On the Role of Splitting and Merging Past Cases for Generation of a New Solution

Carlos Bento Penousal Machado Ernesto Costa

Departamento de Engenharia Informática
Universidade de Coimbra
Vila Franca - Pinhal de Marrocos
3030 Coimbra - PORTUGAL
E-mail: bento@alma.uc.pt ernesto@moebius.uc.pt

Abstract. This paper introduces RECIDE, an implementation of our approach to case-based reasoning. A qualitative and a quantitative metric are used for case retrieval. RECIDE has a library of successful and failure cases. Generation of new solutions is driven by splitting and merging operations on successful cases. Failure cases represent constraints on the application of splitting and merging operators. $RECIDE_{PSY}$, an application of RECIDE in the domain of psychology, is introduced in this paper. We present the results obtained with $RECIDE_{PSY}$ when splitting and merging operations are considered for generation of a new solution and compare them with the ones produced when solutions are constructed from a single case.

1 Introduction

A Case-Based Reasoning (CBR) System depends strongly on its methods for retrieval and reuse of previous experiences. This distinguishes these systems from those relying on the generalisation of solutions from first principles (abstract knowledge).

The combination of CBR and abstract knowledge-guided techniques led to the development of knowledge-based retrieval systems [5]. These systems use domain knowledge for construction of explanations of why a problem had a specific solution in the past. Explanations are necessary to judge the relevance of the facts describing a past problem [1, 7, 2]

In our work on CBR we are mainly concerned with two aspects. One has to do with the fact that the CBR approach is mostly used when a strong theory is not available and past experience is accessible. Lack of a strong theory means that, in general, case explanations are imperfect. We consider three kinds of imperfections and use them for retrieval [2]. A second aspect relates to the role of failure cases in CBR. Some current CBR systems make use of failure cases to represent and explain past unsuccessful experiences [3]. In our approach, failure cases represent *intra* and/or *inter*-case dependencies which were violated during case reuse.

This paper introduces results obtained with $RECIDE_{PSY}$ (REasoning with Cases Imperfectly Described and Explained in the domain of PSYchology), an expert system developed from RECIDE which is a CBR shell that implements our CBR approach.

2 Overview of RECIDE

RECIDE functional structure comprises: a case retriever, and a case reuser (Figure 1). The case retriever accesses successful cases in the case library. For case selection we use a qualitative and a quantitative metric.

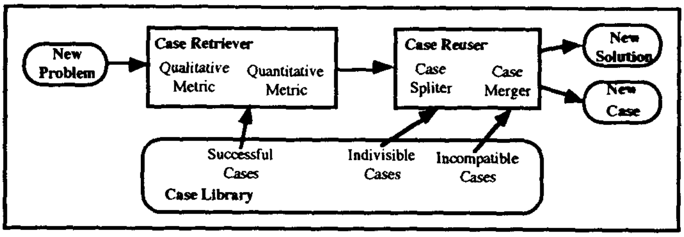

Fig. 1. Functional Structure of RECIDE.

The qualitative metric clusters past cases by the way in which they are potentially useful for creation of a new solution. The quantitative metric ranks cases in each cluster by its similarity with the new problem. The case reuser takes case clusters ordered by decreasing similarity and generates new cases that potentially have the same solution as the new problem. New cases are generated by applying splitting and merging operators, constrained by indivisible and incompatible cases. The need for splitting and merging operations on past cases follows from the fact that in general it does not exist a case in memory that comprises a complete solution for the new problem. In those situations catching the case pieces that have part of the solution for the new problem and merging them hopefully leads to a case comprising the new solution. This method of generating a solution shows to be particularly suitable for design tasks. The drawback of it is that, when a case is split, some *intra*-case constraints may be violated making this operation illegal. Also, in the merging step may be *inter*-case dependencies disable the synthesis of a new case from case pieces.

Within our approach *inter* and *intra*-case dependencies are represented in the form of indivisible and incompatible cases which are two kinds of failure cases. Their syntax is similar to the one for successful cases. Indivisible cases represent case pieces that when occurring in a case cannot be split. Incompatible cases represent case pieces that cannot occur in a new case by means of merging case parts from different cases. The semantic for failure cases is formally introduced in this section.

2.1 Case Library

The case library comprises: successful, indivisible, and incompatible cases. A successful case is represented by a triple $<P, S, R>$ (Figure 2) with P and

S, respectively, a set of facts representing past problem and solution, and R a set of rules given by the expert, representing a set of causal explanations. An explanation is a proof tree that links facts in the problem with a fact in the solution. We consider three kinds of imperfections in explanations: (1) incomplete set of explanations; (2) partial explanations; (3) broken explanations.

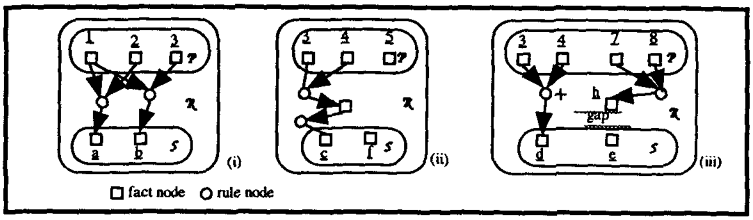

Fig. 2. A case with (i) a complete set of explanations; (ii) an incomplete set of explanations; (iii) a partial and broken explanation.

In a successful case with an incomplete set of explanations some solution facts are not explained and hence are not conclusion for any proof tree (e.g., Cases ii and iii in Figure 2. Facts f and e in these case solutions are not leaves of a proof tree). A partial explanation is one whose proof tree omits some branches. This means that one or more steps in the proof tree apply a rule for which the conditions are necessary but not sufficient. Rule nodes representing these rules are labelled by '+' (e.g., In Figure 2, case iii, the proof tree at the left). A broken explanation is one in which there is a gap between the proof tree and the case solution (e.g., In Figure 2, case iii, the proof tree at the right).

Failure cases (indivisible and incompatible), are represented by a triple $<Pf, Sf, Rf>$ with Pf and Sf the sets of facts representing, respectively, the problem and solution components, and Rf a set of rules. The semantic for these cases is different from the one defined for successful cases and is related to the splitting and merging operations performed during case reuse. The semantic for indivisible cases is (P, S, and R represent, respectively, the components of the case candidate for splitting):

i) If $Pf \neq \emptyset$, $Sf = \emptyset$, $Rf = \emptyset$ and $Pf \subseteq P$ then the subset Pf in P cannot be split.
ii) If $Pf = \emptyset$, $Sf \neq \emptyset$, $Rf = \emptyset$, and $Sf \subseteq S$ then the subset Sf in S cannot be split.
iii) If $Pf = \emptyset$, $Sf = \emptyset$, $Rf \neq \emptyset$, and $Rf \subseteq R$ then the subset Rf in R cannot be split.
iv) If $Pf \neq \emptyset$, $Sf \neq \emptyset$, $Rf \neq \emptyset$, and $Pf \subseteq P \wedge Sf \subseteq S \wedge Rf \subseteq R$ then subsets Pf, Sf, and Rf in P, S, and R have to remain in the same past case piece after the splitting process.
v) If $Pf \neq \emptyset$, $Sf \neq \emptyset$, $Rf = \emptyset$, and $Pf \subseteq P \wedge Sf \subseteq S$ then the subsets Pf and Sf in P and S have to remain in the same past case piece after the splitting process.
vi) If $Pf = \emptyset$, $Sf \neq \emptyset$, $Rf \neq \emptyset$, and $Sf \subseteq S \wedge Rf \subseteq R$ then the subsets Sf and Rf in S and R have to remain in the same past case piece after the splitting process.

vii) If $Pf \neq \emptyset$, $Sf = \emptyset$, $Rf \neq \emptyset$, and $Pf \subseteq P \wedge Rf \subseteq R$ then the subsets Pf and Rf in P and R have to remain in the same past case piece after the splitting process.

Indivisible cases of types i, ii, and iii constrain the splitting of facts in a problem or solution, or in a set of rules. Indivisible cases of type iv through vii constrain splitting between parts of the problem, solution, or set of rules.

Incompatible cases represent merging constraints on cases in memory. The semantic for incompatible cases is (P, S, and R are the components of the new case created by merging two or more cases or case pieces):

i) If $Pf \neq \emptyset$, $Sf = \emptyset$, and $Rf = \emptyset$ then Pf cannot occur in P as a results of merging.
ii) If $Pf = \emptyset$, $Sf \neq \emptyset$, and $Rf = \emptyset$ then Sf cannot occur in S as a results of merging.
iii) If $Pf = \emptyset$, $Sf = \emptyset$, and $Rf \neq \emptyset$ then Rf cannot occur in R as a results of merging.
iv) If $Pf \neq \emptyset$, $Sf \neq \emptyset$, $Rf \neq \emptyset$ and $Pf \subseteq P \wedge Sf \subseteq S \wedge Rf \subseteq R$ then Pf, Sf, and Rf cannot occur all together in the new case as a result of merging.
v) If $Pf \neq \emptyset$, $Sf \neq \emptyset$, $Rf = \emptyset$ and $Pf \subseteq P \wedge Sf \subseteq S$ then Pf and Sf cannot occur all together in the new case as a result of merging.
vi) If $Pf = \emptyset$, $Sf \neq \emptyset$, $Rf \neq \emptyset$ and $Sf \subseteq S \wedge Rf \subseteq R$ then Sf and Rf cannot occur all together in the new case as a result of merging.
vii) If $Pf \neq \emptyset$, $Sf = \emptyset$, $Rf \neq \emptyset$ and $Pf \subseteq P \wedge Rf \subseteq R$ then Pf and Rf cannot occur all together in the new case as a result of merging.

As with indivisible cases, incompatible ones of type i, ii, and iii relate to merging constraints at the fact level. Remaining case types report to constraints at the case component level.

2.2 Case Retrieval

Case retrieval is performed on a flat memory of successful cases. The retrieval process involves two steps:

i) Clustering of potentially useful past cases (qualitative metric).
ii) Ranking of case clusters (quantitative metric).

In the first step five clusters of past cases are created. Let S be the set of facts representing the solution for a case in memory and S' the set of facts representing the solution for a new problem. Each cluster comprises the following cases (in the examples that follow it is assumed the case library is composed by cases in Figure 2, and represented again in Figure 3):

CLUSTER_1 - Cases with $S = S'$.

e.g. If the new problem is described by the set of facts $\{\underline{1}, \underline{2}, \underline{3}\}$, CLUSTER_1 will be composed of case i (see Figure 2). Case i is completely explained, that is, facts $\{\underline{1}, \underline{2}, \underline{3}\}$ describing case problem and the new problem are necessary and sufficient for the solution $S = \{\underline{a}, \underline{b}\}$, therefore the new problem solution is $S' = S = \{\underline{a}, \underline{b}\}$.

CLUSTER_2 - Cases possibly with $S = S'$.

e.g. For a new problem described by the set $\{\underline{3}, \underline{4}, \underline{5}\}$, CLUSTER_2 will be composed of case ii. As the new problem is the same as the one described in case ii it is possible that case and new problem solutions are also similar. The reason why we are not certain about this is that case ii is not completely explained. Therefore we do not know if fact $\underline{5}$ is causally linked with fact $\underline{f}$ in the solution. This means the problem that has the solution $S = \{\underline{c}, \underline{f}\}$ may be different from the one represented in case ii provided it contains facts $\underline{3}$ and $\underline{4}$.

CLUSTER_3 - Cases possibly with $S \supset S'$.

e.g. Considering a new problem $\{\underline{1}, \underline{2}\}$, case i is the one in CLUSTER_3. As $\underline{1}$ and $\underline{2}$ are the causal premises for fact $\underline{a}$ in this case solution, it is possible that the new problem solution is $\{\underline{a}\} = S' \subset S$. The uncertainty about this is due to unknown *intra*-case dependencies which may be violated by splitting case i.

CLUSTER_4 - Cases possibly with $S \subset S'$.

e.g. With a new problem $\{\underline{1}, \underline{2}, \underline{3}, \underline{4}, \underline{5}\}$, cases i and ii are the ones in CLUSTER_4. As case i has the solution $S = \{\underline{a}, \underline{b}\}$ for problem $\{\underline{1}, \underline{2}, \underline{3}\}$ and case ii solution $\{\underline{c}, \underline{f}\}$ is supposed to be the one for problem $\{\underline{3}, \underline{4}, \underline{5}\}$ then it is possible that $\{\underline{a}, \underline{b}\} = S_i \subset S'$ and $\{\underline{c}, \underline{f}\} = S_{ii} \subset S'$, with S_i and S_{ii}, respectively, the solutions for cases i and ii. We are not certain about this as we do not know the *inter* and *intra*-case dependencies between and within cases i and ii.

CLUSTER_5 - Cases possibly with $S \cap S' \neq \emptyset$.

e.g. Assuming the new problem is $\{\underline{1}, \underline{3}, \underline{6}, \underline{9}\}$, CLUSTER_5 is composed by case i. As $\underline{1}$ and $\underline{3}$ are necessary and sufficient for $\underline{b}$ in the context of case i then $S_i \cap S' = \{\underline{b}\}$. The uncertainty on this is related to possibly unknown *intra* and *inter*-case dependencies.

Clusters above are not mutually exclusive. Considering, for instance, a new problem $\{\underline{3}, \underline{4}, \underline{5}\}$ case ii will belong to CLUSTER_2, as explained above, but it will also belong to CLUSTER_5 as it is possible that fact $\underline{5}$ in case ii is not the one responsible for fact $\underline{f}$ in the solution. If this happens then as $\underline{3}$ and $\underline{4}$ are necessary and sufficient for $\underline{c}$ in the context of case ii then $S_i \cap S' = \{\underline{c}\}$, with the constraint that no *intra* and *inter*-case dependencies are violated.

Cases within each cluster are ranked by an explanation-based similarity metric [2]. It assigns a distinct relevance to each fact in a case problem that matches a fact in the new problem, depending on the fact being premise of a complete, partial, broken, or no explanation at all. Clusters are sorted by decreasing similarity values.

Clustering of cases for retrieval embodies two main properties: (1) case clustering organises memory cases accordingly to their kind of potential usefulness for the new problem solution; and (2) it provides information on the most suitable method for creation of a new case. In the next subsection we describe how the reuse unit deals with these clusters.

2.3 Case Reuse

RECIDE reuse unit works with successful cases in terms of case pieces. Four types of pieces are considered (Figure 3): strong, weak, undetermined, and unexplained.

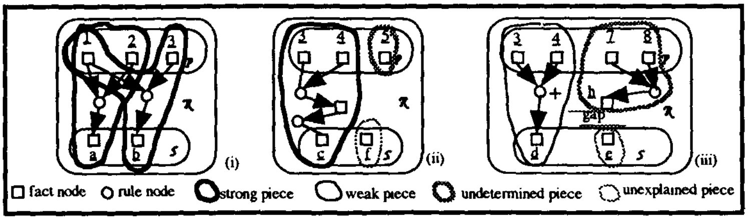

Fig. 3. Types of case pieces.

A strong piece comprises a complete explanation, the facts that are premises of it, and the fact that is its conclusion (e.g. In Figure 3, the pieces in case i). A partial explanation, its premises and its conclusion embody a weak piece (e.g. In Figure 3, case iii, the case piece at its left). A broken explanation and its premises or any single fact that is not premise of an explanation form an undetermined piece (e.g. In Figure 3, case ii, the piece composed by the single fact 5 and the piece in case iii at top right). Any fact in a case solution that is not conclusion of an explanation determines an unexplained piece (e.g. In Figure 3. single facts f and e in cases ii and iii are unexplained pieces). Case splitting is performed at the case piece level.

As described in section 2.2, after giving the system a new problem, successful cases in memory are clustered accordingly to their usefulness for the generation of a new solution. The reuse unit gets those clusters and performs the following steps:

1) generation of new cases;
2) selection of the new case most promising for the solution of the new problem;
3) validation of the solution provided by the selected case.

Each new case is created by splitting and merging operations on cases from a cluster. Two heuristics are applied for selection of the new case most likely to have the same solution as the new problem:

H1: Prefer new cases from clusters with lower index (e.g. CLUSTER_1 over CLUSTER_2).

H2: Prefer new cases with higher similarity values.

Heuristic 1, favours cases from those clusters with lower indexes. The reason to choose CLUSTER_1 is obvious. It is the only cluster that, if not empty, has a case known to have the correct solution. For the other clusters, preferring those with lower index means to choose new cases that required fewer splitting and merging operations for its generation. The more splitting and merging operations are performed, the more likely it is that unknown *intra* and/or *inter*-case dependencies are disregarded.

Heuristic 2 assumes cases with a problem description closer to the new problem description (matching more facts in the new problem, weighted the fact of being premise of a complete, partial, or interrupted explanation) have a higher chance of comprising the same solution as the new problem.

The next step comprises validation of the solution provided by the selected new case. In the validation step, RECIDE searches for a new case for which the splitting and merging operations involved in its construction do not violate the constraints imposed by failure cases in memory. Then it outputs the new case solution and the cases in the origin of it. If the user accepts the solution the validation process is finished.

If the new solution is not accepted, the user is encouraged to give the *intra* and *inter*-case dependencies in the origin of the wrong solution. Those descriptions are recorded as indivisible and/or incompatible cases. With the memory of indivisible and incompatible cases updated in this way the system starts another validation cycle selecting a new case that does not conflict with the updated library of failure cases.

If the user cannot explain why the new solution is wrong in terms of indivisible and incompatible cases then she/he is asked to give the solution for the new problem together with a causal justification. This input is recorded as a new successful case and the process is completed.

3 An Application in the Domain of Psychology

In this section we present an application of RECIDE in the domain of psychology. Results obtained with this application are also described in this section.

3.1 The Domain

RECIDE_{PSY} is an advising system for scholar underachievers. It suggests a new program for improvement of scholar performance supported on previous successful experiences.

A past experience comprises a context (past problem) in which a set of intervention strategies (past solution) was applied successfully. Figure 4 represents a case in the domain as it is output by RECIDE_{PSY}[1]. A '->' symbol in the explanations represents a complete explanation and a '->+' a partial explanation. This case describes a male client between twelve and fourteen years old, with two siblings, both younger and with a conflicting relation with relatives. The level of education achieved is six years of basic education and he is unfavourably

[1] The taxonomy introduced for context and intervention strategies is only relevant at the user's level. For matching a case with a new problem, RECIDE_{PSY} only considers ground facts organised in a flat structure.

PROBLEM:

<Subject Data and Familiar Background>
Sex: Male
Age: 12 - 14
Num. of Siblings: 2
Siblings are: Younger
Familiar Relationship: Conflicting

<Educational Background>
Degree achieved: 6
Educational Branch: Primary School
Siblings' Educational Achievements: Induces NegativeComparision

<Phycological Struct and Development Tasks>
Interpersonal Relationship: Low

<Learning Characteristics>
Num of Areas with Underachiev.: more than 3
Influencial Dispersion Sources: Internal
Underachiev. Started: Years Ago

SOLUTION:

<Main Stractegies>
Assertiveness Training
Selman's Interpersonal Negotiation Strategies
Enhancement of Learning Skills

<Complementary Strategies>
Self-knowledge Enhancement
Familiar Support Mobilization

<Behavioral and Cognitive Strats>
Role Playing
Thinking Cut-off
Recording of Thoughts, Behaviors and Emotions
Disfunctional Behaviors Evaluation

EXPLANATIONS:

Sex : Male AND Age: 12 - 14 -> Adolescence Crisis
Familiar Relationship: Conflicting AND Siblings' Educational Achievements: Induces Negative Comparision AND Adolescence Crisis -> Conflict Situations
Conflict Situations AND Interpersonal Relationship: Low -> Lack of Interpersonal Skills
Num of Areas with Underachiev.: more than 3 AND Underachiev. Started: Years Ago -> Enhancement of Learning Skills
Conflict Situations ->+ Self-knowledge Enhancement
Conflict Situations -> Familiar Support Mobilization
Lack of Interpersonal Skills ->Assertiveness Training AND *Selman's Interpersonal Negotiation* Strategies AND Recording of Thoughts, Behaviors and Emotions AND Disfunctional Behaviors Evaluation
Influencial Dispersion Sources: Internal -> Thinking Cut-off

Fig. 4. A successful case.

compared with his siblings due to their scholar achievements. Interpersonal relationship is low. His grades comprise more than three unsuccessful disciplines, shows internal sources of dispersion and has a long history of underachievement.

The main intervention strategies being applied are assertiveness training, Selman's interpersonal negotiation strategies, and enhancement of learning skills. The complementary strategies are self-knowledge enhancement and familiar support mobilisation. The behavioural and cognitive intervention strategies are role playing, thinking cut-off, recording of thoughts, behaviours and emotions, and dysfunctional behaviours evaluation.

The explanations provided by the experts for this intervention program are: (1) being a male client aged between twelve and fourteen are causing an adolescence crisis, (2) a conflicting familiar relationship marked by negative comparison, associated with the adolescence crisis characterise a conflict situation, (3) the conflicting situation, under development and his low level of interpersonal relationship describe his lack of interpersonal skills, (4) the number of underachievement areas being higher than three and the duration of this problem (starting years ago) cause the need for enhancement of learning skills, (5) the conflicting situation is a partial cause (the only partial explanation step in this case) for using self-knowledge enhancement, (6) the conflict situation is the cause for mobilisation of familiar support, (7) the lack of interpersonal skills is the motive for applying assertiveness training, Selmans interpersonal negotiation strategies, recording of thoughts, behaviours and emotions, and evaluation of dysfunctional behaviours, and (8) presence of internal sources of dispersion is the cause for using thinking cut-off.

In this task indivisible cases are of types i and ii (see subsection 2.1.). Incompatible cases are of types ii, iii, and v. The set of cases given by the experts comprises 47 successful cases and 65 failure cases (43 indivisible and 22 incompatible).

3.2 Experimental Results

Two kinds of tests (labelled TEST #1 and #2) were performed. In TEST #1 each iteration comprises input of the problem component of a case not in memory, generation of new cases in the way described in section 2.3, selection of the potentially best new case, output of its solution, and validation by the user. TEST #2 is like #1 with the difference that a new case is not created by splitting and merging operations but by selecting from memory the case most similar to the new situation. Then its solution is given as the solution for the new problem.

The parameters we consider in judging the quality of a proposed solution are: (1) rate of facts belonging to the generated solution appearing in the correct solution per total number of facts in the correct solution; (2) rate of facts in the generated solution that not belong to the correct solution per total number of facts in the correct solution; (3) difference between the first and second ratios.

The set of successful cases used in these tests was randomly ordered and this ordering was maintained along all the experiments.

Results showing the evolution of these parameters along a working session are presented in Figures 5 through 7.

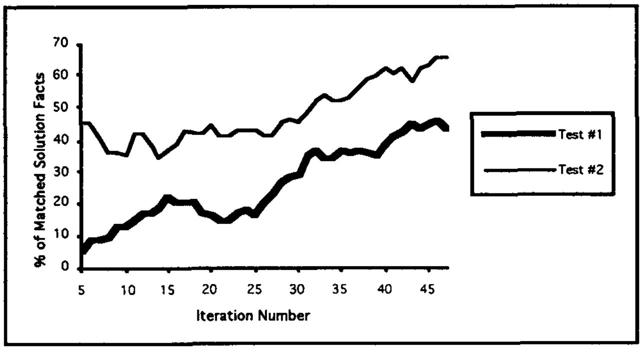

Fig. 5. Percentage of correct facts in the proposed solution relative to the total number of facts in the problem solution.

The number of facts correctly included in the new solution is higher in TEST #2 than in TEST #1 along the 47 iterations (see Figure 5), but it is also TEST #2 that shows the highest rate of facts wrongly included in the new solution (see Figure 6). In particular, till the 12th iteration, solutions produced by TEST #2 involve a high number of facts wrongly included in the generated solution. Figure 7 shows a measure of global quality of solutions as it takes into account the facts correctly and wrongly included in it. It is evident from this figure that till the 25th iteration, generation of a new solution by splitting and merging past

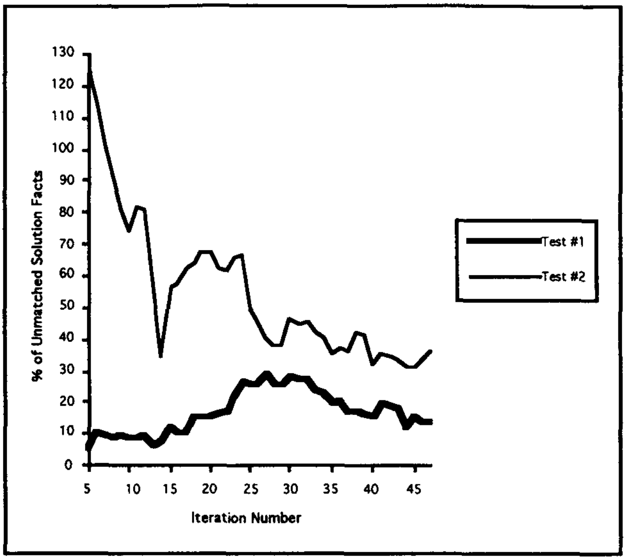

Fig. 6. Percentage of wrong facts in the proposed solution relative to the total number of facts in the problem solution.

cases produces better results than when these operations are disabled. After the 25th iteration using the best case in memory as the solution of a new problem or creating a new case by splitting and merging past cases does not make a difference.

3.3 Analysis of the Experimental Results

Considerations on the experimental results relate, at first to the facts (intervention strategies) correctly proposed by the system for a solution (treatment).

In contrast with previous expectations, the system performs better in terms of this parameter when only the best case in memory is retrieved then when a new one is created from previous cases. Our explanation for this is that when a single case is selected it tends to suggest a huge set of intervention strategies, many of them being correct (see Figure 5), but also with many wrong ones (see Figure 6). When a new case is generated by splitting and merging previous cases, the system leans to be more conservative in the sense that it only chooses case pieces comprising a causal relation between problem and solution pieces. In this way, when splitting and merging takes place for generation of a new solution the rate of facts wrongly included in the solution keeps low along the 47 iterations, never being higher than 30 % of the number of facts in the correct solution. A different result is obtained when a single case is retrieved for generation of a new solution. In this mode the rate of facts wrongly considered is high till the 25th iteration and very high till the 12th iteration.

When the number of correct and wrong facts is weighted for judgement of the solution quality (see Figure 7), it is clear that till the 25th iteration it is

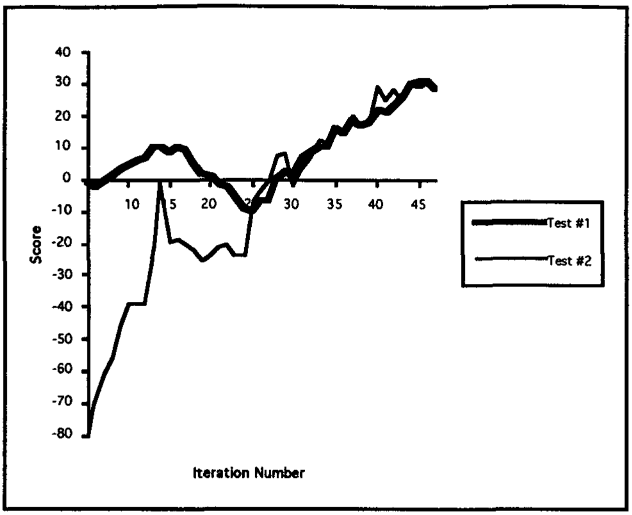

Fig. 7. Difference between percentage of correct and wrong facts in the proposed solution.

worth applying splitting a merging for generation of a new solution. After the 25th iteration no improvement is obtained with this method.

4 Final Remarks

Our last comments concern to the way RECIDE solves and learns and to the role of splitting and merging operations in the generation of new solution.

With respect to problem solving, the retrieval method used by RECIDE takes into consideration two important aspects - usefulness and similarity - assigning a higher importance to usefulness. As pointed out by other authors [4, 6] we believe search driven by usefulness plays a main role in case retrieval. Case clustering. as performed within our approach, relates to the role cases can play in the construction of a new one. Similarity is considered for case ranking within clusters.

Failure cases as they are defined in our framework constrain the generation of new cases by *intra* and *inter*-case dependencies. Many times, the reason why cases created by splitting operations do not have the correct solution roots in *intra*-case dependencies that were not perceived *a priori*. A similar problem takes place when case pieces are merged due to *inter*-case dependencies. Indivisible and incompatible cases are a powerful way to represent those dependencies.

An aspect that needs to be taken into consideration is that if the combination strategies used for case generation are not maintained under control the process leads to combinatory explosion. This is prevented by limiting combination of case pieces to the most promising cases within each cluster.

In RECIDE, the learning process comprises interactive acquisition of failure and successful cases. In general the acquisition *a priori* of *intra* and *inter*-case dependencies is not feasible. The problem-solving process provides a context of failure in which the analysis of the cases in the origin of a wrong solution is a way to detect violated dependencies that were the cause for the wrong solution. Incompatible cases also make possible to represent that a solution proposed by the system is incompatible with the new problem given to the system.

Another remark relates to the role of splitting and merging operators. It is clear that results are improved when this method is applied. Although in this domain the improvements which are obtained are not as important as we would expect. We believe this is related to the way cases are selected for splitting and merging. In this approach we select the most similar cases for splitting and merging. Intuitively, it is better to select cases which are complementar in terms of usefulness for splitting and merging than those which are most similar to the new situation. At the moment we are studding different strategies for selection of cases for splitting and merging.

A last comment has to do with the fact that these results were obtained for a specific domain. It is expected that in different domains a slightly different behaviour is detected.

5 Acknowledgements

We would like to thank Paula Vieira and Eduarda Góis who provided the case library. Fundação Luso-Americana para o Desenvolvimento and Fundação Calouste Gulbenkian financially supported our contacts with other groups working on CBR in the USA.

References

1. Barletta, R., and Mark, W., Explanation-Based Indexing of Cases, in Proceedings of a Case-Based Reasoning Workshop, Morgan-Kaufmann, 1989.
2. Bento, C., and Costa, E., A Similarity Metric for Retrieval of Cases Imperfectly Explained, in Wess, S.; Althoff, K.-D.; and Richter, M. M., eds., Topics in Case-Based Reasoning - Selected Papers from the First European Workshop on Case-Based Reasoning, Springer Verlag, Berlin: Germany, 1994a.
3. Hammond, K., CHEF: A Model of Case-Based Planning, in Proceedings of AAAI-86, Cambridge, MA: AAAI Press / MIT Press, 1986.
4. Kolodner, J., Judging Which is the Best Case for a Case-Based Reasoner, in Proceedings of a Case-Based Reasoning Workshop, Morgan-Kaufmann, 1989.
5. Koton, P., Using Experience in Learning and Problem Solving, Massachusets Institute of Technology, Laboratory of Computer Science (Ph D diss., October 1988). MIT/LCS/TR-441, 1989.
6. Smyth, B., and Keane, M., Retrieving Adaptable Cases: The Role of Adaptation Knowledge in Case Retrieval, in Wess, S.; Althoff, K.-D.; and Richter, M. M.. eds., Topics in Case-Based Reasoning - Selected Papers from the First European Workshop on Case-Based Reasoning. Springer Verlag, Berlin: Germany, 1994.
7. Veloso, M., Learning by Analogical Reasoning in General Problem Solving. Ph D Thesis. School of Computer Science, Carnegie Mellon University, Pittsburgh. PA. 1992.