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

To cite this version:

Jean Lieber. Strong, Fuzzy and Smooth Hierarchical Classification for Case-Based Problem Solving. 15th European Conference on Artificial Intelligence - ECAI’02, A. Mille, Jul 2002, Lyon, France, pp.81–85. inria-00107563

HAL Id: inria-00107563
https://inria.hal.science/inria-00107563
Submitted on 19 Oct 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Strong, Fuzzy and Smooth Hierarchical Classification
for Case-Based Problem Solving

Jean Lieber

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].

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 \( (pb, Sol(pb)) \) where \( pb \) is a problem and \( Sol(pb) \) is a solution of \( pb \). Solving a problem means associating a licit solution with it. A CBR system aims at solving a target problem denoted by \( tgt \) 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 \( src-case = (src, Sol(src)) \). A source problem \( src \) is a problem such that \( (src, Sol(src)) \) belongs to the case base, i.e. it is a problem for which a solution \( Sol(src) \) 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 src-e-case in the case base that is considered to be similar to tgt. Adaptation uses this retrieved case src-e-case in order to give a solution Sol(tgt) to tgt. The new case (tgt, Sol(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 tgt. A retrieval procedure is said to be adaptation-guided when any source case it returns is necessarily adaptable to solve 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 $C_1 \sqsubseteq C_2$ iff any individual denoted by $C_1$ is denoted by $C_2$. Two concepts $C_1$ and $C_2$ are equivalent --$C_1 \equiv C_2$-- if $C_1 \sqsubseteq C_2$ and $C_2 \sqsubseteq C_1$. 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 $\omega_0$ be the maximum for $\sqsubseteq$ (any individual of the domain is denoted by $\omega_0$). A finite set $\{c_0, c_1, \ldots, c_n\}$ can be organised in a hierarchy $\mathcal{H}_0$ for $\sqsubseteq$. This involves that $\omega_0$ is the root of $\mathcal{H}_0$ and that, for each $c_i$ and $c_j$, $c_i \sqsubseteq c_j$ iff there is a path from $c_i$ to $c_j$ in $\mathcal{H}_0$. The hierarchy $\mathcal{H}_0$ 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 $\sqsubseteq$ is given with the following property:

$$\text{if } pb_1 \sqsupseteq pb_2 \text{ then every solution of } pb_1 \text{ is a solution of } pb_2 \quad (1)$$

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

- $pb_1 = \{\text{context} = \text{My car does not work.} \quad \text{question} = \text{What can I do?}\}$
- $pb_2 = \{\text{context} = \text{My car does not work. Its petrol tank is empty.} \quad \text{question} = \text{What can I do?}\}$

It is consistent with (1) to assert that $pb_1 \sqsupseteq pb_2$. For instance, the solution Sol(pb_1) = “Buy a new car” of $pb_1$ is also a solution of $pb_2$ (not a good one, but a solution yet).

For the problems described by a context and a question, $pb_1 \sqsupseteq pb_2$ if the context of $pb_1$ is more general than the context of $pb_2$ and if the question of $pb_1$ “contains” the question of $pb_2$. In RESYN/CBR, $pb_1 \sqsupseteq pb_2$ 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 $pb_1$ and $pb_2$. Indeed, a solution of $pb_1$ 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 $pb_2$. In KASIMIR/CBR, $pb_1 \not\sqsupseteq pb_2$ if the patient description of $pb_2$ is more precise than the one of $pb_1$ and the set of treatment categories required in $pb_1$ contains the one in $pb_2$.

$\xi$ is called the generality relation between problems. Let $\omega_0$ be the maximum for $\sqsubseteq$. It is the generic problem of the application domain. In RESYN/CBR, the meaning of $\omega_0$ 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 $\omega_0$.

Hierarchical Organisation of Problems.

Let $\{pb_0, pb_1, \ldots, pb_n\}$ be a set of problems. This set can be organised in a hierarchy $\mathcal{H}_0$ for $\sqsubseteq$, with root $\omega_0$. This means that $(pb_1, pb_2)$ is an edge of $\mathcal{H}_0$ iff $pb_1 \sqsubseteq pb_2$ and there is no path from $pb_0$ to $pb_2$ of length $t \geq 2$. The algorithm of construction of $\mathcal{H}_0$ is identical to the one of a hierarchy of concepts $\mathcal{H}_0$ which is described in [1]. The source problems are assumed to belong to $\mathcal{H}_0$. These problems src-e are associated with their solutions Sol(src-e) thanks to pointers. The other problems of $\mathcal{H}_0$ are abstract problems: no solution is associated with them and their role is to better structure the hierarchy. The figure 1 illustrates that.

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 oncology 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 src-e are specific problems which are not very helpful to structure $\mathcal{H}_0$ and thus to make the retrieval easier. This is why, RESYN/CBR source cases (src-e, Sol(src-e)) are indexed. This means that, for retrieval purpose, the problem src-e is substituted in $\mathcal{H}_0$ by a problem idx(src-e) which is a generalisation of src-e –src-e $\sqsubseteq$ idx(src-e)– such that the solution Sol(src-e) of src-e is also a solution of idx(src-e). One could say that the case (src-e, Sol(src-e)) has been ossified in the case (idx(src-e), Sol(src-e)). In the following of the paper, source cases are assumed to be ossified.

![Figure 1. A problem hierarchy $\mathcal{H}_0$.](image-url)
4 STRONG CLASSIFICATION

The strong classification process in the hierarchy $\mathcal{H}_{pb}$ consists in searching the source problems $srcr$ that are more general than tgt: $srcr \gg tgt$. If there are such source problems $srcr$, the retrieval chooses one with some preference criterion (e.g., the lowest ones in $\mathcal{H}_{pb}$ are often preferred) and returns the case $(srcr, Sol(srcr))$ to the adaptation module. The adaptation consists simply in copying $Sol(srcr)$ in a solution $Sol(tgt)$ of 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:

$$\frac{tgt \succ srcr \succ tgt}{Sol(srcr)} \text{ is a solution of } srcr \quad Sol(tgt) = Sol(srcr) \text{ is a solution of } tgt$$

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 $\mathcal{H}_{pb}$ taking into account the following property: if $pb \nless tgt$, then no problem of $\mathcal{H}_{pb}$ more specific than $pb$ can be more general than $tgt$ (since $\ll$ is transitive). Therefore, if a node $pb$ of $\mathcal{H}_{pb}$ fails to the test "$pb \gg tgt$", then the sub-hierarchy of root $pb$ 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 texts (and not the problem questions) and reflects the uncertainty associated with the source problems $srcr$ (abstract problems are also fuzzified, so that the search is consistent with (2)). This enables to take into account the threshold effect. For example, let $srcr_1$ and $srcr_2$ be two source problems such that their contexts contain the respective conditions $age \leq 65$ and $age > 65$, the other conditions being equal and the solutions $Sol(pb_1)$ and $Sol(pb_2)$ being different. For a patient of age 66, proposing only the treatment $Sol(pb_2)$ is doubtful: if the patient is "young for her age", the solution $Sol(pb_1)$ might be better. That is why, these conditions are fuzzified so that the two propositions $Sol(pb_1)$ and $Sol(pb_2)$ are proposed.

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 $\succ$ by a similarity measure $S$

The generality relation between problems, $\succ$, can be fuzzified in a fuzzy relation $S$: the value of $pb_1 \succ pb_2$ is either true or false, whereas the value of $S(pb_1, pb_2)$ is a fuzzy truth value. $S$ and $\succ$ are related by the following property, for each problems $pb_1$ and $pb_2$:

$$S(pb_1, pb_2) = 1 \iff pb_1 \gg pb_2 \quad (2)$$

Thus, $S$ is reflexive: $S(pb, pb) = 1$, for each problem $pb$. It is assumed furthermore that $S$ is non-symmetric (but not necessarily anti-symmetrical) and max-min transitive, which means that for each problems $pb_1, pb_2$ and $pb_3$:

$$S(pb_1, pb_3) \geq \min(S(pb_1, pb_2), S(pb_2, pb_3)) \quad (3)$$

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

$$\text{if } S(pb_1, pb_2) = s \quad \text{then every solution of } pb_1 \text{ is an } s\text{-solution of } pb_2$$

(4)

$S$ is said to be a non-symmetrical measure of the similarity between a source problem $srcr$ and a target problem $tgt$ since it measures how $srcr$ "includes" $tgt$. For KASIMIR/CRB, the similarity measure $S$ is defined thanks to local measures $m_p$ defined at the level of each problem $pb$ of $\mathcal{H}_{pb}$:

$$S(pb, tgt) = m_p(tgt)$$

This fuzzification only affects some attributes of the problem contexts (and not the problem questions) and reflects the uncertainty of the treatment propositions $Sol(srcr)$ associated with the source problems $srcr$ (abstract problems are also fuzzified, so that the fuzzification of problems in $\mathcal{H}_{pb}$ is consistent with (2)). This enables to take into account the threshold effect. For example, let $srcr_1$ and $srcr_2$ be two source problems such that their contexts contain the respective conditions $age \leq 65$ and $age > 65$, the other conditions being equal and the solutions $Sol(pb_1)$ and $Sol(pb_2)$ being different. For a patient of age 66, proposing only the treatment $Sol(pb_2)$ is doubtful: if the patient is "young for her age", the solution $Sol(pb_1)$ might be better. That is why, these conditions are fuzzified so that the two propositions $Sol(pb_1)$ and $Sol(pb_2)$ are proposed.

5.2 Fuzzy classification principle

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

$$\text{if } S(srcr, tgt) = s \quad \text{then } Sol(srcr) \text{ is a solution of } srcr \quad Sol(tgt) = Sol(srcr) \text{ is an } s\text{-solution of } tgt$$

(5)

It indicates how the source cases can provide approximate solutions to $tgt$. 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 $\mathcal{H}_{pb}$ 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 $\mathcal{H}_{pb}$. 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 $srcr$ ordered by decreasing $S(srcr, tgt)$. This would require another term in the condition of line 3 replacing "$pb$ is not a source problem". For example, this term can be "$S(pb, tgt) > 0$" (the retrieval would return all the source problems similar to the target problem), "$S(pb, tgt) \geq \alpha$" (where $\alpha$ is a similarity threshold) or "$m_p < k$" (where $m_p$ is the number of source problems in SP, if the $k$ nearest neighbours of $tgt$ are requested).
A fuzzy hierarchical classification algorithm.

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

With $\mathcal{MMGT} = \{pb \mid pb$ is a problem of $\mathcal{HP}$ and $pb \not\in \text{tgt}\}$

$\mathcal{SP}_0$ is the set of maximums of $\mathcal{MMGT}$ for $\mathcal{P}$, i.e.,

$\mathcal{SP}_0 = \{pb \mid pb \in \mathcal{MMGT}$ and $\forall pb' \in \mathcal{MMGT}, pb' \not\approx pb \Rightarrow pb' = pb\}$

This set $\mathcal{SP}_0$ can be used to initialise the set $\mathcal{SP}$ in the algorithm of figure 2, instead of $\{\omega_p\}$, 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

$$S(pb_1, pb_2) = \begin{cases} 0 & \text{if } pb_1 \not\approx pb_2 \text{ then } 1 \text{ else } 0 \end{cases}$$

and if the fuzzy classification returns all the source cases similar to $\text{tgt}$ (term $S(\text{src}, \text{tgt}) > 0$) 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 $\mathcal{HP}$ by decreasing similarity to $\text{tgt}$. If $pb_2$ is considered before $pb_3$ during the process then $S(pb_1, \text{tgt}) \geq S(pb_2, \text{tgt})$. Therefore, if $pb_2$ is the current problem then, for each problem in the hierarchy that has not been treated yet, this problem is less similar to $\text{tgt}$ than $pb_2$. Thus, if $S(pb, \text{tgt})$ is considered to be too low, it is useless to keep on the fuzzy classification process.

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

$$pb_1 \triangleright pb_2 \Rightarrow S(pb_1, \text{tgt}) \geq S(pb_2, \text{tgt})$$

(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 $\text{tgt}$: if the hierarchy is well-structured, most of the problems which are far away from $\text{tgt}$ 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, the solutions are considered to be similar if there is some available adaptation knowledge to adapt $\text{Sol}(\text{src})$ in a solution $\text{Sol}(\text{tgt})$ of $\text{tgt}$.

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

$$pb = pb_0, r_1, pb_1, r_2, pb_2, \ldots, r_{n-1}, r_n, pb_n = \text{tgt}$$

such that an adaptation function $A_{r_i}$ is associated with each $r_i$ and enables to adapt a solution of the problem $pb_{i-1}$ in a solution of the problem $pb_i$. The ordered pair $(r_i, A_{r_i})$ is called a reformation and the reformulations constitute the available adaptation knowledge [7]. For instance, $(\approx, A_{\approx})$, where $A_{\approx}$ is a solution copy, is a reformulation based on property (1). If the current problem is a source problem $\text{src}$, the adaptation is made by $1^\text{st}$ adaptation of $\text{Sol}(\text{src}) = \text{Sol}(pb_0)$ in a solution $\text{Sol}(pb_0)$ of $pb_0$, thanks to $A_{\approx}$, $2^\text{nd}$ adaptation of $\text{Sol}(pb_1)$ in a solution $\text{Sol}(pb_1)$ of $pb_2$, thanks to $A_{\approx}$, ... $q$ adaptation of $\text{Sol}(pb_{q-1})$ in a solution $\text{Sol}(pb_{q-1})$ of $pb_q$, $A_{\approx}$ thanks to $A_{\approx}$. This composition of simple adaptations is denoted by $A_{\approx}(r_1; r_2; \ldots; r_q)$. 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 $d(pb_0, pb_1)$ is the weighted length of the shortest similarity path from $pb_0$ to $pb_1$ ($+\infty$ if there is no such path) and $\mathcal{S}(pb_0, pb_1) = 1/(1+d(pb_0, pb_1))$. It is assumed that the shorter a similarity path from $\text{src}$ to $\text{tgt}$ is, the better (more certain or more precise) the solution $\text{Sol}(\text{tgt})$ obtained by an adaptation along this similarity path is. This involves that the weight associated with $\approx$ in a similarity path is 0. The fuzzy classification algorithm can be easily modified to return an ordered pair $(pb, \text{sim}(pb, \text{tgt}))$ where $pb$ is a source problem: computing $\mathcal{S}(pb, \text{tgt})$ and searching a similarity path $\text{sim}(pb, \text{tgt})$ are done in the same process. The following inference rule summarises the smooth classification:

$$\text{tgt} \quad \text{S(\text{src}, \text{tgt})} = \text{src} \quad r_1, r_2, \ldots, r_n \quad \text{tgt}$$

$$\text{Sol(\text{tgt})} = A_{r_1; r_2; \ldots; r_n}(\text{src}, \text{Sol(\text{src})}, \text{tgt})$$

is a $\text{sol}$-solution of $\text{tgt}$

If the property (5) holds for $\mathcal{S}$, the problems of $\mathcal{HP}$ are still considered with a decreasing similarity to $\text{tgt}$. This is the case for
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/CRB 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 latter 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):

$$S(p_b_1, p_b_2) \leq T(Sol(p_b_1), Sol(p_b_2))$$

with $T$ a similarity measure on the solution space.\(^5\) This relation holds for fuzzy classification for any $T$ (because of its reflexivity) and also for smooth classification for a well-chosen $T$, e.g.,

$$T(sol_1, sol_2) = 1/(1+e)$$

where $e$ is the weighted length of the shortest path from $sol_1$ to $sol_2$ in the solution space structured by the adaptation functions $A_r$ 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 $Sol(tgt)$ –the set of solutions sol such that $T(Sol(source), sol) \geq S(source, tgt)$– thus providing a kind of adaptation. Conversely, for smooth classification, the given adaptation process $A_{r_1,r_2,…,r_q}$ determines the similarity measure $S$. Another difference with our approach is that $S$ 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


