

## Adaptation Cost as a Criterion for Solution Evaluation

Juho Rousu and Robert J. Aarts

VTT Biotechnology and Food Research  
P.O.Box 1500, FIN-02044 VTT, Finland  
{Juho.Rousu,Robert.Aarts}@vtt.fi

**Abstract.** In most case-based systems, the best case is selected by calculating similarities between the surface features of the cases. Sometimes, the most similar case is not the one that is the easiest to adapt. On the other hand, judging the adaptation effort can be tedious. In this paper, we propose case-adaptability to be used as an evaluation criterion as opposed to using it in the case-retrieval phase. Our method can be seen to combine the computational effectiveness of similarity-based retrieval with the better quality solutions of adaptation-guided retrieval.

### 1. Introduction

The traditional approach to the case-based reasoning has been to retrieve the best case from the casebase by using the surface features associated with the cases. The technique of nearest-neighbor matching is employed in many CBR systems (Kolodner 1993). The case that is deemed to be the nearest to the problem at hand, is selected as the ball-park solution, which is then adapted. An implicit assumption that is present in such methods is that the most similar case in the casebase is also the most adaptable. As noted by Smyth and Keane (1995), this assumption is not always valid; it may even be impossible to adapt the proposed ball-park solution. Similarity metrics often assume that the surface features are independent. This assumption is rarely true, hence interdependencies between surface features are left unnoticed. Unnecessarily large adaptations may result when the goodness of the cases is judged by similarity metrics alone.

In case-based planning systems, adaptation is always a step into an unknown neighborhood. The plan of the ball-park solution can be seen as a ‘safe’ point in the search space; the outcome of the case is known. Any adaptation to the plan increases the uncertainty about the feasibility of the plan to the new situation. Making large adaptations to the plan reliably requires that our model of the domain is very good. Most often, this kind of models cannot be constructed. Instead, we must resort to incomplete and imprecise models. On the other hand, if a complete and precise model of the domain is available, there is no need for CBR or any other AI methods at all!

Independent of the nature of the model used for adaptation, adapting as little as possible seems to be the natural way to go. The model, even how weak it may be, is likely to incorporate domain knowledge that is useful for adaptation. In contrast, the similarity metrics rarely take any specificities of the application domain into account. Hence, basing the selection of the ball-park case on the required amount of

adaptation seems to be a more sound approach than relying on a surface-similarity based method.

An ideal way to take adaptability of the cases into account is to use adaptation-guided retrieval (AGR) as suggested by Smyth and Keane (1995). AGR methods are particularly effective when prediction of the adaptability can be done significantly more quickly than actually performing the adaptations. In such situations it is possible to quickly filter out all the cases that are difficult to adapt or non-adaptable. Obviously, efficient use of AGR requires explicit adaptation knowledge and well-defined criteria for adaptability.

Unfortunately, the above conditions are not fulfilled in the bioprocess planning domain (Aarts 1992, Aarts & Rousu 1996) considered in this paper. In this domain we typically have casebases that are relatively homogeneous on the surface, since the case features consist of standard measurements and specifications, and the plans are constructed from a limited set of actions. Hence, we typically have large numbers of potentially adaptable cases. However, below the surface, the bioprocess planning domain is very complex and the planning problems are hard to decompose: a minor adaptation to the case for fixing one problem may cause several other problems. Consequently, predicting the adaptability is essentially as difficult as actually adapting the cases. In a domain like that, calculating adaptability scores for cases that are totally dissimilar to the new case seems to be an unnecessary large effort; in any case we expect that the distances between surface feature values correlate to some extent with adaptability.

In this paper we propose a case-based planning method that combines similarity-based retrieval with evaluation based on *adaptation cost*. In our method, similarity-based retrieval is used for ordering the cases. An adaptation algorithm then calculates the adaptation cost for each case, in decreasing order of similarity, until a special *stopping criterion* is reached. The plan that was adapted by the least amount is returned as the solution. A necessary component for our approach is defining the adaptation cost of a plan. We base our method on *edit distance* (Wagner & Fischer 1974), a measure that defines the distance between two sequences in terms of *edit operations*, such as *insertions*, *deletions* and *changes*. Since the plans we consider are action sequences, edit distance fits the scheme well.

As application domain we consider bioprocess planning. The applicability of our method, however, is in no way restricted to that domain. The aim in bioprocess planning is to make end-products of constant quality even if the raw materials and production environment varies. Complete recipes, including the ingredients, the production environment, the used process parameter profiles and the product specification, are stored as cases. The cases are automatically adapted using a semi-qualitative model of the bioprocess.

The remainder of this article is organized as follows. In section 2 we describe the domain that we are interested in, namely batch-wise planning of bioprocesses. In section 3 we describe our method for evaluating cases in general level, in section 4 we define the distance measures and in section 5 we present some preliminary experiments.

## 2. Bioprocess Planning Using Case-Based Methods

A usual goal in bioprocess planning is to obtain products that are of as even quality as possible. Achieving this goal is hindered by several factors. The most common cause of difficulty is that the quality of raw materials varies. Another one is that production schedules may force the plant to use different equipment at different times. These changes in the production environment cause variations in the quality of the product.

To compensate the variations in the production environment, the process should be somehow adapted. On-line control of bioprocesses is difficult for two reasons: Firstly, bioprocesses tend to be sensitive to their past. Failures cannot be fixed after they have happened. Secondly, adequate on-line sensors are not available in many cases. Therefore, the process must be planned in advance, and the potential problems must be taken into account before the batch is started.

From a planning point of view, bioprocess recipes are quite difficult. A minor change to the value of some parameter may result in large, undesired changes in the outcome in the process. For that reason, it is typical of the bioprocess industry that process recipes are not changed very often; recipe modification is perceived as risky. In other words, the plants do not optimize their recipes. Only when severe problems with several batches have occurred, the recipe is changed. Consequently, systematic adaptation knowledge is lacking and recipe adaptation ends up to be an iterative trial and error process.

On the other hand, production data is usually recorded systematically. Hence, ample process experience in the form of successful and unsuccessful recipes is available in corporate databases. As should be clear from this discussion, case-based planning fits this scheme very well: as theoretical knowledge of the process is lacking, building and tuning a model-based planner can be a tedious task. In contrast, a casebase is rather easy to construct, since the production batches are usually well monitored and analyzed, and the measurements are stored in databases anyway.

For recipe adaptation, a large body of biotechnological information is applicable to many processes. An approximate, qualitative model (Cohn 1989) of the bioprocess planning domain<sup>1</sup> can be constructed from this knowledge. Obviously, the model is process specific and has to be constructed anew for each different bioprocess. With the case-based approach we can manage with such a general, qualitative, model of the process. Since the system learns its environment from recipes that were used in production, the model does not need to be as complete as it would otherwise have to be. On the other hand, the suggested adaptations are not so accurate as the ones given by a carefully tuned quantitative model would be. For that reason, avoiding unnecessarily large adaptations is important. The aim of this paper is to develop methods for quantifying the adaptability, or adaptation cost, of the cases and using the methods effectively in terms of computational resources.

---

<sup>1</sup> See Aarts & Rousu (1996) for a description of this model.

### 3. Evaluation of Cases Using Adaptation Cost

In an industrial setting the casebase consisting of different kinds of recipes may be large. Calculating adaptations for all the cases is clearly not possible. Instead, a filtering scheme is needed. In adaptation-guided retrieval (Smyth & Keane 1995), the filter is based on predicting the adaptability of the cases. This is not feasible in bioprocess planning, since there does not seem to be a quick way of judging the adaptability: Typically, most of the cases in the casebase are adaptable, only the amount of adaptation needed varies. Rapid prediction of the amount of adaptation needed does not seem to be possible. Hence, a different filtering mechanism is needed. We base our filter on the similarity of surface features.

Our planning algorithm is outlined in figure 1. We use the similarity scores to sort the cases. The most similar case is subjected to adaptation first. Remaining cases are treated in the same way in descending order of similarity. A user defined stopping criterion is used; When adaptation cost rises over the best cost found by a fraction  $\delta$ , the adaptation-performing loop is exited and the cheapest plan in terms of adaptation is returned. Alternatively, the user has always the option to exit the search any time; the best plan that was found so far is returned as the solution.

Although adaptation of the plans using the qualitative model is more difficult than calculating similarities, that is not a problem for our algorithm, since it typically examines only a couple of cases in the adaptation and evaluation phase. Since the casebase is not examined in full, we are not guaranteed to find the case with the

*ConstructPlan(newCase, CaseBase,  $\delta$ )*

1. **For each** *oldCase*  $\in$  *CaseBase* **do begin**
2.     Compute a similarity score  $s(\text{newCase}, \text{oldCase})$
3. **end;**
4. Sort the *CaseBase* by the calculated similarity scores.
5. *bestPlan* :=  $\emptyset$ , *bestCost* :=  $\infty$
6. **For each** *oldCase*  $\in$  *CaseBase* **do begin**
7.     *newPlan* := *AdaptPlan(oldCase, newCase)*.
8.     Calculate adaptation cost  $d(\text{newCase}, \text{oldCase})$ .
9.     **If**  $d(\text{newCase}, \text{oldCase}) < \text{bestCost}$  **then**
10.         *bestPlan* := *newPlan*
11.         *bestCost* :=  $d(\text{newCase}, \text{oldCase})$
12.     **Else If**  $(d(\text{newCase}, \text{oldCase}) > (\text{bestCost}) \times (1 + \delta))$
13.         **or** (user wants to quit) **then break;**
14.     **end.**
15. **Return** *bestPlan*.

**Figure 1:** A planning algorithm with evaluation based on adaptation cost.

absolutely best score. Since the most similar cases are always evaluated, we expect to obtain good quality recipes, nonetheless. In comparison to an algorithm that picks only the most similar case for adaptation, we always obtain at least as good solutions and can do significantly better in some situations.

#### 4. Measuring Adaptation Cost

In the bioprocess planning domain, the plan is simply a sequence of actions. Each action is an instance of an *action type*. An action type has an associated set of parameters that define the action. Hence, each action is defined by two aspects: its type and a vector of *action parameter* values. For most action types the parameters are real numbers.

The bioprocess plans that we consider consist of two types of actions: Most actions are targeted to alter reactor conditions such as *temperature*. The second type of actions are additions of some ingredients to the reactor.

As the plans are sequences of actions, it makes sense to define the distance between two plans in terms of *edit distance* (Wagner & Fischer 1974), a measure developed by the string matching community to quantify the distance of two strings. The edit distance is computed as a minimal sum of *edit operations*, namely *insertions, deletions and changes*, that are required to transform a sequence into another. Relatively efficient dynamic programming algorithms for computing the edit distance exist, see e.g. Ukkonen (1985). Note that these algorithms are not needed in our application. The qualitative model is used for generating the adaptation operations and there is no need to search for them using dynamic programming.<sup>2</sup>

In our planning problem, an original plan and the adapted plan are the two ‘strings’ to be compared. The actions are the ‘characters’. Hence, the edit operations in this context are

- *change* of the parameters of an action, or a change of an action to another (denoted by  $a \rightarrow b$ ),
- *insertion* of an action into plan ( $\emptyset \rightarrow b$ ), and
- *deletion* of an action from the plan ( $a \rightarrow \emptyset$ ).

The cost of inserting and deleting an action is the same:  $cost(\emptyset \rightarrow b) = cost(a \rightarrow \emptyset) = 1$ . The cost of changing an action to another depends on the action types:  $cost(a \rightarrow b) = 1$ , if  $a$  and  $b$  are of different types, and  $cost(a \rightarrow b) = m/n$ , when the actions are of the same type;  $m$  is the number of parameters with different values for  $a$  and  $b$ ,  $n$  is the total number of parameters for the action type. The sum of the costs of the individual adaptations is the adaptation cost of the plan.

The above defined measure is unbiased with respect to the number of parameters of the action type. Changing, for example, the values of half of the action parameters costs always 0.5, independent of the total number of parameters.

One could object the simplicity of the measure; it does not take into account the magnitudes of the parameter changes. Indeed, some quantitative properties could be embedded to the measures. We must keep in mind, though, that since our model is

---

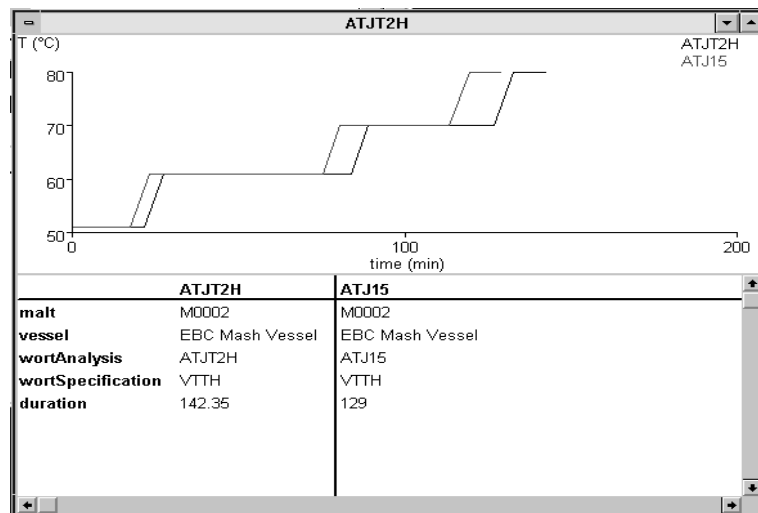
<sup>2</sup> These algorithms could, however, be used as a method for organizing the casebase, by placing similar plans in the proximity of one another.

qualitative, the adaptation magnitudes suggested by the model cannot possibly be very accurate. Hence, exact measurement of the adaptation magnitudes seems somewhat inappropriate. Furthermore, a simple quantitative measure would not necessarily make things better; an adaptation that seems large may actually affect the process less than another adaptation that seems small. Another problem in defining distance metrics is that adaptations of the early stages often have greater effect on the outcome of the process than adaptations of the later stages. Thus, it seems to be necessary to utilize the available domain knowledge in the design of more elaborate distance measures.

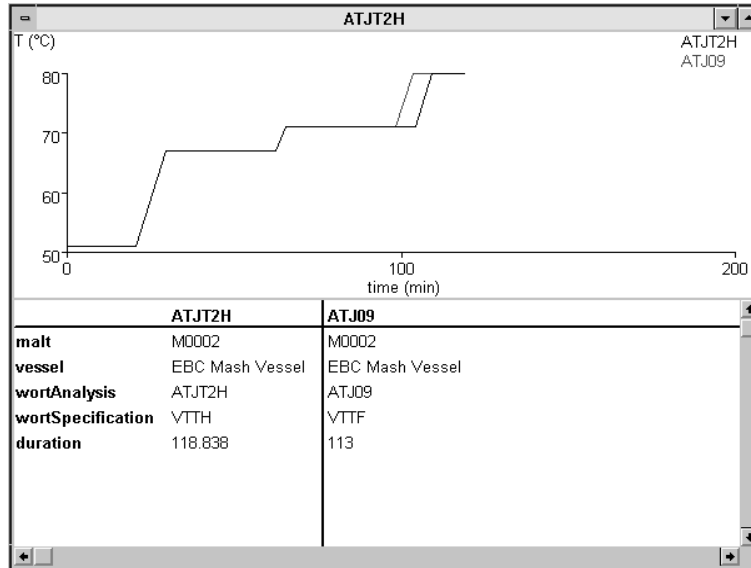
## 5. Empirical Evaluation

We have implemented the adaptation-based evaluation method in a bioprocess planning system. In a preliminary experiment we compared the new method to the performance of a similarity-based retrieval method. The casebase consisted of some 15 recipes for mashing, a process that is the first step in the production of beer.

The results of the preliminary experiments were encouraging. The new method was able to find cases that required less adaptations than what the most similar case required. An example of this situation is depicted in figures 2 and 3: In figure 2, the adapted plan for the new recipe ATJT2H is shown against the plan of the most similar case in the casebase, ATJ15. Three parameter changes were suggested by the qualitative model, causing a total adaptation cost of 1.5. Figure 3 depicts the situation in which the plan was adapted from a less similar case ATJ09. Only one parameter-change, of cost 0.5, was required. In all three cases the raw material, malt, was the same. The specifications for the product, the wort, were identical for the



**Figure 2.** The plan for recipe ATJT2H when adapted from the most similar case ATJ15.



**Figure 3.** The plan for recipe ATJT2H, adapted from the plan of most adaptable case ATJ09.

cases ATJ15 and ATJT2H, the test case. The case ATJ09 had a different wort specification.

In this case, one explanation for this excellent performance is that the case ATJ09 was, in fact, somewhat unsuccessful; the end-product, wort, contained too little sugars compared to the specification VTTF. However, the wort would have met the corresponding requirement in specification VTTH, the specification associated with the case ATJT2H, quite well. Hence only one adaptation is needed.

The fact that the outcome of the case ATJ09 is very good with respect to VTTH is left unnoticed by the retrieval algorithm, as it only compares the similarities of the situations. The retrieval algorithm prefers the case ATJ15 which has the same raw materials and the same wort specification. As the resulting wort of case ATJ15 was not perfectly in line with the specifications, as many as three adaptations were suggested by the domain model in an attempt to get rid of the deviations.

## 6. Conclusions and Future Work

Similarity metrics have been studied rather extensively in the context of case retrieval. The potential of a similar approach for the solution evaluation phase has, however, been left unrealized. We developed a method that tries to quantify the adaptability by using an edit distance metric as means to measure the adaptation cost. We use the metric in case evaluation: the best recipe is the one requiring the least amount of adaptation.

Our method makes a compromise between the computational efficiency of the similarity-based retrieval methods with the good quality solutions of adaptation-guided retrieval. By setting the stopping criterion for the evaluation algorithm appropriately, the balance between quality and speed can be controlled by the user.

Obviously, as the casebase grows, more and more cases will be examined before the stopping criterion will terminate the adaptation-performing loop, provided that the threshold  $\delta$  is kept constant. This may potentially have an effect on the efficiency of the algorithm. Case generalization techniques may be used to circumvent this problem. Note that tightening the threshold  $\delta$  as the casebase grows is not a good idea, since the really alternative ball-park cases will probably drop out.

The preliminary experiments show that the method can actually find better quality solutions in terms of required amount of adaptation. Comprehensive tests in laboratory-scale are still needed, though, in determining the actual benefits to recipe quality. Also, the effect of more elaborate distance measures is still to be investigated. A possible direction is to incorporate some of the process knowledge that is contained in the qualitative model into the distance measures.

## Acknowledgments

The authors wish to thank the anonymous reviewers for their suggestions.

## References

- Aarts, R. J. (1992). *Knowledge-based systems for bioprocesses*. Technical Research Centre of Finland, Publications 120, 116 p.
- Aarts, R. J. & Rousu, J. (1996). Towards CBR for Bioprocess Planning. In this volume.
- Cohn, A. G. (1989). Approaches to qualitative reasoning. *Artificial Intelligence Rev.* **3**, pp. 177-232.
- Kolodner, J. (1993). *Case-Based-Reasoning*. Morgan-Kaufmann Publishers Inc., San Mateo CA, 668 p.
- Smyth, B. & Keane, M.T. (1995). Experiments on Adaptation-Guided Retrieval In Case-Based Design. In *Proceedings of First International Conference on Case-Based Reasoning ICCBR-95*, Veloso M., Aamodt, A., (eds.). Springer-Verlag, 1995, pp. 313-324.
- Ukkonen, E. (1985). Algorithms for approximate string matching. *Information and Control* **64**, pp. 100-118.
- Wagner, R. & Fischer, M. (1974). The string-to-string correction problem. *J. Assoc. Comput. Mach.* **21**, pp. 168-178.