Flexible Feature Deletion: Compacting Case Bases by Selectively Compressing Case Contents

David Leake and Brian Schack

School of Informatics and Computing, Indiana University
Bloomington IN 47408, USA
leake@indiana.edu, schackb@indiana.edu

Abstract. Extensive research in case-base maintenance has studied methods for achieving compact, competent case bases. This work has examined how to achieve good solution performance while limiting the number of cases retained, using approaches such as competence-based case deletion. Two fundamental assumptions of such approaches have been (1) that cases are approximately the same size and (2) that the only way to affect case base size is by deleting or retaining entire cases. However, in some domains different cases may contain different amounts of information, causing widely varying case sizes, and case solutions may themselves be compressible, with the ability to selectively delete portions of indices or solutions while still retaining varying levels of usefulness. In accordance with this more flexible view, this paper proposes a new maintenance approach, *flexible feature deletion*, which removes parts of cases, enabling compression of the case base by selective—and possibly non-uniform—size reduction of individual cases. It proposes and evaluates an initial set of feature deletion strategies. Experimental results support that when cases have varying size and compressible contents, flexible feature deletion strategies may enable better system performance than case-oriented strategies for the same level of compression.

Key words: Case-base maintenance, feature reduction, case deletion, case-base compression

1 Introduction

The performance of case-based reasoning systems depends on the coverage of their case bases and the quality of their cases. As the number of cases in the case base grows, increased retrieval costs [1, 2] or storage constraints may require controlling case base size. Extensive case-based reasoning research has aimed to address this problem through case-base maintenance [3]. A key focus of this work has been on strategies for selecting cases to retain in the case base to maximize the competence achieved for a given number of cases. Approaches include strategies to guide deletion of cases from an existing case base [4], for determining when to retain a new case during problem-solving (e.g., [5]), and for ordering addition of cases from a candidate case set (e.g., [6, 7]). All of these strategies treat cases as single units, adding or deleting entire cases. We call such strategies "per-case" maintenance strategies.

Per-case strategies reflect two common implicit assumptions: (1) that all of the CBR system's cases will be of sufficiently uniform size that the size effects of deletion or addition do not depend on the chosen case, and (2) that the size of the internal contents of cases cannot be reduced. In domains for which each case must contain uniform knowledge, so that removal of any case information would severely impair the ability to use the cases, per-case strategies are the only appropriate choice. However, in some CBR domains, case contents are more flexible.

This paper questions the assumption of uniform case size in case-base maintenance, The assumption of uniform size means that, if cases are of different size, it is not possible, for example, to favor retention of smaller cases when those cases have comparable coverage. It also questions the assumption of maintenance only on a per-case basis, proposing that compression strategies can consider not only case deletion/addition but the deletion of components of particular cases. Rather than pre-determining a static set of features to be used throughout the life of the CBR system, the set of features to include in the case base could be adjusted based on requirements for storage, processing speed, and accuracy. There need be no requirement that all cases in the case base include the same set of features, just as there need not be uniform collections of components in the solution parts of cases, and the solutions need not be represented at the same level of granularity. This paper proposes a new, more flexible maintenance approach in which selective compression can be done at the level of the contents of individual cases, by removing selected features from either indexing or solution information. Thus this can be used to maintain both indexing features and features of a solution.

The motivation for adjusting case contents arises from domains in which cases are large and can be represented in multiple ways. For example, CBR has attracted interest for reasoning from imagery such as medical images (e.g., [8]). From any image, different features may be extracted, at different resolutions, and the amount of information required to represent different images might vary dramatically. In diagnostic domains, numerous features may carry information relevant to the diagnosis, with different pieces relevant to different degrees for different problems. When CBR is applied to design support, stored designs could selectively include different subsets of a full design or could include the design at different levels of detail. In a case-based planner generating highly complex plans, it is possible to retain the entire plan, or only key pieces, or to preserve full details for parts of the plans and high-level abstractions for others. Likewise, when CBR is applied to tasks such as aiding knowledge capture by supporting concept map construction [9], stored concept map cases could be retained at different levels of completeness. Exploiting this flexibility requires maintenance processes that can perform maintenance at a finer-grained level than simple retention or deletion of cases.

Feature compression is especially appropriate for complex domains in which cases are large, may contain extensive indexing or solution information, and in which partial information—for either indices or solutions—may still be useful. Feature compression prompts the question of when to delete an entire case versus when to achieve comparable space savings by abstracting, deleting, or otherwise compressing some of the features contained in one or more cases in the case base. There is no free lunch: Either method may entail accuracy losses, case deletion by removing what may be the most relevant solution for a problem; feature maintenance by affecting retrieval accuracy or quality of the solutions. The interesting question is how these methods compare.

This paper begins by discussing the range of applicability of flexible feature deletion and its relationship to standard case-base maintenance. It then defines a set of simple feature deletion strategies and evaluates their performance compared to per-case strategies for three domains, two with cases containing varying amounts of information and one with uniform size cases. A comparison of competence as a function of compression supports the value of flexible feature deletion for domains with variable-size cases.

2 When Feature Deletion is Appropriate

Feature compression is appropriate for a particular class of domains: Those in which a particular case can be represented with varying levels of information and still be useful. Even if indexing accuracy is reduced, the retrieved cases will provide value if they are still adaptable to usable solutions with an acceptable level of adaptation effort. Even if some poor retrievals result, they may be acceptable given savings in space; just as per-case maintenance usually involves a tradeoff of case base compactness against competence, feature deletion does as well.

The range of problems to which the case can be applied, and the reliability of its application, may vary with the specific information stored. Consequently, different feature deletion domains will exhibit different tradeoffs between per-case maintenance strategies and feature-maintenance strategies, as well as different tradeoffs between compression and quality. For some domains, such as a regression (numerical predication) task, feature deletion may only be possible for indexing information. In domains for which indexing information is based on many features, it may be possible to reduce case size by removing information about the values of some indices. Note that feature deletion of indices contrasts with the extensive work on selecting indexing vocabularies in the CBR literature, in that feature maintenance is aimed not at selecting an indexing vocabulary or maximizing retrieval accuracy, but instead at selectively compressing indexing information by deleting particular features, potentially from individual cases, with the recognition that some accuracy loss may result.

The deletion done by feature deletion is not necessarily limited to particular indexing dimensions (e.g., deleting the “age” attribute from all patient cases), but alternatively may delete specific attribute values (e.g., deleting the “age” attribute-value pair for specific patients, or only for a particular range of age values, such as those patients who fall into a default set for which age is not considered significant).

Feature deletion for indices could be especially relevant to situations in which extremely rich indexing information is available, such as a case-based agent to respond in a real-time strategy game, or a prediction system for driver behavior, for which the situation in which a plan was applied could be described with extremely rich detail—with fine-grained details which might be helpful to finding the perfect case, but not essential to finding a good case. Likewise, in a movie recommender domain, with movies characterized by their list of actors and the goal of recommending similar movies, a subset of the actors might be sufficient for good retrievals.

2.1 When to Apply Feature Deletion to Indices

Tasks are potential targets for feature deletion of indices if their cases have large indexing structures which can be reduced while retaining an acceptable level of indexing/similarity performance. Specifically, domains are appropriate if:

- *Indexing or similarity assessment depends on information about detail-rich situations from which many features could be generated.* If any low-level features of the current situation, or of a sequence of situations, might be available and potentially be relevant to deciding a response. In such domains, due to the potential for large amounts of indexing information, feature deletion could have significant effects on case base size.
- *Indexing or similarity assessment features are sufficiently closely related that acceptable accuracy is possible after removal of some features.* If features are closely related—even if they are not redundant—feature removal may have limited effects on system accuracy, helping to boost the amount of compression possible per unit of retrieval accuracy loss.

The CBR community has devoted substantial effort to methods for refining the indices used for cases, as well as on developing methods for assigning weights to features for similarity assessment. However, work in index/similarity refinement differs from feature deletion in a key way: The focus of index/similarity refinement is generally increasing retrieval accuracy, rather than compression of case data. Consequently, research on such methods does not address space/accuracy tradeoffs. Feature deletion is a primary focus of research on dimensionality reduction for CBR. However, such deletion is done uniformly across all cases; this work does not attempt selective deletion of a feature from some cases but not others.

2.2 When to Apply Feature Deletion to Solutions

Feature deletion is useful for domains in which the solution to a single problem can capture varying levels of information and still be useful. In such domains, parts of a large or complex solution may be removed or abstracted while still retaining the usefulness of a case, even if the level of usefulness varies with the specific information retained.

For example, as previously mentioned, in case-based planning, certain parts of a plan could be elided or abstracted to reduce storage. When a new planning problem is precisely covered by the retained material, there is no solution quality or efficiency loss. When it is not, the maintenance may result in increased adaptation cost to reconstruct the plan, or some competence could be lost—in weak-theory domains, plan failures could result if adaptation did not generate a perfect solution. However, partial deletion of case contents might still cause less competence loss than deletion of an entire case by per-case methods.

Case-based support for concept mapping [9] provides another example. Concept maps [10] are informal two-dimensional visual representations of concepts and their relationships, representing a particular user's conceptualization of a domain. The goal of support systems is to aid humans using electronic tools to build concept maps, by monitoring the concept map under construction, retrieving a past concept map relevant to the partial concept map they have constructed, and using it to suggest extensions to the concept map. Concept map cases contain rich structures of interconnected concepts, from which deletion of some parts may reduce the range of problems for which suggestions can be provided, but for which the remaining parts are still useful.

We note that for supporting concept map extension, any part of a concept map case may be viewed as the index or the solution, depending on which features are available as the input problem and the context of the retrieval [11]. Thus in the concept mapping domain, the same feature deletion process can be seen as simultaneously maintaining indices and solutions.

3 Bundling Features for Deletion

We can consider cases as composed of a set of primitive features which cannot be further decomposed. In what follows, for simplicity we will consider these to be attribute–value pairs. However, other representations are possible. Both indexing and solution information are defined by sets of features. For example, basic features could be combined to form complex structured cases, from which flexible feature deletion could remove multiple features corresponding to substructures.

Maintenance approaches for case-base compression can be seen as "bundling" different types of information together, to treat as a unit. Traditional per-case maintenance for case-base compression bundles together all features associated with a particular case and deletes the entirety of features associated with a particular case. In contrast, feature-bundled maintenance does an orthogonal bundling, deleting a single feature in all cases for which it appears. Flexible feature deletion can also apply an "unbundled" approach, simply deleting specific features from selected individual cases. To distinguish unbundled individual features from feature-based bundles, we call the individual features of a specific case "case-features." Figure 1 illustrates the case-bundled, feature-bundled, and unbundled approaches.

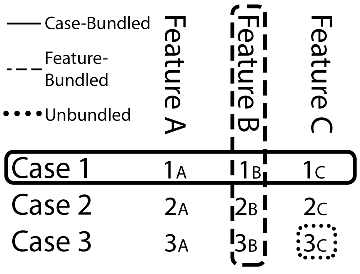

Fig. 1. Feature selection with case-bundled, feature-bundled, and unbundled strategies.

Figure 2 summarizes eleven simple candidate strategies for selecting the next case or feature to delete, spanning case-bundled, feature-bundled, unbundled, and hybrid strategies, which we describe in more detail below. Random deletion strategies are included as a baseline. The simplicity of these strategies enables comparing case-bundled and feature-bundled strategies on an equal footing. Section 6 discusses future paths for more sophisticated flexible feature deletion strategies.

Strategy	Type of Bundling	Hybrid or Non-Hybrid
Random Case-Features	Unbundled	Non-Hybrid
Random Cases	Case-Bundled	Non-Hybrid
Large Cases	Case-Bundled	Non-Hybrid
Least Coverage	Case-Bundled	Non-Hybrid
Most Reachability	Case-Bundled	Non-Hybrid
Random Features	Feature-Bundled	Non-Hybrid
Rarest Features	Feature-Bundled	Non-Hybrid
Most Common Features	Feature-Bundled	Non-Hybrid
Rarest Cases / Least Coverage	Case-Bundled	Hybrid
Rarest Features / Least Coverage	Unbundled	Hybrid
Rarest Features / Large Cases	Unbundled	Hybrid

Fig. 2. Strategies for selecting the next case, feature, or case-feature to delete

1. **Case-Bundled Strategies**
   Case-bundled strategies follow the traditional CBR compression approach of removing entire cases, i.e., the bundle of features determined by the case. A key question for case deletion is the order in which to delete cases. A classic approach is to consider cases' *coverage* as the set of target problems that a case can solve, and *reachability* as the set of cases that can solve a given target problem [7]. Cases with higher coverage are considered more valuable to preserve; cases with low reachability are considered harder to replace. We consider simple strategies favoring each criterion. Another simple criterion is to include removing largest cases first (aiming to maximize size reduction).
2. **Feature-Bundled Strategies**
   Feature-bundled strategies ignore the boundaries of cases, replacing deletion of cases with deletion of common features across cases. For example, in a movie recommendation domain, one feature might be the presence of a particular (little-known) individual; if that was unimportant to recommendations, that feature could be deleted from all cases without impairing recommendation performance. We consider the baseline strategy of random deletion, a strategy of removing the most common features (which might be expected to have the least information content), and an inverse strategy of removing the rarest features (which might be expected to be useful in fewer instances).
3. **Unbundled Strategies**
   Unbundled strategies ignore the boundaries of both cases and features. Deletion need not be done uniformly on a per-case or per-feature basis; individual features may be deleted from some cases and retained in others. For example, in the movie

domain, the feature corresponding to a particular actor could be deleted only from selected cases (e.g., those in which the actor had a walk-on role). We consider only one basic unbundled strategy, removing random features of random cases.

4. **Hybrid Strategies**
   We also consider three hybrid strategies, each combining two strategies with equal weight (weightings could also be tuned). The strategies are Large Cases / Least Coverage, Rare Features / Least Coverage, and Rare Features / Large Cases. Combining two case-bundled strategies, as in Large Cases / Least Coverage, yields a case-bundled strategy, and combining two feature-bundled strategies yields a feature-bundled strategy. However, combining two differently-bundled strategies (e.g., Rare Features / Least Coverage) yields an unbundled strategy in which the scorings of the constituent parts are used to determine case-features to delete.

We note that different strategies have substantially different computational cost. Case size and feature rarity can be calculated rapidly because they do not require problem solving. However, coverage depends on the ability of a case to solve the problems associated with other cases, so requires more costly testing involving other cases in the case base.

4 Evaluation

To help understand the relationship of per-case and flexible feature deletion strategies, we tested the compression/competence tradeoff for the strategies in Table 2, across three domains. Our evaluation addresses two questions:

1. For given compression, how does the retrieval accuracy of the strategies compare?
2. How does the retrieval time change as the number of case-feature pairs decreases, and does this depend on the retrieval strategy?

We hypothesize that at higher compressions, accuracy will tend to decrease for all strategies, but that non-case-bundled maintenance strategies will outperform case-bundled strategies. We also hypothesize that, as the total number of features decreases, retrieval time will decrease as well, with decreases roughly independent of strategy used.

4.1 Test Data

Tests used three data sets, from movie, legal, and travel domains. Movie data was drawn from the Internet Movie Database (IMDb).[1], in which each case is a film or television show, and each feature is an actor in that film or show. The sample contained 100,000 case-feature pairs in 74,720 cases with 38,374 features.

Legal data was extracted from the LegiScan database on the 113th session of the United States Congress.[2] Each case is a bill, and each feature is a sponsor or co-sponsor of a bill. The sample contained 50,000 case-feature pairs in 7,785 cases with 552 features.

[1] http://www.imdb.com/interfaces

[2] https://legiscan.com/

Travel data was the CBR Wiki travel package case base.[3] Each case is a travel package and features are the types, prices, regions, etc. This case base contains 14,700 case-feature pairs in 1,470 cases, with 2,902 distinct feature-value pairs.

All features for the IMDb and law domains are Boolean; features correspond to the presence of a particular actor in a film or sponsor of a bill. The features for the travel domain are key-value pairs, which were treated as Boolean features based on whether a particular pair was present.

4.2 Indexing and Similarity Criteria

In the experiments, when features were deleted from case content, the corresponding indices were deleted as well, keeping indices and case content synchronized.

Case similarity was calculated by Jaccard similarity of case-features. For calculating competence, problems were considered to be solved successfully if the system was able to retrieve a case for which the Jaccard similarity of features exceeded 50%. Additional tests were run for a scenario assuming minimal shared coverage, in which cases were considered to cover only with the closest adjacent case in the original case base, so successful retrieval was defined as the system retrieving the same case retrieved during the initial leave-one-out testing. Results were similar in both conditions. For reasons of space, only the results for traditional similarity are reported here.

4.3 Hybrid Strategies

The hybrid strategies in the experiments rank cases by summing normalized scores corresponding to each of their constituent strategies. The score assigned to a case for Large Cases is the size of that case divided by the size of the largest case in the case base. The coverage score assigned to a case for Least Coverage is the coverage of the case divided by the maximal case coverage. The score for Rare Features is based on the commonality of the feature, defined as the number of cases that contain that feature divided by the number of cases containing the maximally common feature in the case base; rarity of a feature f is $1 - commonality(f)$.

4.4 Evaluation Procedure

The evaluation first establishes baseline performance by leave-one-out testing for the entire case base. Next, performance is tested for compression to nine different case base sizes, ranging from 90% to 10% of the case base. For each test, the entire original case base is used as test problems, and a test problem is considered solved if there exists in the compressed case base a case (other than the test case) within the 50% similarity threshold.

When compressing the case bases, if the desired number of case-feature pairs does not fall exactly on a boundary between cases, then the single case in which this division falls is unbundled to delete features within a case.

[3] http://cbrwiki.fdi.ucm.es/mediawiki/index.php/Case_Bases

4.5 Experimental Results

Figures 3-5 show accuracy after each round of maintenance. The graphs compare the eleven strategies across the IMDb, law, and travel domains. For readability, the graphs are divided into three parts with the same horizontal and vertical scales. The third graph compares the best four strategies from the other two graphs. Each type of bundling has a different type of connecting line. Solid lines indicate case-bundled strategies, dashed lines indicate feature-bundled strategies, and dotted lines indicate unbundled strategies.

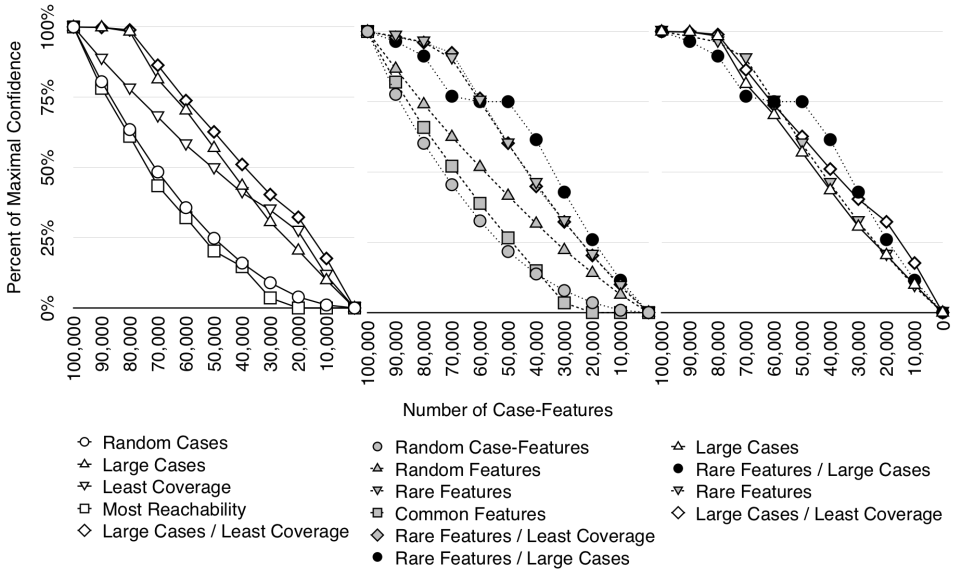

Fig. 3. Competence retention for varying compression levels for the IMDb case base

Figure 3 shows results for the IMDb data, for which the best four strategies were Large Cases, Rare Features / Large Cases, Rare Features, and Large Cases / Least Coverage. Three of the best strategies consider the size of the cases, which supports having maintenance consider not only the benefit of retaining a case (its solution coverage) but also its storage cost. Two of the best strategies are hybrid strategies, and two are non-case-bundled. The worst strategy was Most Reachability.

Given the established importance of coverage, that Least Coverage is outperformed by Large Cases on the IMDb and law data sets might seem surprising, but this is explained by the substantial case size variation in these domains. For example, the IMDb case base includes multi-episode soap operas such as *The Bill*, which span hundreds of actors but also include numerous relatively unknown actors who never appear widely.

Figure 4 shows results on the law data, for which the best four strategies were Large Cases / Least Coverage, Large Cases, Rare Features / Large Cases, and Most Reachability. Note that these overlap with three of the best strategies on the IMDb case base

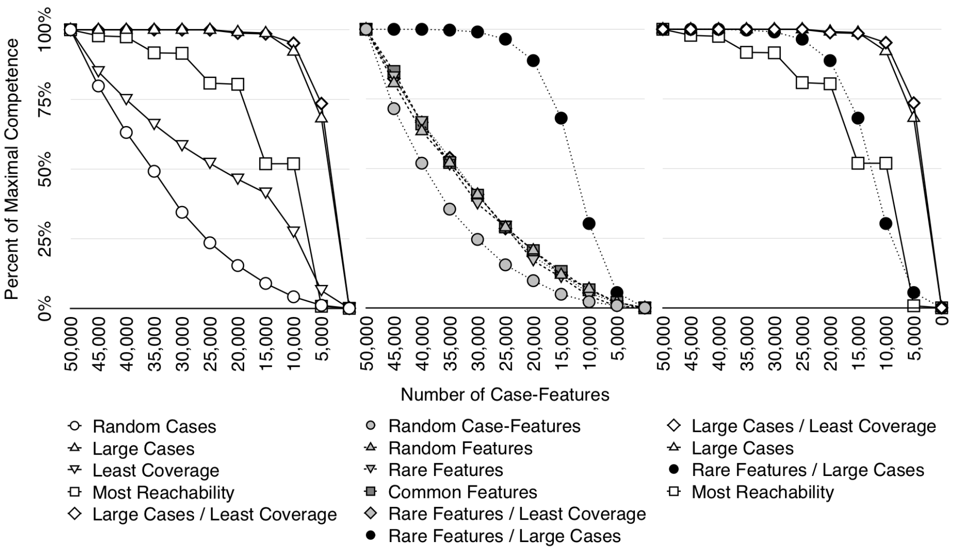

Fig. 4. Competence retention for varying compression levels for the law case base

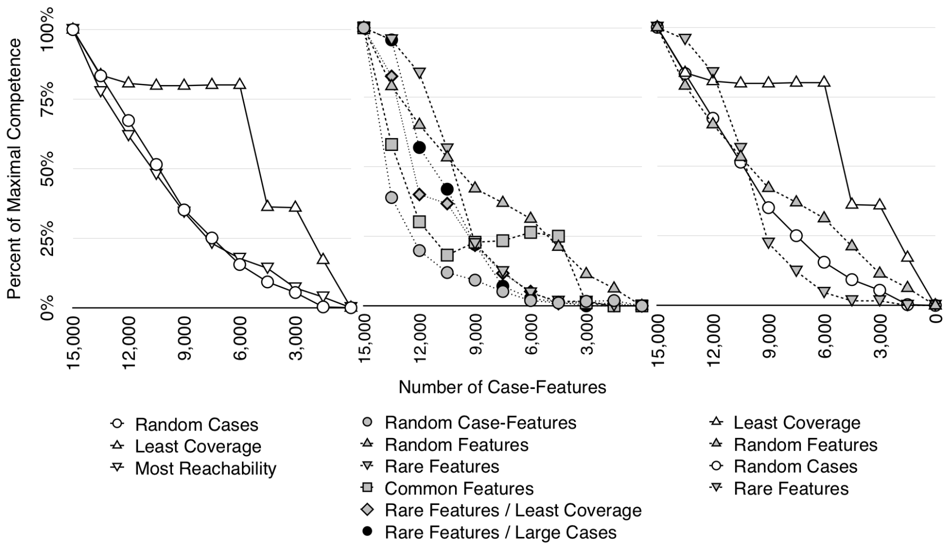

Fig. 5. Competence retention for varying compression levels for the travel case base

but in a different order. The worst strategy was Random Case-Features. The law data set has a much smaller number of features than the IMDb data set, and we hypothesize that its features are more likely to have comparable importance, making random deletion more likely to remove significant content.

Figure 5 shows results for the travel data. Because all cases are initially the same size, the strategies Large Cases and Large Cases / Least Coverage do not apply, so are omitted from the graphs. However, the hybrid strategy Rare Features / Large Cases is still applicable, because as the Rare Features strategy deletes features, only cases with those features will be compacted, resulting in different case sizes. The best strategies were Least Coverage, Random Features, Random Cases, and Rare Features. That deleting cases with least coverage is best is consistent with the key role coverage has has been ascribed in case-base maintenance research. That Random Features is second is surprising, but could be explained if many features in this domain have comparatively low information content. Although Rare Features is one of the top four strategies, its performance is quite poor, which could correspond to rare features tending to be important for distinguishing relevant cases. As with the other two data sets, two of the best strategies were non-case-bundled. However, in contrast, none of the best strategies were hybrid.

4.6 Retrieval Speed

Figure 6 compares the retrieval times after each round of maintenance for each of the four best strategies for the IMDb case base, for retrieval from a MySQL database. It also includes Random Case-Features as a baseline. The Average line shows the mean retrieval time of the five strategies in the graph. All tests were run on a MacBook Pro with a 2.5 GHz Intel Core i5 processor and 8 GB of RAM.

Random Case-Features, the baseline, gave the best retrieval times, and Rare Features, the only feature-bundled strategy, gave the worst. Both of the case-bundled strategies, Large Cases and Large Cases / Least Coverage yield similar retrieval times, but the two unbundled strategies, Rare Features / Large Cases and Random Case-Features, yield very different retrieval times. Most of the strategies have a fairly linear decline, but Rare Features declines slowly until the 10,000 case-feature mark where it drops abruptly. Because the retrieval function uses Jaccard similarity, retrieval time depends on the number of case features in the intersection between cases. However, the rarest features would seldom fall into any intersections, which explains why removing them has the least effect on retrieval time.

5 Related Work

Case-based reasoner maintenance [3] is an active area of CBR research. Much of this work develops methods to compress the case base, such as competence-based case deletion [4], deletion methods taking class boundaries into account by considering local complexity [12], optimizing the tradeoff between size and accuracy [13], deletion aimed at preserving diversity [14], and strategies for case retention and forgetting (e.g., [5, 15–17]. Such methods differ from flexible feature deletion that they retain or delete entire cases without adjusting case contents.

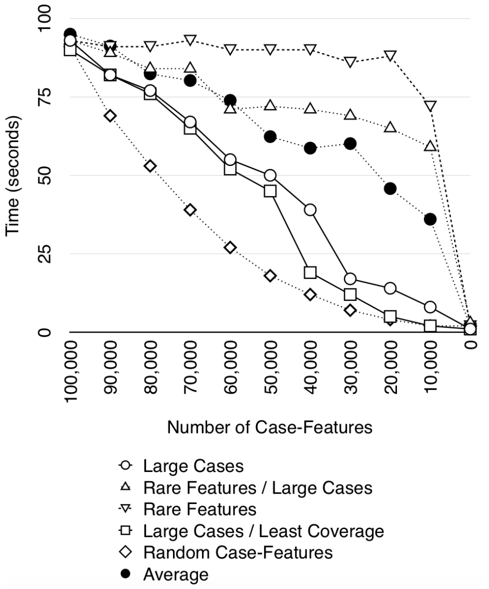

Fig. 6. Comparison of the retrieval times after each round of maintenance between the four best strategies on the cinema data set

Research on maintenance of case contents has generally focused on quality improvement rather than case base compression (e.g., [18, 19]). However, research on case abstraction research, in aiming to compact the case base by removing concrete cases subsumed by abstractions [20], can be seen as in the spirit of replacing cases with more compact versions.

Flexible feature deletion applies to indexing features as well as cases. Maintenance of indexing features has been extensively studied in CBR, applying methods such as feature deletion, addition, and reweighting, but again with the goal of improving retrieval accuracy rather than decreasing the amount of storage required for the indices themselves (e.g., [21–24]). Feature set reduction has been combined with case selection, to improve accuracy while compressing the case base [25].

6 Future Research Questions for Feature Deletion

The feature deletion approach raises a rich range of questions for fully exploiting its potential. A key question is how to develop knowledge-based feature deletion rules, especially for flexible feature deletion for complex structured cases. Other questions include how feature deletion strategies should interact with the indexing and adaptation knowledge containers, how feature deletion can preserve case integrity, and how feature deletion should be reflected in case provenance and explanation.

- *Coupling feature deletion with index maintenance:* As case contents are deleted, the relevance of case indices may change. Consequently, feature deletion may need to be accompanied by index maintenance to assure that as cases are compressed the system still retrieves the most similar cases. Feature weight information might be used to suggest features which could be deleted with limited harm.
- *Benefiting from the relationship of feature deletion to case adaptation:* Feature deletion can be seen as a form of "before the fact" adaptation of cases, in which the adaptation is driven not by a new problem to solve, but by a combination of (1) compression goals, and (2) performance goals. Richer feature deletion methods could draw on a CBR system's adaptation procedures to perform operations beyond simple deletion of case components, such as abstractions or substitutions of alternatives requiring less space. Enabling such methods requires reasoning about the competence effects of replacing a case with various candidate adapted versions, as well as performance effects (whether replacing a case with a given compressed version will decrease problem-solving speed), and the balance to strike between them.
- *Maintaining case integrity despite feature deletion:* Another question is the relationship of feature deletion to the cohesiveness of a case. From the early days of CBR, an argument for CBR has been that cases can implicitly capture interactions among case parts. Deleting portions of a case risks some of that cohesion, making it a concern to address in feature deletion strategies. That case adaptation faces the same risks but is effective supports optimism for some levels of compression, and research on hierarchical CBR has supported the usefulness of sometimes considering subparts of complete cases individually. However, how much compression can be done without excessive harm to case integrity, and how to manage the process to avoid such harm, are interesting questions.
- *Reflecting feature deletion in provenance and explanation:* Because feature deletion results in stored cases which differ from the cases originally captured, it (like case adaptation) may weaken the ability to justify proposed solutions by past experience. Likewise, changes from the original cases may make it difficult to apply provenance-based methods for predicting solution characteristics such as solution accuracy and trust (e.g., [26]). Addressing these complications might require maintaining records of case maintenance process as part of the provenance trace used for explanation, as well as reasoning about (and presenting to users) information about the parts of the case which have been affected by feature maintenance.

7 Conclusion

This paper has proposed a new case-base maintenance approach, flexible feature deletion, which questions the assumptions that cases are of uniform size and that maintenance must treat cases as unitary objects. Flexible feature deletion enables selective deletion of case contents rather than restricting deletion to the case level. It has illustrated tasks for which flexible feature deletion may be desirable, such as domains in which reasoning can be done with different amounts of information, in which flexible feature deletion enables selectively compressing different parts of different cases. Its experimental results show that case-based maintenance may need to change when case contents are non-uniform; in such settings feature-based strategies may give better accuracy than per-case strategies, and that total case-base size and retrieval times may not always be aligned, giving a space/time tradeoff which it may be possible to exploit.

The paper focuses primarily on knowledge-light maintenance strategies. Interesting future directions are to refine the strategies tested here with additional knowledge, for example, leveraging case adaptation knowledge, and to explore when other knowledge-light techniques for compression of cases and feature bundlings could yield useful maintenance strategies.

References

1. Francis, A., Ram, A.: Computational models of the utility problem and their application to a utility analysis of case-based reasoning. In: Proceedings of the Workshop on Knowledge Compilation and Speed-Up Learning. (1993)
2. Smyth, B., Cunningham, P.: The utility problem analysed: A case-based reasoning perspective. In: Proceedings of the Third European Workshop on Case-Based Reasoning, Berlin, Springer (1996) 392–399
3. Wilson, D., Leake, D.: Maintaining case-based reasoners: Dimensions and directions. Computational Intelligence **17**(2) (2001) 196–213
4. Smyth, B., Keane, M.: Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems. In: Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence, San Mateo, Morgan Kaufmann (1995) 377–382
5. Muñoz-Avila, H.: A case retention policy based on detrimental retrieval. In: Proceedings of ICCBR-99. (1999)
6. Zhu, J., Yang, Q.: Remembering to add: Competence-preserving case-addition policies for case base maintenance. In: Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, Morgan Kaufmann (1999) 234–241
7. Smyth, B., McKenna, E.: Building compact competent case-bases. In: Proceedings of the Third International Conference on Case-Based Reasoning, Berlin, Springer Verlag (1999) 329–342
8. Wilson, D., O'Sullivan, D.: Medical imagery in case-based reasoning. In Perner, P., ed.: Case-Based Reasoning on Images and Signals. Volume 73 of Studies in Computational Intelligence. Springer (2008) 389–418
9. Leake, D., Maguitman, A., Reichherzer, T.: Experience-based support for human-centered knowledge modeling. Knowledge-based systems **68** (2014) 77–87
10. Novak, J., Gowin, D.: Learning How to Learn. Cambridge University Press, New York (1984)
11. Leake, D., Maguitman, A., Reichherzer, T., Cañas, A., Carvalho, M., Arguedas, M., Brenes, S., Eskridge, T.: Aiding knowledge capture by searching for extensions of knowledge models. In: Proceedings of the Second International Conference on Knowledge Capture (K-CAP), New York, ACM Press (2003) 44–53
12. Craw, S., Massie, S., Wiratunga, N.: Informed case base maintenance: A complexity profiling approach. In: Proceedings of the Twenty-Second National Conference on Artificial Intelligence, AAAI Press (2007) 1618–1621
13. Lupiani, E., Craw, S., Massie, S., Juárez, J.M., Palma, J.T.: A multi-objective evolutionary algorithm fitness function for case-base maintenance. In: Case-Based Reasoning Research and Development - 21st International Conference, (ICCBR-13), Springer 218–232

14. Lieber, J.: A criterion of comparison between two case bases. In: Advances in Case-Based Reasoning. Volume 984 of Lecture Notes in Computer Science. Springer, Berlin (1995) 87–100
15. Ontañón, S., Plaza, E.: Collaborative case retention strategies for CBR agents. In: Case-Based Reasoning Research and Development: Proceedings of the Fifth International Conference on Case-Based Reasoning, ICCBR-03, Berlin, Springer-Verlag (2003)
16. Romdhane, H., Lamontagne, L.: Forgetting reinforced cases. In: Advances in Case-Based Reasoning, 9th European Conference, ECCBR 2008, Trier, Germany, September 1-4, 2008. Proceedings. (2008) 474–486
17. Salamó, M., López-Sánchez, M.: Adaptive case-based reasoning using retention and forgetting strategies. Know.-Based Syst. **24**(2) (March 2011) 230–247
18. Racine, K., Yang, Q.: Maintaining unstructured case bases. In: Case-Based Reasoning Research and Development, ICCBR 1997, Berlin, Springer (1997) 553–564
19. Salamó, M., López-Sánchez, M.: Rough set based approaches to feature selection for case-based reasoning classifiers. Pattern Recognition Letters (2011) 280–292
20. Bergmann, R., Wilke, W.: On the role of abstraction in case-based reasoning. In: Advances in Case-Based Reasoning, Third European Workshop, EWCBR-96, Lausanne, Switzerland, November 14-16, 1996, Proceedings. (1996) 28–43
21. Arshadi, N., Jurisica, I.: Feature selection for improving case-based classifiers on high-dimensional data sets. In: Proceedings of the Eighteenth International Florida Artificial Intelligence Research Society Conference (FLAIRS-05), AAAI Press (2005) 99–104
22. Fox, S., Leake, D.: Learning to refine indexing by introspective reasoning. In: Proceedings of First International Conference on Case-Based Reasoning, Berlin, Springer Verlag (October 1995) 431–440
23. Muñoz-Avila, H.: Case-base maintenance by integrating case-index revision and case-retention policies in a derivational replay framework. Computational Intelligence **17**(2) (2001) 280–294
24. Zhang, Z., Yang, Q.: Towards lifetime maintenance of case base indexes for continual case based reasoning. In: Artificial Intelligence: Methodology, Systems, and Applications. Springer (1998) 489–500
25. Li, Y., Shiu, S., Pal, S.: Combining feature reduction and case selection in building CBR classifiers. IEEE Transactions on Knowledge and Data Engineering **18**(3) (March 2006) 415–429
26. Leake, D., Whitehead, M.: Case provenance: The value of remembering case sources. In: Case-Based Reasoning Research and Development: Proceedings of the Seventh International Conference on Case-Based Reasoning, ICCBR-07, Berlin, Springer-Verlag (2007) 194–208