Strong, Fuzzy and Smooth Hierarchical Classification for Case-Based Problem Solving

Jean Lieber

Strong, Fuzzy and Smooth Hierarchical Classification for Case-Based Problem Solving

Jean Lieber[1]

Abstract. This paper explains how case-based problem solving can have benefit from a hierarchical organisation of problems based on a generality relation. Three adaptation-guided retrieval processes are described. The strong classification in a problem hierarchy is a classical deductive process. It is based on the generality relation between problems which organises the hierarchy. The fuzzy classification is a fuzzification of the strong classification. It is based on a fuzzy generality relation between problems, which can be seen as a non-symmetrical similarity measure. The smooth classification extends the fuzzy classification: it is also based on a similarity or dissimilarity measure but takes into account problem and solution adaptation knowledge. These processes have been successfully implemented in two case-based reasoning systems: RESYN/CBR in the domain of organic synthesis and KASIMIR/CBR in the domain of cancer treatment.

1 INTRODUCTION

Case-based reasoning (CBR) is a type of reasoning based on the reuse of past experiences called cases [11]. A case is usually given by a problem and its solution. This paper explains how one can have benefit of a hierarchical organisation of problems based on a generality relation to perform case-based problem solving. Three processes of retrieval based on classification are presented.

The first process, the strong hierarchical classification, is a classical deductive process. It is based on the generality relation between problems which organises the hierarchy. The second process is the fuzzy hierarchical classification, which is a fuzzification of the previous one. It is based on a fuzzy relation obtained by a fuzzification of the generality relation between problems. This fuzzy relation can be seen as a non-symmetrical similarity measure. Finally, the third process is an extension of the second process and is called smooth hierarchical classification. Its main algorithm is close to the one of fuzzy hierarchical classification, but uses a similarity or dissimilarity measure based on knowledge about problem and solution adaptations.

These retrieval processes are adaptation-guided: the retrieved source case is ensured to be adaptable to the target problem. They have been successfully implemented in two CBR systems. RESYN/CBR is a case-based planner dedicated to organic chemistry synthesis and uses the strong and smooth hierarchical classification processes [6]. KASIMIR/CBR is a breast cancer treatment decision support system based on CBR principles; it uses strong and fuzzy classification processes and is intended to use smooth classification [5].

[1] LORIA (CNRS, INRIA, Nancy Universities), BP 239, 54 506 Vandœuvre-lès-Nancy, France, email: lieber@loria.fr

Section 2 presents some notions about CBR and describes the application domains of RESYN/CBR and KASIMIR/CBR. Section 3 shows how problems can be hierarchically organised. The strong, fuzzy and smooth hierarchical classification processes are described in sections 4, 5 and 6. A discussion and a conclusion end the paper (sections 7 and 8).

2 CASE-BASED PROBLEM SOLVING

This section presents some notions about CBR in the context of problem solving. These notions are exemplified in two applications: RESYN/CBR, a case-based planner dedicated to organic chemistry synthesis [6] and KASIMIR/CBR, a breast cancer treatment decision support system based on CBR principles [5].

Problems and Solutions. The nature of problems and solutions depends on the application domain. A problem is often composed of two (explicit or not) parts: a *context* and a *question* (similar to the *initial state* and the *goal statement* in planning, see e.g. [4]). Then, a solution is an answer to the question in the given context. For example, in RESYN/CBR, a problem context is a molecular graph m and the problem question (the same for each problem in this application) is "How could m be chemically synthesised?" A RESYN/CBR solution is a synthesis plan. In KASIMIR/CBR, a problem context is the description of the patient state (age, size and localisation of the tumour, etc.). This description can be incomplete (imprecise or even missing values for some attributes). A problem question is given by a set of treatment categories, for instances: "What is the proposed *chemotherapy* for this patient?", "What are the proposed *surgery* and *radiotherapy* for this patient?", "What are the proposed treatments for this patient?" (the last one corresponds to the set of all the treatment categories). A KASIMIR/CBR solution is a treatment proposition. Although the problem representations of the two systems are very different –non-directed graphs for the former and sets of attribute-value pairs for the latter– they share a same approach to case-based problem solving.

Case-Based Problem Solving. A *case* is a pair $(\texttt{pb}, \texttt{Sol}(\texttt{pb}))$ where $\texttt{pb}$ is a problem and $\texttt{Sol}(\texttt{pb})$ is a solution of $\texttt{pb}$. Solving a problem means associating a licit solution with it. A CBR system aims at solving a *target problem* denoted by $\texttt{tgt}$ with the help of a *case base* which is a finite set of cases. A case from the case base is called a *source case* and is denoted by $\texttt{srce-case} = (\texttt{srce}, \texttt{Sol}(\texttt{srce}))$. A *source problem* $\texttt{srce}$ is a problem such that $(\texttt{srce}, \texttt{Sol}(\texttt{srce}))$ belongs to the case base, i.e. it is a problem for which a solution $\texttt{Sol}(\texttt{srce})$ is known. In general, a CBR session is composed of three main phases: retrieval, adaptation and storage.

The goal of retrieval is to find a case $\texttt{srce-case}$ in the case base that is considered to be similar to $\texttt{tgt}$. Adaptation uses this retrieved case $\texttt{srce-case}$ in order to give a solution $\texttt{Sol}(\texttt{tgt})$ to $\texttt{tgt}$. The new case $(\texttt{tgt}, \texttt{Sol}(\texttt{tgt}))$ is stored in the case base if this storage is appropriate. Only the retrieval and adaptation phases are studied in this paper.

Adaptation-Guided Retrieval. The aim of retrieval is to provide a source case to be adapted to solve the target problem $\texttt{tgt}$. A retrieval procedure is said to be adaptation-guided when any source case it returns is necessarily adaptable to solve $\texttt{tgt}$ [13]. Such a property requires a tight link between adaptation knowledge and retrieval knowledge.

3 PROBLEM HIERARCHY

In description logics, the notion of *concept* is central [9]. It is similar to the notion of *class* in object-based representation systems (see, e.g. [8]). A concept (or a class) denotes a (possibly infinite) set of individuals. The concepts are organised thanks to a *subsumption* relation $\sqsubseteq$ such that $\texttt{C}_1 \sqsubseteq \texttt{C}_2$ iff any individual denoted by $\texttt{C}_1$ is denoted by $\texttt{C}_2$. Two concepts $\texttt{C}_1$ and $\texttt{C}_2$ are equivalent $-\texttt{C}_1 \equiv \texttt{C}_2-$ if $\texttt{C}_1 \sqsubseteq \texttt{C}_2$ and $\texttt{C}_2 \sqsubseteq \texttt{C}_1$. If the concepts are considered modulo $\equiv$ (i.e., two concepts denoting the same set of individuals are equal), then $\sqsubseteq$ is an order relation. Let $\omega_{\texttt{C}}$ be the maximum for $\sqsubseteq$ (any individual of the domain is denoted by $\omega_{\texttt{C}}$). A finite set $\{\omega_{\texttt{C}}, \texttt{C}_1, \ldots, \texttt{C}_n\}$ can be organised in a hierarchy $\mathcal{H}_{\texttt{C}}$ for $\sqsubseteq$. This involves that $\omega_{\texttt{C}}$ is the root of $\mathcal{H}_{\texttt{C}}$ and that, for each $\texttt{C}_i$ and $\texttt{C}_j$, $\texttt{C}_i \sqsubseteq \texttt{C}_j$ iff there is a path from $\texttt{C}_i$ to $\texttt{C}_j$ in $\mathcal{H}_{\texttt{C}}$. The hierarchy $\mathcal{H}_{\texttt{C}}$ facilitates the access to the concepts.

This section describes how problems can be organised in a similar way, taking into account the main difference between problems and concepts: a concept is related to the individuals it denotes whereas a problem can be apprehended by the solutions that solve it.

Generality Relation Between Problems. In the application domain considered, it is assumed that an order relation between problems $\preccurlyeq$ is given with the following property:

if $\texttt{pb}_1 \succcurlyeq \texttt{pb}_2$

then every solution of $\texttt{pb}_1$ is[2] a solution of $\texttt{pb}_2$ (1)

For example, let us consider the two following problems in an everyday life domain:

$$\texttt{pb}_1 = \left| \begin{array}{l} \text{context} = \text{My car does not work.} \\ \text{question} = \text{What can I do?} \end{array} \right.$$

$$\texttt{pb}_2 = \left| \begin{array}{l} \text{context} = \text{My car does not work. Its petrol tank is empty.} \\ \text{question} = \text{What can I do?} \end{array} \right.$$

It is consistent with (1) to assert that $\texttt{pb}_1 \succcurlyeq \texttt{pb}_2$. For instance, the solution $\texttt{Sol}(\texttt{pb}_1)$ = "Buy a new car" of $\texttt{pb}_1$ is also a solution of $\texttt{pb}_2$ (not a good one, but a solution yet).

For the problems described by a context and a question, $\texttt{pb}_1 \succcurlyeq \texttt{pb}_2$ if the context of $\texttt{pb}_1$ is more general than the context of $\texttt{pb}_2$ and if the question of $\texttt{pb}_1$ "contains" the question of $\texttt{pb}_2$. In RESYN/CBR, $\texttt{pb}_1 \succcurlyeq \texttt{pb}_2$ if the molecular graph m_1 is a substructure of the molecular graph m_2 (i.e., there is a partial subgraph isomorphism from m_1 to m_2), m_1 and m_2 being the respective contexts of $\texttt{pb}_1$ and $\texttt{pb}_2$. Indeed, a solution of $\texttt{pb}_1$ explains how the structure m_1 can be built and therefore how the part of m_2 isomorphic to m_1 can be built, providing a (partial) solution to $\texttt{pb}_2$. In KASIMIR/CBR, $\texttt{pb}_1 \succcurlyeq \texttt{pb}_2$ if the patient description of $\texttt{pb}_2$ is more precise than the one of $\texttt{pb}_1$ and the set of treatment categories required in $\texttt{pb}_1$ contains the one in $\texttt{pb}_2$.

$\preccurlyeq$ is called the *generality relation between problems*. Let $\omega_{\texttt{pb}}$ be the maximum for $\preccurlyeq$. It is the generic problem of the application domain. In RESYN/CBR, the meaning of $\omega_{\texttt{pb}}$ is "How can a molecule be synthesised?" In KASIMIR/CBR, it is "What treatment can be given to a patient suffering from breast cancer?" Usually, no explicit solution is known for $\omega_{\texttt{pb}}$.

Hierarchical Organisation of Problems.

Let $\{\omega_{\texttt{pb}}, \texttt{pb}_1, \ldots, \texttt{pb}_n\}$ be a set of problems. This set can be organised in a hierarchy $\mathcal{H}_{\texttt{pb}}$ for $\preccurlyeq$, with root $\omega_{\texttt{pb}}$. This means that $(\texttt{pb}_i, \texttt{pb}_j)$ is an edge of $\mathcal{H}_{\texttt{pb}}$ iff $\texttt{pb}_i \preccurlyeq \texttt{pb}_j$, $\texttt{pb}_i \neq \texttt{pb}_j$ and there is no path from $\texttt{pb}_i$ to $\texttt{pb}_j$ of length $\ell \geq 2$. The algorithm of construction of $\mathcal{H}_{\texttt{pb}}$ is identical to the one of a hierarchy of concepts $\mathcal{H}_{\texttt{C}}$ which is described in [1]. The source problems are assumed to belong to $\mathcal{H}_{\texttt{pb}}$. These problems $\texttt{srce}$ are associated with their solutions $\texttt{Sol}(\texttt{srce})$ thanks to pointers. The other problems of $\mathcal{H}_{\texttt{pb}}$ are *abstract* problems: no solution is associated with them and their role is to better structure the hierarchy. The figure 1 illustrates that.

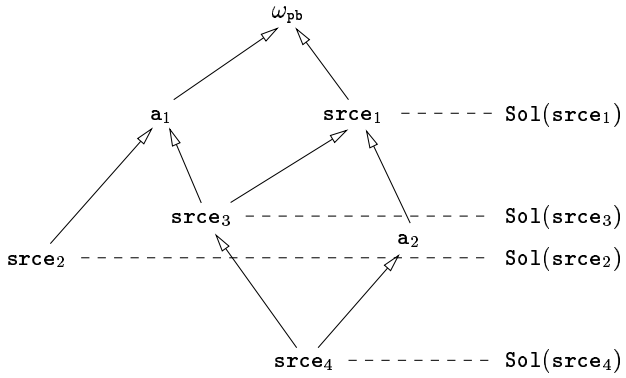

Figure 1. A problem hierarchy $\mathcal{H}_{\texttt{pb}}$.

Ossified Cases and Indexing issue. The book [11] distinguishes the *stories* –description of actual events– and the *ossified cases* – general cases, similar to rules. In KASIMIR/CBR, the source cases are ossified: they are rules of a medical guideline in cancerology but are used in a CBR manner, i.e. they are adapted to solve the target problem. Conversely, RESYN/CBR source cases are stories of actual synthesis, thus $\texttt{srce}$ are specific problems which are not very helpful to structure $\mathcal{H}_{\texttt{pb}}$ and thus to make the retrieval easier. This is why, RESYN/CBR source cases $(\texttt{srce}, \texttt{Sol}(\texttt{srce}))$ are *indexed*. This means that, for retrieval purpose, the problem $\texttt{srce}$ is substituted in $\mathcal{H}_{\texttt{pb}}$ by a problem $\texttt{idx}(\texttt{srce})$ which is a generalisation of $\texttt{srce}$ $-\texttt{srce} \preccurlyeq \texttt{idx}(\texttt{srce})-$ such that the solution $\texttt{Sol}(\texttt{srce})$ of $\texttt{srce}$ is also a solution of $\texttt{idx}(\texttt{srce})$. One could say that the case $(\texttt{srce}, \texttt{Sol}(\texttt{srce}))$ has been ossified in the case $(\texttt{idx}(\texttt{srce}), \texttt{Sol}(\texttt{srce}))$. In the following of the paper, source cases are assumed to be ossified.

[2] (1) can be relaxed by replacing "is" by "can be specialised in" if a solution specialisation procedure is available. This holds for RESYN/CBR.

4 STRONG CLASSIFICATION

The strong classification process in the hierarchy $\mathcal{H}_{\mathtt{pb}}$ consists in searching the source problems $\mathtt{srce}$ that are more general than $\mathtt{tgt}$: $\mathtt{srce} \succcurlyeq \mathtt{tgt}$. If there are such source problems $\mathtt{srce}$, the retrieval chooses one with some preference criterion (e.g., the lowest ones in $\mathcal{H}_{\mathtt{pb}}$ are often preferred) and returns the case $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}))$ to the adaptation module. The adaptation consists simply in copying $\mathtt{Sol}(\mathtt{srce})$ in a solution $\mathtt{Sol}(\mathtt{tgt})$ of $\mathtt{tgt}$, having benefit of the property (1) defined in section 3. This deductive reasoning can be written in an inference rule similar to the *modus ponens*:

$$\frac{\mathtt{tgt} \quad \mathtt{srce} \succcurlyeq \mathtt{tgt} \quad \mathtt{Sol}(\mathtt{srce}) \text{ is a solution of } \mathtt{srce}}{\mathtt{Sol}(\mathtt{tgt}) = \mathtt{Sol}(\mathtt{srce}) \text{ is a solution of } \mathtt{tgt}}$$

From an algorithmic viewpoint, strong hierarchical classification is similar to classification in a hierarchy of concepts (see [1] for efficient algorithms). It can be performed by a depth-first search in $\mathcal{H}_{\mathtt{pb}}$ taking into account the following property: if $\mathtt{pb} \not\succcurlyeq \mathtt{tgt}$, then no problem of $\mathcal{H}_{\mathtt{pb}}$ more specific than $\mathtt{pb}$ can be more general than $\mathtt{tgt}$ (since $\preccurlyeq$ is transitive). Therefore, if a node $\mathtt{pb}$ of $\mathcal{H}_{\mathtt{pb}}$ fails to the test "$\mathtt{pb} \succcurlyeq \mathtt{tgt}$", then the sub-hierarchy of root $\mathtt{pb}$ can be pruned away from the search. This pruning can lead from a linear complexity for a flat organisation of cases to a logarithmic complexity for some structures of the hierarchy (e.g., if $\mathcal{H}_{\mathtt{pb}}$ is a well-balanced decisional tree[3]).

Unfortunately, it may occur that no source problem is more general than the current target problem. The idea is then to relax the condition $\mathtt{srce} \succcurlyeq \mathtt{tgt}$. A first way to do it is to fuzzify the relation $\succcurlyeq$, which requires another retrieval algorithm.

5 FUZZY CLASSIFICATION

The strong classification and the adaptation by copy that follows it form a deductive process in the classical logic, i.e. the logic with two truth values: 0 (or *false*) and 1 (or *true*). Fuzzifying consists in extending some notions defined in classical logic (e.g. set, relation, logical connectives) in a fuzzy logic, i.e. a logic with truth values in the interval $[0, 1]$ (see, e.g. [12]). Such a logic enables to represent imprecision, uncertainty or vagueness. This section presents a way to fuzzify the strong classification.

5.1 Fuzzification of $\succcurlyeq$ by a similarity measure $\mathcal{S}$

The generality relation between problems, $\succcurlyeq$, can be fuzzified in a fuzzy relation $\mathcal{S}$: the value of $\mathtt{pb}_1 \succcurlyeq \mathtt{pb}_2$ is either true or false, whereas the value of $\mathcal{S}(\mathtt{pb}_1, \mathtt{pb}_2)$ is a fuzzy truth value. $\succcurlyeq$ and $\mathcal{S}$ are related by the following property, for each problems $\mathtt{pb}_1$ and $\mathtt{pb}_2$:

$$\mathcal{S}(\mathtt{pb}_1, \mathtt{pb}_2) = 1 \quad \Leftrightarrow \quad \mathtt{pb}_1 \succcurlyeq \mathtt{pb}_2 \tag{2}$$

Thus, $\mathcal{S}$ is reflexive: $\mathcal{S}(\mathtt{pb}, \mathtt{pb}) = 1$, for each problem $\mathtt{pb}$. It is assumed furthermore that $\mathcal{S}$ is non-symmetrical (but not necessarily anti-symmetrical) and max-min transitive, which means that for each problems $\mathtt{pb}_1$, $\mathtt{pb}_2$ and $\mathtt{pb}_3$:

$$\mathcal{S}(\mathtt{pb}_1, \mathtt{pb}_3) \geq \min(\mathcal{S}(\mathtt{pb}_1, \mathtt{pb}_2), \mathcal{S}(\mathtt{pb}_2, \mathtt{pb}_3)) \tag{3}$$

Finally, since $\mathcal{S}$ is a fuzzification of $\succcurlyeq$, the property (1) relating $\succcurlyeq$ with the link between problems and solutions have to be fuzzified.

[3] $\mathcal{H}_{\mathtt{pb}}$ is a decisional tree if for each $\mathtt{pb}_1$ and $\mathtt{pb}_2$ of $\mathcal{H}_{\mathtt{pb}}$ such that neither $\mathtt{pb}_1 \succcurlyeq \mathtt{pb}_2$ nor $\mathtt{pb}_2 \succcurlyeq \mathtt{pb}_1$, there is no problem $\mathtt{pb}$ –in or outside $\mathcal{H}_{\mathtt{pb}}$– being more specifi c than $\mathtt{pb}_1$ and $\mathtt{pb}_2$ at one and the same time.

The classical link "is a solution of" between a problem and a solution is binary: a solution $\mathtt{Sol}(\mathtt{pb})$ either solves or does not solve a problem $\mathtt{pb}$. This link can be fuzzified: $\mathtt{Sol}(\mathtt{pb})$ solves $\mathtt{pb}$ with a truth value of $v \in [0, 1]$. v measures the confidence or the precision of $\mathtt{Sol}(\mathtt{pb})$ wrt $\mathtt{pb}$. $\mathtt{Sol}(\mathtt{pb})$ is said to be an s-solution of $\mathtt{pb}$ if $v \geq s$, with v, the truth value of "$\mathtt{Sol}(\mathtt{pb})$ solves $\mathtt{pb}$". It is assumed that for each case $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}))$ of the case base, $\mathtt{Sol}(\mathtt{srce})$ is a 1-solution of $\mathtt{srce}$ (the "1-" is omitted in the following). The removal of this assumption leads to a slightly more complex approach. The following property of $\mathcal{S}$, fuzzification of (1), is assumed:

$$\begin{aligned} &\textbf{if } \mathcal{S}(\mathtt{pb}_1, \mathtt{pb}_2) = s \\ &\quad \textbf{then} \text{ every solution of } \mathtt{pb}_1 \text{ is an } s\text{-solution of } \mathtt{pb}_2 \end{aligned} \tag{4}$$

$\mathcal{S}$ is said to be a non-symmetrical measure of the similarity between a source problem $\mathtt{srce}$ and a target problem $\mathtt{tgt}$ since it measures how $\mathtt{srce}$ "includes" $\mathtt{tgt}$.

For KASIMIR/CBR, the similarity measure $\mathcal{S}$ is defined thanks to local measures $\mathtt{m}_{\mathtt{pb}}$ defined at the level of each problem $\mathtt{pb}$ of $\mathcal{H}_{\mathtt{pb}}$:

$$\mathcal{S}(\mathtt{pb}, \mathtt{tgt}) = \mathtt{m}_{\mathtt{pb}}(\mathtt{tgt})$$

This fuzzification only affects some attributes of the problem contexts (and not the problem questions) and reflects the uncertainty of the treatment propositions $\mathtt{Sol}(\mathtt{srce})$ associated with the source problems $\mathtt{srce}$ (abstract problems are also fuzzified, so that the fuzzification of problems in $\mathcal{H}_{\mathtt{pb}}$ is consistent with (2)). This enables to take into account the threshold effect. For example, let $\mathtt{srce}_1$ and $\mathtt{srce}_2$ be two source problems such that their contexts contain the respective conditions $\mathtt{age} \leq 65$ and $\mathtt{age} > 65$, the other conditions being equal and the solutions $\mathtt{Sol}(\mathtt{pb}_1)$ and $\mathtt{Sol}(\mathtt{pb}_2)$ being different. For a patient of age 66, proposing only the treatment $\mathtt{Sol}(\mathtt{pb}_2)$ is doubtful: if the patient is "young for her age", the solution $\mathtt{Sol}(\mathtt{pb}_1)$ might be better. That is why, these conditions are fuzzified so that the two propositions $\mathtt{Sol}(\mathtt{pb}_1)$ and $\mathtt{Sol}(\mathtt{pb}_2)$ are proposed.

5.2 Fuzzy classification principle

The following inference rule, based on (4), can be proposed:

$$\frac{\mathtt{tgt} \quad \mathcal{S}(\mathtt{srce}, \mathtt{tgt}) = s \quad \mathtt{Sol}(\mathtt{srce}) \text{ is a solution of } \mathtt{srce}}{\mathtt{Sol}(\mathtt{tgt}) = \mathtt{Sol}(\mathtt{srce}) \text{ is an } s\text{-solution of } \mathtt{tgt}}$$

It indicates how the source cases can provide approximate solutions to $\mathtt{tgt}$. Furthermore, the values s enable to rank the retrieved cases. The adaptation process is a solution copy.

5.3 A fuzzy classification algorithm

The fuzzy classification algorithm in the hierarchy $\mathcal{H}_{\mathtt{pb}}$ is a search of the source cases by decreasing similarity to the target problem.

The algorithm of figure 2 is a fuzzy classification algorithm based on a best-first search [10] according to the similarity to the target problem in the hierarchy $\mathcal{H}_{\mathtt{pb}}$. This algorithm returns a sole source problem that is the closest one to the target problem. It can easily be modified in order to return a list of source problems $\mathtt{srce}$ ordered by decreasing $\mathcal{S}(\mathtt{srce}, \mathtt{tgt})$. This would require another term in the condition of line 3 replacing "$\mathtt{pb}$ is not a source problem". For examples, this term can be "$\mathcal{S}(\mathtt{pb}, \mathtt{tgt}) > 0$" (the retrieval would return *all* the source problems similar to the target problem), "$\mathcal{S}(\mathtt{pb}, \mathtt{tgt}) \geq \alpha$" (where α is a similarity threshold) or "$\mathtt{nsp} < k$" (where $\mathtt{nsp}$ is the number of source problems in $\mathtt{SP}$, if the k nearest neighbours of $\mathtt{tgt}$ are requested).

fuzzy hierarchical classification

Input: • a target problem tgt.
• a problem hierarchy $\mathcal{H}_{\mathtt{pb}}$, of root $\omega_{\mathtt{pb}}$.
Output: the closest source problem to tgt according to $\mathcal{S}$ (or failure, if no source problem is similar to tgt).

begin (algorithm)
1. SP $\leftarrow \{\omega_{\mathtt{pb}}\}$ /* SP: *a set of problems of* $\mathcal{H}_{\mathtt{pb}}$ */
2. pb $\leftarrow \omega_{\mathtt{pb}}$ /* pb: *the current problem* */
3. **while** pb is not a source problem **and** SP $\neq \emptyset$
4. SP $\leftarrow$ (SP $-$ {pb}) $\cup$ desc, where desc is the set of the direct descendants of pb in $\mathcal{H}_{\mathtt{pb}}$.
5. Remove from SP the non-maximum elements for $\succcurlyeq$.[4]
6. pb $\leftarrow$ the problem in SP maximising $\mathcal{S}(\mathtt{pb}, \mathtt{tgt})$.
7. **end** (while)
8. **if** pb is a source problem
9. **then return** pb
10. **else return** failure

end (algorithm)

Figure 2. A fuzzy hierarchical classification algorithm.

If the complexity of $\mathcal{S}$ is big compared to the complexity of $\succcurlyeq$, then another change in the algorithm should be made. This change consists in doing first a strong classification that provides the source problems srce such that $\mathcal{S}(\mathtt{srce}, \mathtt{tgt}) = 1$ (cf. property (2)) but also the set $\mathtt{SP}_0$ of problems defined as follows:

$$\text{With } \mathtt{NMGT} = \{\mathtt{pb} \mid \mathtt{pb} \text{ is a problem of } \mathcal{H}_{\mathtt{pb}} \text{ and } \mathtt{pb} \not\succcurlyeq \mathtt{tgt}\}$$

$$\mathtt{SP}_0 \text{ is the set of maximums of NMGT for } \succcurlyeq, \text{ i.e.,}$$

$$\mathtt{SP}_0 = \{\mathtt{pb} \mid \mathtt{pb} \in \mathtt{NMGT} \text{ and } \forall \mathtt{pb}' \in \mathtt{NMGT}, \mathtt{pb}' \succcurlyeq \mathtt{pb} \Rightarrow \mathtt{pb}' = \mathtt{pb}\}$$

This set $\mathtt{SP}_0$ can be used to initialise the set SP in the algorithm of figure 2, instead of $\{\omega_{\mathtt{pb}}\}$, and thus, a part of the hierarchy would be already searched by strong classification before the execution of the fuzzy classification, which would save a part of the execution time.

5.4 Properties of the fuzzy classification

The fuzzy classification is an extension of the strong classification in the sense that if $\mathcal{S}$ were defined by

$$\mathcal{S}(\mathtt{pb}_1, \mathtt{pb}_2) = \textbf{if } \mathtt{pb}_1 \succcurlyeq \mathtt{pb}_2 \textbf{ then } 1 \textbf{ else } 0$$

and if the fuzzy classification returns all the source cases similar to tgt (term "$\mathcal{S}(\mathtt{pb}, \mathtt{tgt}) > 0$" at the line 3 of the algorithm), then the fuzzy classification would return the same result as the strong classification.

The fuzzy classification algorithm considers problems of $\mathcal{H}_{\mathtt{pb}}$ by decreasing similarity to tgt. If $\mathtt{pb}_1$ is considered before $\mathtt{pb}_2$ during the process then $\mathcal{S}(\mathtt{pb}_1, \mathtt{tgt}) \geq \mathcal{S}(\mathtt{pb}_2, \mathtt{tgt})$. Therefore, if pb is the current problem then, for each problem in the hierarchy that has not been treated yet, this problem is less similar to tgt than pb. Thus, if $\mathcal{S}(\mathtt{pb}, \mathtt{tgt})$ is considered to be too low, it is useless to keep on the fuzzy classification process.

[4] If $\mathtt{pb}_1, \mathtt{pb}_2 \in \mathtt{SP}$ are such that $\mathtt{pb}_1 \succcurlyeq \mathtt{pb}_2$ and $\mathtt{pb}_1 \neq \mathtt{pb}_2$, then $\mathtt{pb}_2$ must be removed from SP. If $\mathcal{H}_{\mathtt{pb}}$ is a tree, this situation cannot occur and this line of the algorithm is useless.

This property is a consequence of the similarity to tgt decreasing when pb becomes more specific in $\mathcal{H}_{\mathtt{pb}}$:

$$\text{for } \mathtt{pb}_1 \text{ and } \mathtt{pb}_2, \text{ two problems of } \mathcal{H}_{\mathtt{pb}},$$
$$\mathtt{pb}_1 \succcurlyeq \mathtt{pb}_2 \quad \Rightarrow \quad \mathcal{S}(\mathtt{pb}_1, \mathtt{tgt}) \geq \mathcal{S}(\mathtt{pb}_2, \mathtt{tgt}) \qquad (5)$$

(5) is a consequence of (2) and (3).

This property involves that the fuzzy classification complexity is not very sensitive to the source problems dissimilar to tgt: if the hierarchy is well-structured, most of the problems which are far away from tgt are not taken into account during the process.

6 SMOOTH CLASSIFICATION

In the previous sections, CBR was considered with a "null adaptation" [11], doing a simple copy of the retrieved source case. Many CBR systems, however, take advantage of more sophisticated adaptation processes. This holds for RESYN/CBR and must hold for further versions of KASIMIR/CBR. For a non-adaptation-guided retrieval based on a similarity measure $\mathcal{S}$, the algorithm of the previous section can be directly reused. For an adaptation-guided retrieval, srce and tgt are considered to be similar if there is some available adaptation knowledge to adapt $\mathtt{Sol}(\mathtt{srce})$ in a solution $\mathtt{Sol}(\mathtt{tgt})$ of tgt.

Smooth classification is based on $\mathtt{sim}(\mathtt{pb}, \mathtt{tgt})$, an explicit representation of how a solution of the current problem pb could be adapted to solve tgt. $\mathtt{sim}(\mathtt{pb}, \mathtt{tgt})$ is a *similarity path*, i.e. a sequence of relations

$$\mathtt{pb} = \mathtt{pb}_0\ \mathtt{r}_1\ \mathtt{pb}_1\ \mathtt{r}_2\ \mathtt{pb}_2 \ldots \mathtt{pb}_{q-1}\ \mathtt{r}_q\ \mathtt{pb}_q = \mathtt{tgt}$$

such that an adaptation function $\mathcal{A}_{\mathtt{r}_i}$ is associated with each $\mathtt{r}_i$ and enables to adapt a solution of the problem $\mathtt{pb}_{i-1}$ in a solution of the problem $\mathtt{pb}_i$. The ordered pair $(\mathtt{r}_i, \mathcal{A}_{\mathtt{r}_i})$ is called a *reformulation* and the reformulations constitute the available adaptation knowledge [7]. For instance, $(\succcurlyeq, \mathcal{A}_{\succcurlyeq})$, where $\mathcal{A}_{\succcurlyeq}$ is a solution copy, is a reformulation based on property (1). If the current problem is a source problem srce, the adaptation is made by 1°/ adaptation of $\mathtt{Sol}(\mathtt{srce}) = \mathtt{Sol}(\mathtt{pb}_0)$ in a solution $\mathtt{Sol}(\mathtt{pb}_1)$ of $\mathtt{pb}_1$, thanks to $\mathcal{A}_{\mathtt{r}_1}$, 2°/ adaptation of $\mathtt{Sol}(\mathtt{pb}_1)$ in a solution $\mathtt{Sol}(\mathtt{pb}_2)$ of $\mathtt{pb}_2$, thanks to $\mathcal{A}_{\mathtt{r}_2}$, $\ldots q^\circ$/ adaptation of $\mathtt{Sol}(\mathtt{pb}_{q-1})$ in a solution $\mathtt{Sol}(\mathtt{pb}_q) = \mathtt{Sol}(\mathtt{tgt})$ of $\mathtt{pb}_q = \mathtt{tgt}$, thanks to $\mathcal{A}_{\mathtt{r}_q}$. This composition of simple adaptations is denoted by $\mathcal{A}_{\mathtt{r}_1;\mathtt{r}_2;\ldots\mathtt{r}_q}$, $\mathtt{r}_1;\mathtt{r}_2;\ldots\mathtt{r}_q$ being the composition of the $\mathtt{r}_i$'s. The search of a similarity path is made in RESYN/CBR thanks to an A* search [10].

Let d and $\mathcal{S}$ be defined by $d(\mathtt{pb}_1, \mathtt{pb}_2)$ is the weighted length of the shortest similarity path from $\mathtt{pb}_1$ to $\mathtt{pb}_2$ ($+\infty$ if there is no such path) and $\mathcal{S}(\mathtt{pb}_1, \mathtt{pb}_2) = 1/(1 + d(\mathtt{pb}_1, \mathtt{pb}_2))$. It is assumed that the shorter a similarity path from srce to tgt is, the better (more certain or more precise) the solution $\mathtt{Sol}(\mathtt{tgt})$ obtained by an adaptation along this similarity path is. This involves that the weight associated with $\succcurlyeq$ in a similarity path is 0. The fuzzy classification algorithm can be easily modified to return an ordered pair $(\mathtt{pb}, \mathtt{sim}(\mathtt{pb}, \mathtt{tgt}))$ where pb is a source problem: computing $\mathcal{S}(\mathtt{pb}, \mathtt{tgt})$ and searching a similarity path $\mathtt{sim}(\mathtt{pb}, \mathtt{tgt})$ are done in the same process. The following inference rule summarises the smooth classification:

$$\frac{\mathtt{tgt} \quad \mathcal{S}(\mathtt{srce}, \mathtt{tgt}) = s \quad \mathtt{srce}\ \mathtt{r}_1;\mathtt{r}_2;\ldots\mathtt{r}_q\ \mathtt{tgt} \quad \mathtt{Sol}(\mathtt{srce}) \text{ is a solution of } \mathtt{srce}}{\mathtt{Sol}(\mathtt{tgt}) = \mathcal{A}_{\mathtt{r}_1;\mathtt{r}_2;\ldots\mathtt{r}_q}(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}), \mathtt{tgt}) \text{ is an } s\text{-solution of } \mathtt{tgt}}$$

If the property (5) holds for $\mathcal{S}$, the problems of $\mathcal{H}_{\mathtt{pb}}$ are still considered with a decreasing similarity to tgt. This is the case for

RESYN/CBR in particular thanks to the use of the reformulation $(\succcurlyeq, \mathcal{A}_{\succcurlyeq})$. It involves that the deeper the current problem gets in $\mathcal{H}_{\mathtt{pb}}$, the longer the similarity path is and the worse the final solution will be. In other words, the source cases that will provide the best solutions to $\mathtt{tgt}$ (more sure or more precise) are discovered first.

7 DISCUSSION AND RELATED WORKS

This paper presents an approach to problem-solving based on hierarchical classification, fuzzy logic and CBR. A selection of other works based on some of these notions is discussed in this section.

A hierarchical classification in a fuzzy object-based representation is studied in [3] where the representations and inferences of object-based representation systems are fuzzified (fuzzy ranges and fuzzy typical ranges for the attributes, weighted hierarchy of classes, inheritance and classification in such a hierarchy, etc.). Two main differences with the approach presented here must be noted. First, the entities manipulated are classes and not problems, which involves a difference on the points of view of the two papers. Second, the representation in this work is central whereas we have presented an approach that is independent of the representation (RESYN/CBR and KASIMIR/CBR are based on different representation formalisms but share some principles).

Hierarchical classification and CBR are combined in [4], in which an approach to case retrieval based on classification in a description logic is presented. Two processes are described. Strong classification is similar to the strong classification presented here, whereas weak classification can be seen as a hierarchical classification with a relaxed comparison between problems (the initial state *or* the goal statement of the source planning problem has to match, and not necessarily both). The main difference between the relaxation of weak classification and the ones of fuzzy and smooth classification is that the former is done *a priori* while the latters are done only when necessary and are better-suited for the current pair of problems being compared. This involves on the one hand that the retrieval of the current paper is more accurate and is adaptation-guided and, on the other hand, that it is more time-consuming.

Fuzzy logic is used for CBR in [2]. The so-called CBR principle –similarity of problems entails similarity of their solutions– is expressed in the following way (with our notations):

$$\mathcal{S}(\mathtt{pb}_1, \mathtt{pb}_2) \leq \mathcal{T}(\mathtt{Sol}(\mathtt{pb}_1), \mathtt{Sol}(\mathtt{pb}_2))$$

with $\mathcal{T}$ a similarity measure on the solution space.[5] This relation holds for fuzzy classification for any $\mathcal{T}$ (because of its reflexivity) and also for smooth classification for a well-chosen $\mathcal{T}$, e.g., $\mathcal{T}(\mathtt{sol}_1, \mathtt{sol}_2) = 1/(1+e)$ where e is the weighted length of the shortest path from $\mathtt{sol}_1$ to $\mathtt{sol}_2$ in the solution space structured by the adaptation functions $\mathcal{A}_{\mathtt{r}}$ of the available reformulations. Nevertheless, the two approaches are quite different. In [2] the similarities are used in order to propose a restricted domain for $\mathtt{Sol}(\mathtt{tgt})$ –the set of solutions $\mathtt{sol}$ such that $\mathcal{T}(\mathtt{Sol}(\mathtt{srce}), \mathtt{sol}) \geq \mathcal{S}(\mathtt{srce}, \mathtt{tgt})$– thus providing a kind of adaptation. Conversely, for smooth classification, the given adaptation process $\mathcal{A}_{\mathtt{r}_1;\mathtt{r}_2;\ldots\mathtt{r}_q}$ determines the similarity measure $\mathcal{S}$. Another difference with our approach is that $\mathcal{S}$ is symmetrical in [2]. This can be explained by the fact that source cases are specific ("stories") in this work, whereas they are ossified in the current work which involves a non-symmetrical comparison between (general) source problems and (usually specific) target problems.

[5] A more general modelling of the CBR principle for the non-deterministic problems, based on the possibility theory, is also described in [2].

8 CONCLUSION

This paper presents an approach to case-based problem solving having benefit of hierarchical classification techniques, fuzzy logic principles and methods and techniques proper to CBR. Classification techniques enable to organise the case base thanks to a problem hierarchy based on a generality relation. Three processes of retrieval using this hierarchy are described: the strong, fuzzy and smooth classification processes. Each of them takes advantage of the hierarchical organisation of problems by facilitating access to the similar cases and avoiding the too dissimilar cases. Strong classification is a classical deductive process. Fuzzy classification is an extension of strong classification and is a deductive process in a fuzzy logic: the problems that did not match in strong classification may match with a similarity degree expressed by a fuzzy truth value. Finally, the smooth classification is an extension of fuzzy classification based on adaptation principles proper to CBR. Among the possible future work, an experimental study of the efficiency of the approach presented here and an extension of it to the reuse of several source cases in order to solve a sole target problem are envisaged.

ACKNOWLEDGEMENTS

The author would like to thank the persons of the Resyn and Kasimir projects and the referees who have helped improve this paper and have suggested some ideas for future work (not all their suggestions could have been integrated in the paper due to the lack of space).

REFERENCES

[1] F. Baader, B. Hollunder, B. Nebel, and H.-J. Profi tlich, 'An Empirical Analysis of Optimization Techniques for Terminological Representation Systems', in *Proceedings of the third International Conference on Principles of Knowledge Representation and Reasoning (KR'92), Cambridge, Massachussetts*, pp. 270–281, (1992).

[2] D. Dubois, F. Esteva, P. Garcia, L. Godo, R. L. de Màntaras, and H. Prade, 'Fuzzy set modelling in case-based reasoning', *International Journal of Intelligent Systems*, **13**, 345–373, (1998).

[3] D. Dubois, H. Prade, and J.-P. Rossazza, 'Vagueness, Typicality and Uncertainty in Class Hierarchies', *International Journal of Intelligent Systems*, **6**, 167–183, (1991).

[4] J. Koehler, 'Planning from Second Principles', *Artificial Intelligence*, **87**, 145–186, (1996).

[5] J. Lieber and B. Bresson, 'Case-Based Reasoning for Breast Cancer Treatment Decision Helping', in *Advances in Case-Based Reasoning — Proceedings of the fifth European Workshop on Case-Based Reasoning (EWCBR-2k)*, eds., E. Blanzieri and L. Portinale, LNAI 1898, 173–185, Springer, (2000).

[6] J. Lieber and A. Napoli, 'Using Classifi cation in Case-Based Planning', in *Proceedings of the 12th European Conference on Artificial Intelligence (ECAI'96), Budapest, Hungary*, ed., W. Wahlster, pp. 132–136. John Wiley & Sons, Ltd., (1996).

[7] E. Melis, J. Lieber, and A. Napoli, 'Reformulation in Case-Based Reasoning', in *Proceedings of the Fourth European Workshop on Case-Based Reasoning, EWCBR-98*, eds., B. Smyth and P. Cunningham, LNAI 1488, pp. 172–183. Springer, (1998).

[8] A. Napoli, C. Laurenço, and R. Ducournau, 'An object-based representation system for organic synthesis planning', *International Journal of Human-Computer Studies*, **41**(1/2), 5–32, (1994).

[9] B. Nebel, *Reasoning and Revision in Hybrid Representation Systems*, LNCS 422, Springer-Verlag, Berlin, 1990.

[10] J. Pearl, *Heuristics – Intelligent Search Strategies for Computer Problem Solving*, Addison-Wesley Publishing Co., Reading, MA, 1984.

[11] C. K. Riesbeck and R. C. Schank, *Inside Case-Based Reasoning*, Lawrence Erlbaum Associates, Inc., Hillsdale, New Jersey, 1989.

[12] *Handbook of Fuzzy Computation*, eds., E. H. Ruspini, P. P. Bonissone, and W. Pedrycz, Institute of Physics Publishing, 1998.

[13] B. Smyth and M. T. Keane, 'Using adaptation knowledge to retrieve and adapt design cases', *Knowledge-Based Systems*, **9**(2), 127–135, (1996).