s$ and $\mathtt{x}^{\mathtt{tgt}}$. Assuming this best match is, for the example (with $i^{\mathtt{tgt}} = 11$):



Then, $\mathtt{score}(S, \mathtt{x}^s, i^s, \mathtt{x}^{\mathtt{tgt}}, i^{\mathtt{tgt}})$ is (with the matching lines indicated below):

$$\underbrace{0 \times \alpha^0}_{0} + \underbrace{1 \times \alpha^1 \times \beta}_{1\ell} + \underbrace{1 \times \alpha^1}_{1r} + \underbrace{0 \times \alpha^2 \times \beta}_{2\ell} + \underbrace{0 \times \alpha^2}_{2r} + \underbrace{0 \times \alpha^3 \times \beta}_{3r}$$

where α is a penalty for the distance to the word containing S and β is a mismatch penalty, when there is a need of words insertions, which corresponds to slanted lines in the matching (in the experiments, $\alpha = \beta = 0.5$). The score is computed for every position $i^{\mathtt{tgt}}$ of S in $\mathtt{x}^{\mathtt{tgt}}$ and its complexity in the worst case is in $\mathcal{O}(\#w(\mathtt{x}^s) \times \#w(\mathtt{x}^{\mathtt{tgt}}))$ where $\#w(\mathtt{x}^s)(\mathtt{x})$ is the number of words in $\mathtt{x}$.

In future studies, the computing of the score may take into account linguistic knowledge:

- Various linguistic entities could be considered, such as word parts, words or groups of words.

- The closeness between such entities could be more accurately defined than the mere proximity in the string. Indeed, syntactical dependency shows that words that are distant in a sentence may have strong connections.
- The similarity between two linguistic entities could be gradual. The cosine similarity between word representations obtained from any pre-trained word embedding model will do it.

A simple modification of the retrieval procedure has been developed after some preliminary tests. It consists simply in extending the substring S to its two neighbor characters. For example, if $\mathtt{x}^s$ = *I does it.* and $\mathtt{y}^s$ = *I do it.*, then $S = oes_\smile$ and $T = o_\smile$ (for the filter and ranking phases). This modification has improved the result of the knowledge-light retrieval procedure presented above and has also highly increased the speed of retrieval, especially for cases for which the replay consists only in adding a substring (i.e., in the previous version, $S = \epsilon$).

7 Evaluation

TFC uses a knowledge-light approach, i.e. working only on strings and without language knowledge. This simple approach can be seen as a baseline for further more sophisticated knowledge-based systems. So, the goal of the evaluation is to establish the baseline.

The experiment consists in solving random problems using a case base. For that, we use an initial CB containing 300 cases to build the set of problems, called the test base and denoted by TB and smallest case bases, denoted by $\mathtt{CB}_n$, where n is the size of the case base. The number of problems to be solved has been fixed to 100 and the problems composing TB have been chosen randomly from CB. Four sizes of case base have been used, with $n \in \{50, 100, 150, 200\}$, to study the impact of the case base in the CBR system. The case bases $\mathtt{CB}_n$ are generated randomly from $\mathtt{CB} \setminus \mathtt{TB}$, with $\mathtt{CB}_{50} \subset \mathtt{CB}_{100} \subset \mathtt{CB}_{150} \subset \mathtt{CB}_{200}$.

TFC is evaluated according to three measures: the answer rate, the answer precision and the correct answer rate. Let ntp be the number of target problems posed to the system, na be the number of (correct or incorrect) answers and nca be the number of correct answers. Answer rate is defined as the average of the ratios na/ntp, the precision is defined as the average of the ratios nca/na, and the correct answer rate is defined as the average of the ratios nca/ntp. The averages of the three measures are computed on 100 runs, one run consisting in solving all the problems of TB using all $\mathtt{CB}_n$.

Table 1 presents the three measures for the different sizes of CB. The results show that, even if all measures increase wrt |CB|, the precision and the correct answer rate remain weak, e.g. a precision of 18.1% means that when the system returns an answer, this answer if false in more than four times out of five, which is not a surprising result. Another expected result is that the answer rate increases with the case base size. With a small case base, the system is only able to solve a few problems (59.2% of them for $\mathtt{CB}_{50}$). The reason is that no similar case can

$\|\mathtt{CB}\|$	50	100	150	200
answer rate	59.2%	80.4%	89.7%	93.8%
precision	15.9%	16.2%	17.2%	18.1%
correct answer rate	9.4%	13.0%	15.4%	17.0%

Table 1. Answer rate, answer precision and correct answer rate for the different sizes of CB.

be found because none of the case of $\mathtt{CB}_n$ addresses the error of $\mathtt{x}^{\mathtt{tgt}}$. By adding more source cases in $\mathtt{CB}_n$, the probability to have, in $\mathtt{CB}_n$, an error *similar* to the one of $\mathtt{x}^{\mathtt{tgt}}$ increases and the system is able to provide more answers.

However, studying the error causes shows that wrong substitutions are applied, coming from a source case (the most similar to $\mathtt{x}^{\mathtt{tgt}}$ from a string point of view) which is not a *good* case for solving $\mathtt{x}^{\mathtt{tgt}}$, i.e. the way the $\mathtt{x}^s$ is corrected into $\mathtt{y}^s$ is not suitable to solve $\mathtt{x}^{\mathtt{tgt}}$. So, the crucial issue is the retrieval process in order to retrieve a source case whose correction is suitable to the context of the target case.

8 Discussion

For many CBR systems, the retrieval phase design precedes the adaptation phase design. Then, this latter has to deal with the retrieved case to solve the target problem. This makes sense in many applications, for which the principle "similar problems have similar solutions" holds, where similarity between problems is defined by some similarity measure (or distance function) suited to the problem representation language and similarity between solutions reflects the easiness of adaptation. This is true in particular when adaptation by copy gives good results, or when adaptation consists in minor adjustments from $\mathtt{y}^s$ to $\mathtt{y}^{\mathtt{tgt}}$.

By contrast, for this first version of TFC, the reverse took place: the adaptation phase was designed before the retrieval phase. Indeed, an adaptation approach at the character level can be easily specified based on the idea of string replacement (which amounts to solve an equation $\mathcal{R}(\mathtt{x}^s, \mathtt{y}^s, \mathtt{x}^{\mathtt{tgt}}, x)$) and then improved thanks to the analogy on strings defined in [4] (which applies when several string replacements at non connected places of the target problem string are needed). Therefore, for TFC, the main issue is how retrieval has to deal with a given adaptation procedure. This can be considered at the light of two previous lines of studies in CBR.

The principle of adaptation-guided retrieval (AGR) already mentioned above is useful here. Indeed, AGR stands that a retrieved case has to be adaptable to solve $\mathtt{x}^{\mathtt{tgt}}$, which corresponds to the filter phase of the retrieval procedure presented in Section 6.

In early research on case-based planning (called *planning by analogy* at that time), the distinction between transformational and generative adaptations has been introduced [2] (actually, they were called "transformational and derivational analogies", but have been renamed in the wider scope of CBR [12]). Transformational adaptation aims at modifying $\mathtt{y}^s$ in $\mathtt{y}^{\mathtt{tgt}}$ on the basis of the differences between $\mathtt{x}^s$ and $\mathtt{x}^{\mathtt{tgt}}$. Generative adaptation consists in analyzing the transformation $\tau : \mathtt{x}^s \mapsto \mathtt{y}^s$ and then in *replaying* τ on $\mathtt{x}^{\mathtt{tgt}}$ (which may involve some modifications if τ is not applicable as such). TFC adaptation is a generative adaptation: τ is given by the string pair (S, T) such that $\mathtt{y}^s$ is obtained by substituting S with T in $\mathtt{x}^s$ and replay consists in making the same substitution on $\mathtt{x}^{\mathtt{tgt}}$.

This S can be linked to the notion of *footprint* of an initial state in a case-based planner such as Prodigy/analogy [11], i.e., the part of the initial state of the planning problem $\mathtt{x}^s$ that "plays" a role in the plan. Therefore, if S denotes also the footprint associated to a case $(\mathtt{x}^s, \mathtt{y}^s)$ in Prodigy/analogy then the condition "The target problem contains S" is, for both systems, a necessary and sufficient condition for the case $(\mathtt{x}^s, \mathtt{y}^s)$ be replayable as such on $\mathtt{x}^{\mathtt{tgt}}$ to get a solution $\mathtt{y}^{\mathtt{tgt}}$ (hence the filter phase of TFC's retrieval). A difference between these systems is that Prodigy/analogy has a complete knowledge for determining whether a solution $\mathtt{y}$ solves a problem $\mathtt{x}$ whereas TFC has not. Thus, in Prodigy/analogy, the above condition entails that $\mathtt{y}^{\mathtt{tgt}}$ is a correct plan whereas in TFC this condition is only a necessary condition for such a correctness, not a sufficient one. This justifies the use of the notion of the context of S in the source and target problems, with the idea that the more the contexts are similar, the more likely the replay gives a correct solution to $\mathtt{x}^{\mathtt{tgt}}$.

The idea of footprints has also been adapted for the system Resyn/CBR which aims at proposing synthesis plans in organic chemistry [6] and uses a hierarchical organization of state substructures (graphs in this application) to speed-up the process: such a hierarchical organization could also be used for TFC for the same purpose, but has not been implemented yet.

9 Conclusion

This paper has presented a challenge for the CBR community: how can correction of sentences be treated by CBR?

A first version of the *The French Correction* has been implemented in order to address this challenge for French sentences. A particularity of this application is that a first version of the adaptation process has appeared to be much simpler to design than the one of retrieval and also that designing adaptation before retrieval has appeared to be the right thing to do. Indeed, the design of retrieval without the knowledge of how the adaptation works has appeared to be a dead-end, hence the necessity of an adaptation-guided approach to retrieval.

Now, the knowledge-light approach to retrieval with a rather small case base that has been implemented gives weak results, which was not unexpected and provides a baseline for future versions. Therefore, two main directions of work for next versions of TFC can be envisaged. The first one consists in obtaining a large case base: this constitutes an ongoing work with the exploitation of the WiCoPaCo collection (cf. Section 4). The second one aims at designing more sophisticated inference engines. For this purpose, it is expected that the use of linguistic knowledge about French will improve the results with respect to the baseline defined in this first version (see Section 6.2). For this research directions, the question raised is what additional cases and additional pieces of linguistic knowledge will have a higher impact on the increase of TFC competence. For this purpose, the research presented in [10], that addresses a different type of case-based sentence modification and uses POS-tagging should be inspiring.

The first application of the analogy on strings defined in [4] is case-based machine translation: a case is a pair $(\mathtt{x}^s, \mathtt{y}^s)$, where $\mathtt{x}^s$ is a sentence in a natural language and $\mathtt{y}^s$ is a translation of $\mathtt{x}^s$ in another natural language. The approach proposed in [4] and studied in [5] at the light of the CBR methodology consists first in finding 3 source cases $(\mathtt{x}^a, \mathtt{y}^a)$, $(\mathtt{x}^b, \mathtt{y}^b)$ and $(\mathtt{x}^c, \mathtt{y}^c)$ such that $\mathtt{x}^a\mathtt{:}\mathtt{x}^b\mathtt{::}\mathtt{x}^c\mathtt{:}\mathtt{x}^{\mathtt{tgt}}$ holds and then in solving the analogical equation $\mathtt{y}^a\mathtt{:}\mathtt{y}^b\mathtt{::}\mathtt{y}^c\mathtt{:}y$: a solution y of this equation is a candidate solution $\mathtt{y}^{\mathtt{tgt}}$ of $\mathtt{x}^{\mathtt{tgt}}$. Now, this idea could be reused for TFC: this would mean that the problems would be sentences in an "incorrect French" language and the solutions would be sentences in a "correct French" language.

References

1. Aamodt, A., Plaza, E.: Case-based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Communications **7**(1) (1994) 39–59
2. 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
3. Dutrey, C., Bouamor, H., Bernhard, D., Max, A.: Local modifications and paraphrases in Wikipedia's revision history. Procesamiento del Lenguaje Natural **46** (2010) 51–58
4. Lepage, Y., Denoual, É.: Purest ever example-based machine translation: detailed presentation and assessment. Machine Translation **19** (2005) 251–282
5. Lepage, Y., Lieber, J.: Case-Based Translation: First Steps from a Knowledge-Light Approach Based on Analogy to a Knowledge-Intensive One. In: ICCBR 2018 - 26th International Conference on Case-Based Reasoning, Stockholm, Sweden (2018)
6. Lieber, J., Napoli, A.: Using Classification in Case-Based Planning. In Wahlster, W., ed.: Proceedings of the 12th European Conference on Artificial Intelligence (ECAI'96), Budapest, Hungary, John Wiley & Sons, Ltd. (1996) 132–136
7. Richter, M.M., Weber, R.O.: Case-based reasoning, a textbook. Springer (2013)
8. Riesbeck, C.K., Schank, R.C.: Inside Case-Based Reasoning. Lawrence Erlbaum Associates, Inc., Hillsdale, New Jersey (1989)
9. Smyth, B., Keane, M.T.: Using adaptation knowledge to retrieve and adapt design cases. Knowledge-Based Systems **9**(2) (1996) 127–135
10. Valls, J., Ontañón, S.: Natural language generation through case-based text modification. In: International Conference on Case-Based Reasoning, Springer (2012) 443–457
11. Veloso, M.M.: Planning and Learning by Analogical Reasoning. LNAI 886. Springer Verlag, Berlin (1994)
12. Wilke, W., Bergmann, R.: Techniques and Knowledge used for Adaptation during Case-Based Problem Solving. In: Proc. of the 11th Int. Conf. on Industrial, Engineering and Other Applications of Applied Intelligent Systems. (1998)