Computational Intelligence, Volume , Number ,

CASE-BASE MAINTENANCE BY INTEGRATING CASE INDEX REVISION AND CASE RETENTION POLICIES IN A DERIVATIONAL REPLAY FRAMEWORK

Héctor Muñoz-Avila
Department of Computer Science
University of Maryland
College Park, MD 20742-3255
munoz@cs.umd.edu
(301) 405-2684 | FAX: 405-6707

This article presents case-base maintenance policies for case index revision and case retention. The policies are formulated in the context of case-based planners performing case adaptation by derivational replay. We implemented these policies on a particular case-based planner and claim that our case index revision policy improves the accuracy of the retrieval and that the case retention policy filters redundant cases better than other case retention policies known from the literature. Our claims are validated by empirical validation. We will observe that there is an inter-relation between the two policies that improves the filtering process of the case retention policy.
Key words: case-based maintenance, derivational analogy, feature weighting, planning

1. INTRODUCTION

Researchers in machine learning have long observed that unsupervised growth of knowledge will reduce the overall *utility* of a learning system (Minton, 1886; Markovich & Scott, 1993) but only until recently with the growing number of applications of Case-based Reasoning (Althoff et. al., 1995), researchers and practitioners have increasingly drawn their attention towards case-base maintenance (CBM). Studies include policies for case deletion (Smyth & keane, 1995), over-all reduction of case libraries (Lei et. al., 1999) and algorithms for detection of case redundancy and inconsistency (Racine & Yang, 1997). In addition only until recently, a framework has been proposed for describing CBM systems (Leake & Wilson, 1998).

In this article we report on our study on CBM policies for general-purpose case-based planners that perform case adaptation by Derivational Replay. In this adaptation method, an underlying first-principles planner is guided by following the sequence of planning decisions, also known as the derivational trace, that solved previous problems (Veloso & Carbonel, 1993). Adaptation by derivational replay is a powerful method and has been the subject of study on a variety of planning paradigms including non-linear planning (Veloso, 1994), partial-order planning (Ihrig & Kambhampati, 1995) and hierarchical planning (Muñoz-Avila et. al., 1994).

Our CBM policies are based, both, on the *outcome* and the *benefits* of the retrieved cases during case-based problem solving episodes. Intuitively, the outcome of the retrieved case(s) is considered successful if the case(s) can be extended to a solution without revising the planning decisions that were made by following the cases. In such a situation the retrieval is said to be successful. Otherwise it is considered a failure (Ihrig & Kambhampati, 1997). The benefit of the retrieved case(s) is a measure of the effort in the adaptation process (Muñoz-Avila, 1999). If the adaptation effort is low, the retrieval is considered beneficial. Otherwise it is considered detrimental. As we will state later, the outcome and the benefit of the retrieved cases have no logical relation (i.e., neither implies the other one). However, we will use these concepts to state CBM policies and show an inter-relation between these policies. The following are the contributions of this article:

- We state a case index revision policy based on the outcome of the retrieval.
- We state a case retention policy based on the benefit of the retrieval to the adaptation process.
- We show how the index revision policy improves the case retention policy.
- We classify previous work on general purpose case-based planning using Leakes and Wilsons CBM framework.

This article continues as follows: in the next section we motivate our studies on CBM for general purpose case-based planners. Section 3 discusses derivational replay. Next, we present the CBM policies developed in our work and discuss their relation. Then, in Section 5 we present an empirical evaluation. After that, other CBM approaches to general-purpose case-based planning are compared using Wilsons and Leakes framework. Finally, some concluding remarks are made.

2. MOTIVATION

The study of CBM policy in the context of synthesis tasks such as planning is not new. Fox and leake (1995), for example, introduced a policy for revising indexes in situations occurring during case based problem solving episodes. Basically, if a more suitable case existed in the case base than the one selected during the retrieval process, an introspective analysis of the retrieval is made to avoid making the same mistake in future retrieval episodes. This approach requires, in addition to the planning knowledge, an introspective model of the retrieval method. A motivation of our work is that we wanted to develop CBM policies which obviate additional requirement besides the planning knowledge.

Another well-known CBM policy is based on the notions of *coverage* and *reachability* sets (Smyth & keane, 1995; Smyth & McKenna, 1999). In the context of case-based planning we can re-phrase these concepts as follows: given a plan pl, its coverage set, written *CoverageSet(pl)*, is formed by all problems $\{p\}$ such that pl can be adapted to solve p. Given a problem p its reachability set, written *ReachabilitySet(p)*, is formed by all plans $\{pl\}$ such that pl can be adapted to solve p.

One of the first questions that any general purpose planning algorithm must answer is if it is complete or not. A planning algorithm is *complete* if given any solvable problem, the algorithm can produce a plan solving that problem. Several planning algorithms such as SNLP (Weld, 1993) or Graphplan (???) have been shown to be complete. In the context of case-based planning, Hank & Weld (1995) demonstrated the completeness of an adaptation algorithm based on SNLP. Muñoz-Avila (1998) showed the completeness of adaptation by derivational replay based on SNLP and conjectures that if its underlying first-principles planning system is complete, adaptation by derivational replay will also be complete.

If an adaptation algorithm is complete, and given any solvable problem p and any plan pl, then the following holds:

- $ReachabilitySet(p) = \{pl : pl \text{ } is \text{ } a \text{ } plan\}$
- $CoverageSet(pl) = \{p : p \text{ } is \text{ } a \text{ } solvable \text{ } problem\}$

This follows directly from the notion of completness. If p is a solvable problem, then given any plan pl, pl can be adapted to solve p. Otherwise, the algorithm would not be complete. Notice that this doesn't mean that having a single plan for adaptation is adequate as the effort required to adapt it might be too large or even larger than planning by first principles.

This means that the notions of coverage and reachability sets are not particularly useful in the context of a case-based planner build on top of a first-principles planner as they do not allow grouping cases in several classes making them indistinguishable . Part of the difficulty of applying this and other techniques in this context is the role of the cases. In general purpose case-based planning, cases are used to guide the search (Veloso & Carbonell, 1993) as opposed to other approaches in which cases enhance the knowledge about the domain itself (Hammond, 1986). This of course doesn't mean that the work on reachability and coverage or approaches requiring additional knowledge besides planning knowledge are not useful for synthesis tasks such as planning. It is well known that the requirement of first-principles planners to have a symbolic representation is infeasible for many domains. For those domains, however, in which that requirement can be met (e.g., (Koehler, 2000)), we want to introduce a CBM approach more suitable for this context.

3. ADAPTATION BY DERIVATIONAL REPLAY

In Adaptation by derivational replay, cases contain the derivational trace that led to a plan instead of the plan itself (Veloso, 1994). A derivational trace is a sequence of planning decisions that led a first-principles planning system to obtain a plan. Figure 1(a) depicts a typical planning decision that a first-principles planner must confront: given a goal G, the planner must choose between several operations (i.e., *opn-1*, *opn-2*,...,*opn-m*). Depending on which operation it chooses, a new set of subgoals $G1$, $G2$, ..., Gn is obtained. Some choices may led to a solution, others might led to a dead end requiring costly backtracking to search for an alternative solution. Furthermore, in some situations none of the alternatives may led to a solution if in a previous planning decision a wrong choice was made.

Derivational replay guides the choice that has to be made during the planning decisions. Figure 1(b) illustrates the replay step for the planning decision depicted in Figure 1(a). The derivational trace of a case indicates that to achieve a goal G' the operation $opn - i'$ was chosen, which resulted in a new set of subgoals $G1'$, $G2'$, ..., Gn'.

Derivational replay. Derivational replay of a planning decision is performed in three steps (see Figure 1(b)):

1. Unifying G with G'.
2. If this succeeds, a search is made among the operations achieving G (i.e., *opn-1*, ...,*opn-m*) for one which unifies with *opn-i'*.
3. If such operation, *opn-i*, exists, it is applied to decompose G into subgoals $G1$, $G2$, ..., Gn, which are unified against the subgoals of the case $G1'$, $G2'$, ..., Gn'.

How the adaptation process continues depends on the particular implementation of adaptation by derivational replay. Some implementations will continue the process by repeating the same three steps recursively with each subgoal Gi that unifies with a subgoal Gi' in the derivational trace until either all subgoals are solved or no subgoals in the derivational trace can be found that unify with the unsolved subgoals of the current problem. In the latter case, first-principles planning can be used to solve the remaining subgoals (e.g., Ihrig & Kambhampati, 000). Other implementations may use heuristic knowledge to interrupt the replay process and select between alternative cases and first-principles planning to solve the remaining subgoals (Veloso & Carbonell, 1996). However, independent of the particular implementation, adaptation by derivational replay always involve the combination of first-principles planning and derivational replay. That is, a derived plan will typically contain parts

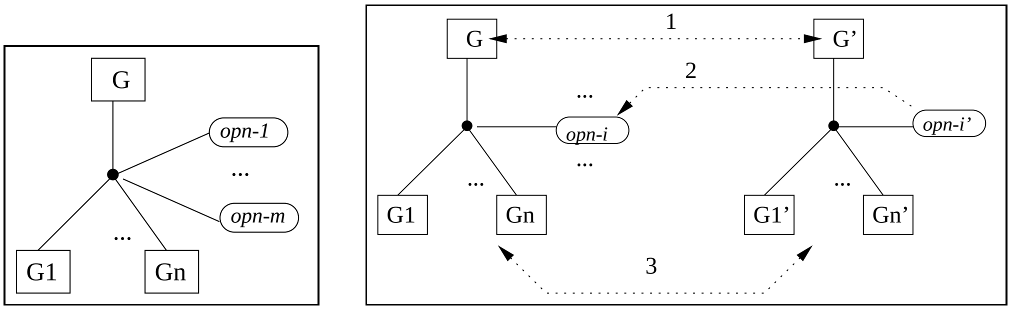

FIGURE 1. (a) a planning decision made by a first-principles planner and (b) a planning decision made by derivational replay.

which were obtained by derivational replay and parts which were obtained by first-principles planning.

It is well known that the benefit of adapting a rior plan depends on large measure on the accuracy of the retrieval (Smyth & keane, 1994). An inadequate retrieval procedure may lead to plans that are difficult or not adaptable at all. For the particular situation of adaptation by derivational replay, though cases can always be adapted, an inadequate retrieval procedure may led to a significant effort by the underlying first-principles planner. In a worst case scenario the first-principles planner may need to backtrack on all decisions made during replay. Thus, the resulting effort might be larger than if the first-principles planner would have solve the problem from the scratch even without taking into account the overhead caused by the retrieval process. Thus, an accurate retrieval technique combined with complementing CBM techniques that target reduction in case redundancy and increase in case competence are key for the effectiveness of adaptation by derivational replay.

4. TWO CBM POLICIES FOR DERIVATIONAL REPLAY

In this section we present two policies for CBM in the context of case-based planners adapting cases by derivational replay. We will describe each separately and then we will discuss how these policies are related.

4.1. Case Index Refinement Policy

Our case index refinement policy follows a punishment/reward system based on the outcome of the adaptation process. The outcome of the retrieval is considered a *failure* if planning decisions made with derivational replay need to be revised by the underlying first-principles planner (Ihrig & Kambhampati, 000). Otherwise the retrieval is considered successful. Figure 2 illustrates a retrieval failure. In a typical situation, part of the retrieved case (labelled B) is not replayed in the new situation. This happens when the operation (corresponding to B) in the derivational trace is not applicable in the current problem and as such it does not unify any of the operations solving the current goal (step 2 of the derivational replay process discussed in the previous section). When completing the solution, the first-

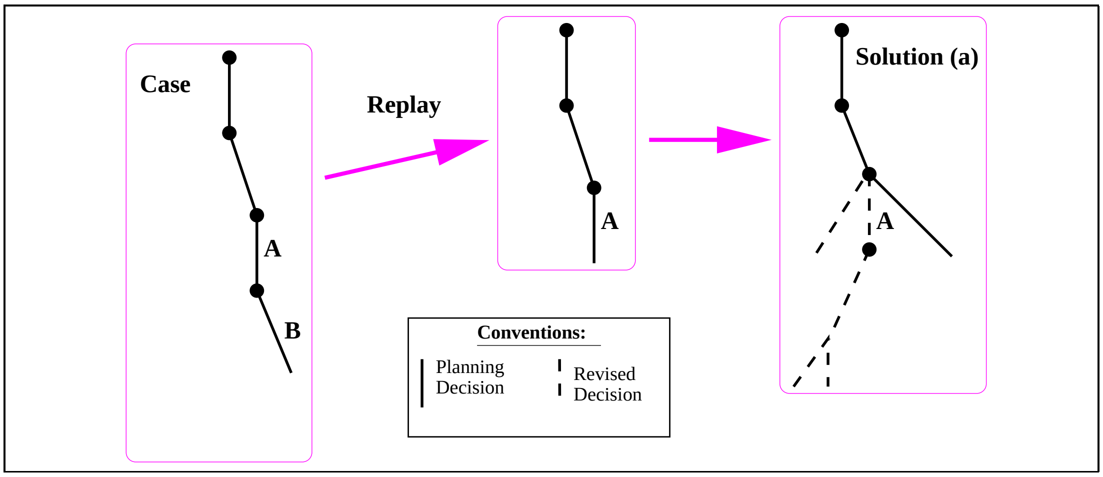

FIGURE 2. Example of a retrieval failure.

principles planner is forced to revise part of the replayed subplan (labelled A) and, thus, the retrieval is considered a failure.

Case Index Revision Process. Ihrig & Kambhampati use retrieval failures as case retention policies (we will discuss this in more detail later). In our approach we use this notion to determine the kind of case index revision that will be made to the retrieved case(s) index. Cases consist of a problem and the derivational trace that led to a plan solving the problem. A *problem* consists of a collection of features and goals. Features are statements about the world known to be true and goals are the statements that have to be achieved. Features and goal statements are represented as ground predicates. Each feature, i, has an associated weight, $\omega_{i,C}$, depending on the particular case C in which that feature occurs. When a new case is created the associated weight for each feature is set to one. With time these weights change indicating the relative importance of the feature in the case. Given a candidate case C and a problem P, our similarity assesment, $sim^{wg}(C, P)$, counts the weighted proportion of features in the case that occur in the new problem:

$$sim^{wg}(C,P) = \begin{cases} \sum_{i \text{ common to } P \text{ and } C} \omega_{i,C} & : \; if\ G_C \subset G_P \\ 0 & : \; otherwise \end{cases}$$

where G_C and G_P are the goals of the C and P respectively and $\subset$ is the subset-set relation.

The features and goals serve as indexes for the cases during the retrieval process. At each CBR problem solving episode, the feature weights of the retrieved case(s) may be revised.

The case index revision process for each retrieved case C occurs in three steps:

1. The outcome of the retrieval is stated for C (i.e., if the retrieval fails or is a success).
2. The set, $feat_C$ of all features in C that didn't occur in the new problem is determined.
3. The weight of each feature in $feat_C$ is revised.

For revising the feature weights, a feedback model based on incremental optimizersis used (Salzberg, 1991; Wettschreck, 1995). Each case, C, maintains two counters: k^C and f^C.

The first one indicates the number of times a case was successfully retrieved, and the second one the number of times its retrieval resulted in a failure. The weight, $\omega_{i,C}$, of a feature i is updated according to the following equations:

$$\omega_{i,C} = \begin{cases} \omega_{i,C} + \triangle_{k^C,f^C} & : \; if\ the\ retrieval\ is\ a\ sucess \\ \omega_{i,C} - \triangle_{k^C,f^C} & : \; if\ the\ retrieval\ fails \end{cases}$$

where the incremental factor $\triangle_{k^C,f^C}$ is always constrained by: $0 \leq \triangle_{k^C,f^C} \leq \beta \times n^C$. The number of features in the initial state of C is denoted by n^C. Thus, the change in the weight of the features is bound by a factor, $\beta \times n^C$, directly proportional to the number of features in the case ($\beta \geq 1$). If the value of ω_i^C is smaller than 1, then $\omega_{i,C}$ is assigned the value 1 and the weights of the other features in the case are incremented proportionally. The incremental factor $\triangle_{k^C,f^C}$ is computed as follows:

$$\triangle_{k^C,f^C} = \begin{cases} \beta \times n^C - (k^C/f^C) & : \; k^C < f^C \\ \beta & : \; k^C = f^C \\ \beta \times f^C/(k^C) & : \; k^C > f^C \end{cases}$$

The incremental factor $\triangle_{k^C,f^C}$ depends on the values of k^C and f^C in the following way: the larger the ratio of k^C to f^C, the smaller is the value of $\triangle_{k^C,f^C}$. Thus, as the number of adequate retrieval episodes increases, the effect of a retrieval episode on the feature weights decreases. In contrast, the smaller the ratio of k^C to f^C, the closer is $\triangle_{k^C,f^C}$ to $\beta \times n^C$. Thus, the effect is the opposite: the larger the ratio of f^C to k^C, the higher the value of $\triangle_{k^C,f^C}$ (i.e., $\triangle_{k^C,f^C}$ comes closer to $\beta \times n^C$).

The purpose of our case index revision policy is to tune the feature weights according to relative importance of a feature in a particular case. The relative importance of a feature for retrieval is indicated by the following explanation:

> If the weights of the features, i_1, i_2, ..., i_n of a case C were normalized so that $\omega_{i_1,C} + \omega_{i_2,C} + ... + \omega_{i_n,C} = 1$, then the factor $\sum_{j \neq k} \omega_{j,C} / \sum_i \omega_{i,C}$ expresses the *reliability* of making an adequate retrieval, when the feature i_k is the only one not occurring in the new problem.

Case Indexing Structure. We conceived an indexing structure that discriminates cases at two levels (see Figure 3): at the top level, the goal discrimination network (GDN) discriminates cases by the goals they achieve. It forms collections $\{Coll\}$ of cases such that within each collection $Coll$ all cases achieve the same goals. The GDN is not affected by the case index revision policy so we omit further discussion (details about the GDN can be found at (Muñoz-Avila, 1998)).

The second level indexes cases within the same collection $Coll$ by their weighted features. The weight of the feature relative to a collection $Coll$ is determined as follows:

$$\omega_{i,Coll} = \sum_{C \in Coll} \omega_{i,C}$$

The features occurring in cases in the same collection $Coll$ are grouped in intervals according to their weight relative to $Coll$ such that each interval has at most a predefined number, max, of features. In Figure 3 $w1$ corresponds to the weight of the feature with maximal weight. Within an interval features are grouped relative to their relative frequency in the collection in so-called discrimination trees (Veloso, 1994). Nodes in discrimination trees are sets of features. They have the property that the more frequently a feature occurs

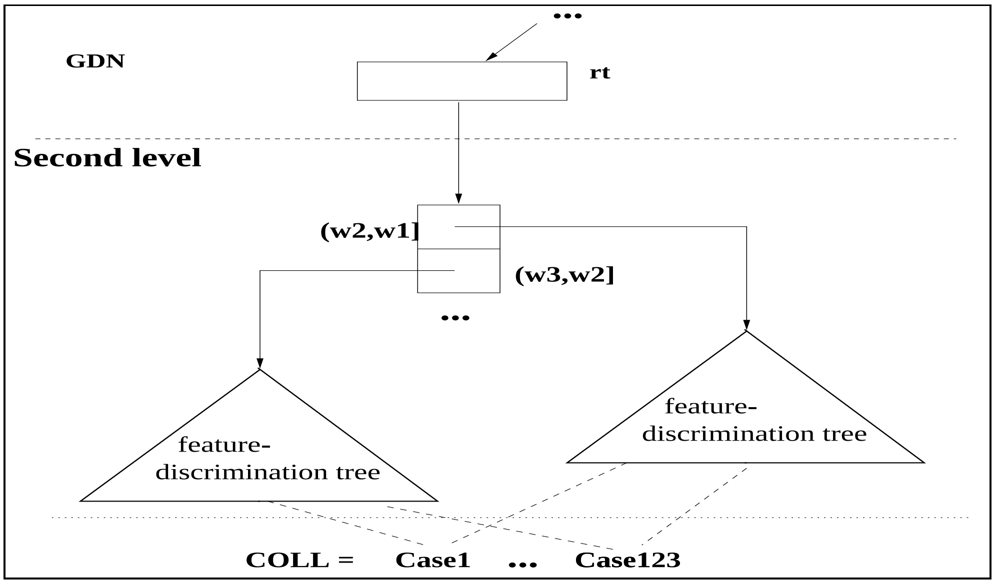

FIGURE 3. Fragment of the second level of the indexing structure.

in a collection *Coll*, the smaller the distance between the node containing that feature and the root. Thus, if a feature occurs in all cases, then it is contained in the root. Conversely, if a feature occurs in a single case, it is contained in a leaf of the tree. The root will contain no features if no feature is common to all cases in the collection.

During retrieval the second level is traversed keeping track of the weighted proportion of features in each case in *Coll* that occur in the current problem. The first interval visited is the one containing the features with the heavier weights. Retrieval continues by visiting the next interval with heavier weights and so on. Retrieval finishes when either a case C is found such that $sim^{wg}(C, P)$ is greater than a pre-defined threshold or the interval with the lighter weights has been visited and no case was found meeting this condition.

When case indexes are revised, the second level of the hierarchy may be revised as well because the weight of a feature relative to the corresponding collection of cases might change. There are two possibilities:

- The updated weight of the feature remains within the same interval as before. In this situation no further revision is necessary.
- the updated weight of the feature is no longer within the same interval. This leads to the reorganization of one or more intervals, which in turn leads to the creation of new discrimination trees for each interval that was modified.

4.2. Case Retention Policy

The outcome of the retrieved case is independent of the effort required to adapt the cases. If a retrieval failure occurs, it does not necessarily means that a large adaptation effort took place. The first-principles planner, for example, might only need to revise few of the planning decisions made during replay and might need to perform little search to solved the remaining subgoals. In such situations, retaining the found solution as a new case may led to an increment in the redundancy of the case base because clearly the existing case can cope with the new problem without much effort.

Conversely, the fact that the retrieval is a success does not necessarily means that the adaptation effort was low. If, for example, the derivational trace in the retrieved case contains a single planning decision and the first-principles planner must perform a large search effort to solve the remaining subgoals. Clearly in such a situation the retrieval is non beneficial. However, since a retrieval failure does not occur, the newly found solution is not stored as a new case. As a result this would be an opportunity to increase the competence of the case base.

To overcome these problems we stated a case retention policy based on the effort needed by the contribution of the retrieved cases to the overall adaptation effort. The effort is measured in terms of the size of the search space that was traversed to solve the problem which in turn is measured in terms of the number of planning decisions made for solving the problem. During the adaptation effort, the following values are computed: *SizeCases*, the number of decisions that were replayed from the cases, and *SizePlan*, the total number of decisions made by the underlying first-principles planner. *sizePlan* includes not only the planning decisions that led to a solution but also the decisions that were revised (including any decisions in which revisions to replayed decisions were made). The retrieval is considered *beneficial* if the following holds:

$$sizeCases/(sizeCases + sizePlan) \leq thr$$

where *thr* is a predefined threshold. If this condition does not hold we say that retrieval is non beneficial or detrimental.

Our case retention policy is to stored found solutions as new cases only if the retrieval is detrimental. The value of the threshold *thr* is a parameter of the system. For example, if it is set to 2, means that we are only going to create new cases if the size of the search space explored by the first principles planner is at least as large as the size of derivational traces replayed from the cases.

4.3. Case Index Revision Improving Retain Policies

We found an interesting effect by integrating both policies into a single system: if a retrieval is successful but detrimental, it means that despite not requiring to revise any of the decisions of the case, still a significant effort was required to solve the new problem. Clearly in such a situation, retaining the found solution as a new case is necessary to increase the competence of the case based planner. On the other hand, if a retrieval fails and is detrimental, its qualification as detrimental may be cause because several decisions from the retrieved cases need to be revised. In such a situation, adding the solution as a new case may be redundant. Ideally one could try to solve the problem again without the failing cases to determine of the retrieval is still detrimental. This is clearly not a feasible possibility in many situations. However, because of the case index revision policy, the more frequently a case is retrieved, the less likely is that the retrieval of the case results in a failure. That is, it becomes less likely that planning decisions replayed from the case need to be revised by the first-principles planner. This means, that over a period of time the case index revision policy makes even less likely that the case retention policy stores redundant cases. Our experiments will confirm this observation.

5. EMPIRICAL EVALUATION

We performed two experiments. In the first experiment we wanted to observe the effects of the case index revision and case retention policies locally for single cases. In the second experiment we wanted to observe the effect of the two policies to the overall case-based planning process.

5.1. Problem Domains

We performed experiments with the the logistics transportation domain (Veloso, 1994) and the domain of process planning (Muñoz-Avila & Weberskirch, 1996a). The logistics transportation domain was originally specified in (Veloso, 1994), In this domain, problems consist, starting from a certain configuration of packages at different locations, to relocate those packages to other locations. There are different types of locations and two types of assets: trucks and airplanes, which can be used to move packages between locations. However, the assets have restriction on their use. For example, trucks can only travel between locations at the same city and airplanes can only travel between airports. This domain has a total number of 8 types of object and 9 operations.

The second domain is the domain of process planning for manufacturing mechanical workpieces symmetrical with respect to an axis. A planning problem in this domain is given by a geometrical description of a workpiece and of the stock (raw material). The description of a workpiece is built up from geometrical primitives like cylinders, cones and toroids that describe monotone areas of the outline, possibly augmented by features (threads, undercuts, surface conditions). For such a planning problem a sequence of processing operations is to be found that will machine the workpiece considering available resources (i.e. tools, machines) and technological constraints related to the use of these resources. The process begins with clamping a piece of raw material on a lathe machine that rotates it at a very high speed. In most situations, the outline of the workpiece cannot be machined in one step but repeated cutting operations are necessary to cut the difference between the raw material and the workpiece in thin horizontal or vertical layers. The domain has a total of 33 types of objects and 27 operators.

5.2. Evaluating the Case Index Revision Policy

Our first experiment measures the performance of the case index revision policies and case retention policy on local cases. Retrieval was performed in two modes, dynamic and static. In the dynamic mode, the case index revision process was performed. In the static mode, the weights of the features were always set to one and they remain unchanged throughout the experiment.

Experiment Setup.. The experiment setup consisted of 5 runs. In each run, a problem, called the pivot problem, was stated. A solution for the pivot problem was found; the solution together with the problem were used to form a case, C. All feature weights of C were set to 1 and k_C, f_C were set to 0. Then, some features of the pivot problem were randomly selected. A new goal and new features that do not occur in the pivot problem were also given. Taking as basis the pivot problem, new problems were formed by changing the fixed features, or/and by adding the new goal and the new features. Changing a fixed feature means changing the relations between the objects mentioned in the feature. For example, if a feature states that a truck is in a certain location, the changed feature will state that the truck is in another location. The problem collection met the following conditions:

1. For every problem in the collection and if the weights of all features is set to 1, C the the similarity is greater or equal to the threshold, which was set to 75%.
2. The number of times that fixed features were changed in the collection is the same. For example, if a fixed feature indicates the location of a truck and another fixed feature indicates that a post office is in a certain city, the number of problems in which the truck is changed of location is the same as the number of problems in which the post office is changed of city.
3. If n denotes the number of fixed features, then problems were ordered in a way that within a sequence of problems, $Problem_{mn+1}, ..., Problem_{mn+n}$, the number of changes of a fixed feature is the same ($m = 0, 1, ...$). For this reason, the number of problems in the collection is a multiple of the number of selected features.

In the experiments the multiple factor was 5. In addition, in the logistics transportation domain 5 features were fixed and in the domain of process planning 6. Thus, the collections consisted of 25 problems in the first domain and 30 in the second one. The total number of problems involved were 125 in the logistics transportation domain and 150 in the domain of process planning.

Discussion about the Experimental Setup.. Condition (2) and (3) ensure fairness. Condition (2) constraint that feature takes advantage by occurring more frequently than others. Condition 3 ensures that the distribution of the changing features is uniform through the collection.

Results.. The first two rows of Tables **??** and **??** summarize the results of this experiment for the domain of process planning and for logistics transportation domain respectively. The first row of these tables presents the percentage of times the pivot cases was retrieved. The second row presents the percentage of times that retrieving the pivot cases resulted in a retrieval failure. Each of the first five columns presents the results for a sequence $Problem_{mn+1}, ..., Problem_{mn+n}$ in the collection ($m = 1, 2, ..., 5$). The sixth column shows the results for the whole collection when the cases were retrieved in static mode.

Discussion of the Experiment Results.. These experiments show that case index revision increases the reliability of the retrieval by decreasing the percentage of retrieval failures. This can be seen by comparing the fifth column against the first column of each table. That is, comparing the reliability of retrieval with the dynamic mode after several retrieval episodes against the reliability of retrieval in the static mode. Notice, in addition, that in the later runs the changes in the percentages tend to decrease. This shows that there is a tendency of the feature weights to converge in the average situation constructed in the experiment setup.

5.3. Evaluating the Case Retention Policy

We compared our case retention policy based on detrimental retrievals against previous policies for case retention in which either new cases are added every time a solution is found or if a retrieval failure occurs. This experiment was performed in parallel to the experiments performed in the previous section. The beneficial threshold was set to 2 and the following items were measured:

1. Percentage of nonbeneficial retrievals (row 3 of Tables **??** and **??**).
2. Percentage of case-based, problem-solving episodes in which the retrieval is a failure but beneficial (row 4 of Tables **??** and **??**).

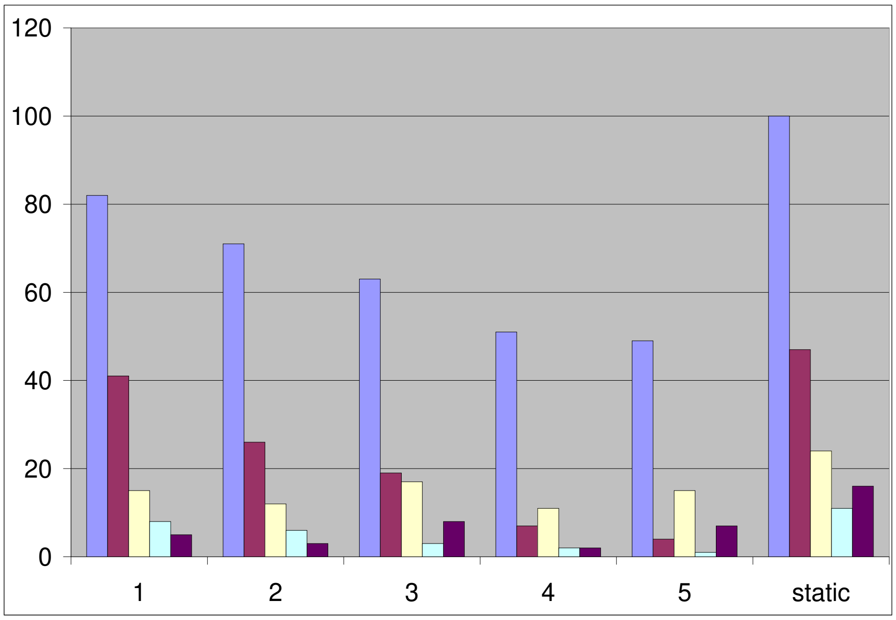

FIGURE 4. Results for the Domain of process planning.

3. Percentage of case-based, problem-solving episodes in which the retrieval is successful but non beneficial (row 5 of Tables ?? and ??).

Discussion of the Experiment Results.. From these results, some observations can be made: first, even in the last sequences, when the percentage of retrieval failures decreases, non beneficial retrieval episodes are still likely to occur. Thus, the adaptation effort can be significant independent of the fact that the retrieval tends to be successful. This is particularly interesting in the logistics transportation domain where even though the percentage of retrieval decreases in a significant way, the percentage of non beneficial retrievals oscillates. Second, by observing the results of the whole collection (i.e., the 6th column), when no feature weights are considered, it can be observed that the concepts of retrieval failure is also independent of the fact that the retrieval is beneficial or not. We conclude that creating cases based on the benefits of the retrieval is a more adequate policy than creating cases based on retrieval failures. Two arguments can be given supporting this conclusion:

- Cases may be created which can be generated with the existing cases with little effort.
- Situations may occur in which the adaptation is large but no new cases are created because no retrieval failure occurs.

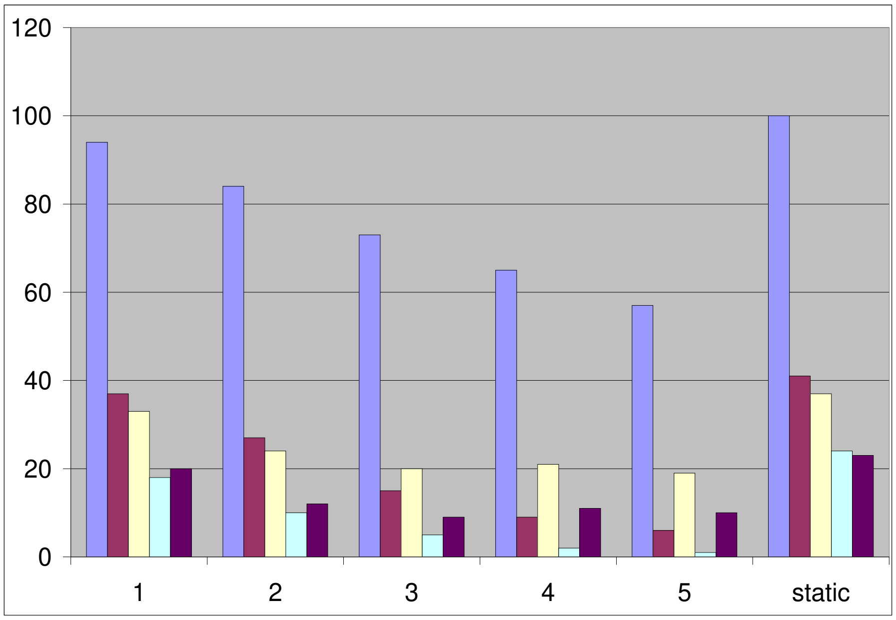

FIGURE 5. Results for the Domain of transportation domain.

5.4. Observing the Relation between the CBM Policies

We observe that if a case is retrieved frequently, it is likely that the retrieval failures involving that case will decrease in a significant way. Thus, a case retention policy based on retrieval failures will result in a significant decrease in cases being retained independently of the actual effort to solve the problems. Thus, we conclude that a case index revision policy based on retrieval failures is incompatible with a case retention also policy based on retrieval failures.

On the other hand, new cases should be create in situations in which the outcome of the retrieval is detrimental and no retrieval failures because even without revising decisions taken from the cases, the adaptation effort is large. Thus, we conclude that as the likelihood of a failure in the retrieval of a case reduces, any detrimental retrieval indicate the need for a creation of a new case because it cannot be redundant relative to the retrieved cases.

5.5. Evaluating the Impact of the CBM Policies on overall performance

In the previous sections we evaluated the performance of each of the CBM policies separately on pivot cases. We performed a second experiment in which we wanted to study the effect that the CBM polices has on overall CBR problem solving.

The Experiment Setup.. The experiment setup was designed in five runs. At each run a collection of problems was randomly formed. Each collection consisted of 2 one-goal problems, 4 two-goal problems, 8 three-goal problems, 10 four-goal problems and so on until 10 eight-goal problems. No two problems were the same cp Decision.epswithin a collection or in different collections. This experiment setup allowed a better observation of the effects of the learning process. In addition, it was intended to be a fair simulation of realistic situations; by not saturating the case base first with 1-goal problems, then with 2-goal problems and so on, problem-solving episodes are likely to occur in which, say, to solve an 8-goal problem only a 2-goal case is available. This is particularly significant given our previous experiments that showed the independence between successful retrieval and beneficial retrieval. The case base was maintained through all runs. The retrieval threshold was set to 75%. The beneficial threshold was set to 4 to 1. We performed the experiments on the domain of process planning.

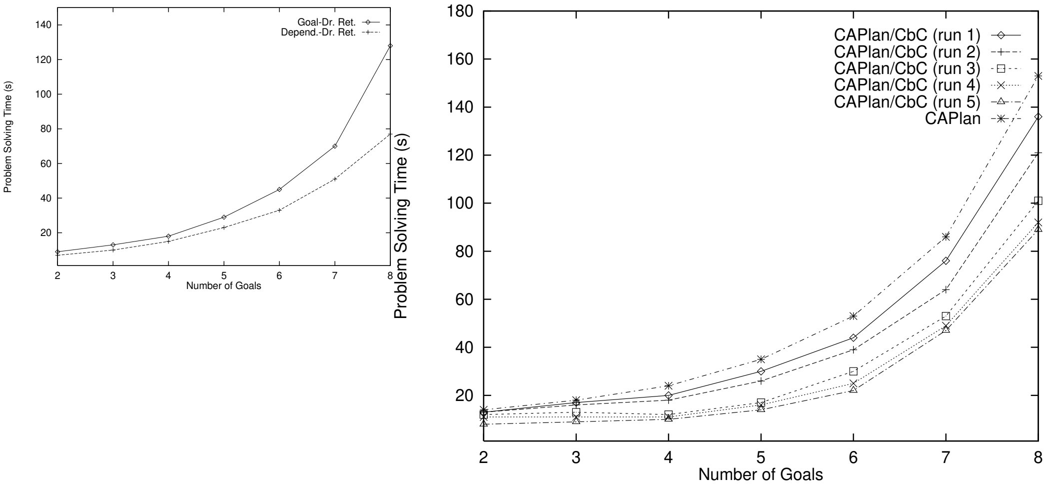

Figure 6. Performance gains with (a) standard retrieval and (b) the complete case-based planner.

Results.. We measured the time for solving each problem by CAPlan and CAPlan/CbC at each run when the CBM policies were not considered (Figure 6 (a)). We also measure the results with an retrieval procedure known as dependency-driven retrieval (Muñoz-Avila, 1995), which is particularly suitable for the domain of process planning as it uses domain-specific information about geometrical relations of the workpieces. We wanted to observe how our general CBM policies will compared with this specialized procedure. The results for CAPlan/CbC under the consideration of the CBM policies is shown in Figure 6 (b). The values for CAPlan and without the CBM policies correspond to the average of the five runs. For CAPlan/CbC with the CBM policies, each run is correspondingly labeled in one curve.

Discussion of the Experiment Results.. We observe that with each run the performance of CAPlan/CbC increases. Given that at the first two runs, less impact because of the case index revision policy take place, we conclude that the improvement there is mainly due to standard CBR guidance. This is corroborated by the fact that the values obtained are in the same range of those in Figure 6 (a). Thus, the further improvements in the next runs must be due to the combination of CBM policies. A significant increase in the performance takes place between runs 2 and 3 and then the increase rate is relatively low. This suggests that at a certain point no further efficiency gains will be made. Overall, the increase in performance of the case-based problem solving process is at least of 40% in the fifth run. In addition, on the fifth run, the results are comparable to those of the specialized retrieval procedure.

6. RELATED WORK

Leake and Wilson (1998) presented a framework for clasifying CBM systems among several dimensions, namely, the activation timing, the integration type, the scope of changes and the type of data:

- The *activation timing* can be periodic, if it takes place in a pre-set point of the CBR cycle, conditional, if it takes place when a condition is met or ad-hoc if it is defined by conditions external to the CBR system.
- The *integration type* can be on-line or off-line depending if it takes place during a CBR problem solving episode or not.
- The *scope of changes* can be narrow or broad depending if it affects a small portion of the case base or the whole case base
- The *type of data* reflects how the collection of data is done. It can be none, if no collection is done, synchronic, if the collection is done by considering the current case base or diachronic, if the collection is done by considering the evolution of the case base over a period of time.

We studied several case-based planners build on top of a first principles planner and classify their CBM policies according to Leakes and Wilsons framework. Prodigy/Analogy (Veloso & carbonell, 1993) implemented derivational analogy on top of a first-principles planner for the first time. Priar (Kambhampati & Hendler, 1994) introduced cases containing hierarchical plans. SPA is a case-based planner performing single-case adaptation (Hanks & Weld, 1995) and MPA is uses SPA to perform multiple-case adaptation (Francis, 1995). derSNLP+EBL (Ihring and Kambhampati, 000) performs case adaptation with standard replay based on a partial-order planner (SNLP). MRL uses deductive planning as inferencing mechanism Koehler, 1995). Paris stores and reuses reuses abstract cases (Bergmann & Wilke, 1995)

Most of the systems, namely, Prodigy/Analogy, Priar, MRL, SPA/MPA and Paris record new solutions after they are found in the CBR problem solving episodes. Thus, the activation timing is periodic because it always takes place at the same point of the CBR process and their integration type is on-line because it occurs during the course of the CBR problem solving episode. Because only a case is added, the scope is narrow. The type of data is none because no data analysis is performed; the new solution is always stored.

derSNLP+EBL CBM policy retains the solution found as a new case if a retrieval failure occurs. Thus, the activation timing is conditional. It is also on-line because it occurs during the course of the CBR problem solving episode. Because only a case is added, the scope is narrow. The type of data is synchronic because it considers the retrieved cases.

CAPlan/CbCs case retention policy has the same classification as derSNLP+EBL: it is conditional (i.e., if the retrieval is detrimental), is on-line, narrow and synchronic. CA-Plan/CbCs case indexes revision policy is also on-line. However, its activation timing is periodic because it always takes place (how the indexes are updated depends on the success or failure of the retrieval but the point is that the update always takes place). Its scope might be narrow if the feature weight intervals do not change but the scope is large if the feature weight intervals change as a result of the revision.q Another, difference is type of data is diachronic as it considers the trend of the retrieved cases. Leake and Wilson observed that diachronic data collection is the most informative.

7. FINALREMARKS

Impact: is general, independent of the previous decisions.
Future work: case deletion poilicies.

References

Aamodt, A. & Plaza, E. (1994). Case-based reasoning: Foundation issues, methodological variations and system approaches. *AI-Communications*, 7(1):pp 39–59.

Bergmann, R., Muñoz-Avila, H., Veloso, M., Melis, E. (1998). Case-based reasoning applied to planning tasks. In M. Lenz, B. Bartsch-Spoerl, H.-D. Burkhard, & S. Wess (Eds.) *Case-Based Reasoning Technology: From Foundations to Applications.* Berlin: Springer.

Ihrig, L. & Kambhampati, S. (1996). Design and implementation of a replay framework based on a partial order planner. In Weld, D., editor, *Proceedings of AAAI-96*. IOS Press.

Ihrig, L., & Kambhampati, S. (1994). Derivational replay for partial-order planning. *Proceedings of AAAI-94*, AAAI Press.

Kitano, H. & Shimazu, H. (1996). The experience-sharing architecture: a case study on corporate-wide case-based software quality control. In *Case-Based Reasoning: Experience, Lessons, & Future Directions*, Leake, D. B. (Editor). MA: AAAI Press / MIT Press.

Leake, D.B., & Wilson, D.C. (1998). Categorizing case-base maintenance: dimensions and directions. preprint.

M. Lenz, B., Bartsch-Spoerl, H.-D. Burkhard, & S. Wess (Eds.). (1998). *Case-Based Reasoning Technology: From Foundations to Applications.* Berlin: Springer.

Markovitch, S., & Scott, P.D. (1993). Information Filtering: selection Mechanisms in learning systems. *Machine Learning*, 10.

McAllester, D. & Rosenblitt, D. (1991). Systematic nonlinear planning. *Proceedings of AAAI-91*, AAAI Press.

Minton, S. (1990). Quantitative results concerning the utility of explanation-based learning. *Artificial Intelligence*, 42:363–391.

Muñoz-Avila, H. (1998). *Integrating Twofold Case Retrieval and Complete Decision Replay in CAPlan/CbC.* PhD Thesis. University of Kaiserslautern, Germany.

Muñoz-Avila, H. & Hüllen, J. (1995). Feature weighting by explaining case-based planning episodes. In *Third European Workshop (EWCBR-96)*, number 1168 in LNAI. Springer.

Muñoz-Avila, H. & Weberskirch, F. (1996a). A specification of the domain of process planning: Properties, problems and solutions. Technical Report LSA-96-10E, Center for Learning Systems and Applications, University of Kaiserslautern, Germany.

Muñoz-Avila, H. & Weberskirch, F. (1996b). Planning for manufacturing workpieces by storing, indexing and replaying planning decisions. In *Proc. of the 3nd International Conference on AI Planning Systems (AIPS-96)*. AAAI-Press.

Kirsti, R., & Qiang, Y., (1997). Maintaining Unstructured Case Bases, in Case-Based

Reasoning Research and Development, In Leake D., B., and Plaza E., (Eds). *Proceedings of the International Conference on Case Based Reasoning*, Springer.

Smyth, B., & Keane, M.T., (1995). Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems. In: *Proceedings of the International Joint Conference on Artificial Intelligence.*

Smyth, B., & Keane, M.T., (1998). Adaptation-guided retrieval: Questioning the similarity assumption in Reasoning. *Artificial Intelligence*, 102.

Veloso, M. (1994). *Planning and learning by analogical reasoning.* Number 886 in Lecture Notes in Artificial Intelligence. Springer.

Veloso, M. & Carbonell, J. (1993). Derivational analogy in prodigy: Automating case acquisition, storage, and utilization. *Machine Learning*, 10.