Towards a New Formal Model of Transformational Adaptation in Case-Based Reasoning

Ralph Bergmann, Wolfgang Wilke
University of Kaiserslautern
Centre for Learning Systems and Applications (LSA)
PO-Box 3049
D-67653 Kaiserslautern, Germany
{bergmann, wilke}@informatik.uni-kl.de

Abstract

Although several systematic analyses of existing approaches to adaptation have been published recently, a general formal adaptation framework is still missing. This paper presents a step into the direction of developing such a formal model of transformational adaptation. The model is based on the notion of the quality of a solution to a problem, while quality is meant in a more general sense and can also denote some kind of appropriateness, utility, or degree of correctness. Adaptation knowledge is then defined in terms of functions transforming one case into a successor case. The notion of quality provides us with a semantics for adaptation knowledge and allows us to define terms like soundness, correctness and completeness. In this view, adaptation (and even the whole CBR process) appears to be a special instance of an optimization problem.

1 Introduction

Today, adaptation is still a big research issue and major challenge in CBR. Although several systematic analyses of existing approaches to adaptation have been published recently and new frameworks have been proposed (e.g. (Kolodner, 1993; Hanney et al., 1995; Voß, 1996)) most adaptation components are more or less developed in an ad-hoc fashion. For pure case retrieval systems, there is a good theory and mathematical formalization focusing on similarity measures, preference relations, and related properties of the similarity measures and retrieval algorithm like correctness and completeness (Wess, 1995). This solid foundation laid the ground for the success of such systems in practice. For CBR systems which include adaptation such a formalization is still missing.

This paper presents a step into the direction of developing a formal model of transformational adaptation. The model is based on the notion of the quality of a solution to a problem, while quality is meant in a more general sense and can also denote some kind of appropriateness, utility, or degree of correctness. Adaptation knowledge is then defined in terms of functions transforming one case into a successor case. We will see that the notion of quality provides us with a semantics for adaptation knowledge and allows us to define terms like soundness, correctness and completeness. In this view, adaptation (and even the whole CBR process) appears to be a special instance of an optimization problem.

2 Basic Terms

We first introduce the three basic terms: *problem*, *solution*, and *quality*. These terms are used to characterize the task which the CBR system has to solve and builds the foundation of our model.

Notions of Problem, Solution and Quality

Let P be the (possibly infinite) problem space and let S be the (possibly infinite) solution space to be considered. We don't make any assumptions about a structure of the elements from P and L.

A basic notion that we now introduce is that of the *quality* of a solution $s \in S$ with respect to a problem $p \in P$. We introduce Q to be a total function: $Q : P \times S \rightarrow \mathcal{R}$ called *quality function*. The symbol $\mathcal{R}$ denotes a (possibly infinite) set of ordered elements, usually the set of real numbers[1]. This quality function assigns a quality value to each problem-solution-pair. We assume that a larger value reflects a higher quality. The meaning of this quality value is that solutions with a higher quality are prefered over solutions with a lower quality value. This notion of quality can express the appropriateness of a solution to a problem, the utility of a solution for a problem, or the degree of correctness of a solution to the problem. Instead of having just a binary notion of correct

[1]Sometimes it is useful to allow also $-\infty$ and ∞ as results of the quality function.

or false (like for example in classical planning), we allow a more fine-grained measurement of the solution.

The Problem Solving Task

We can now formally specify the problem solving task as follows. Given a problem $p \in P$ as input, the task is to determine a solution $s \in S$ as output such that $\forall s' \in S \, Q(p,s') \leq Q(p,s)$ holds. This means that we are looking for the solution with the highest quality with respect to the quality function Q. If we have just a binary quality function, this specification states that we are looking for a correct solution. Then Q states what correct means. Please note that Q contains the whole specification of this problem solving task. Thereby, it provides the *semantics* for the problem solving process.

From what is stated till now, one might think about solving this problem by using an optimization algorithm: searching through the space of solutions, computing the quality function Q, and selecting the best solution. However, for the kind of applications one usually has in mind for CBR, the quality function Q cannot be easily formalized or is not even known. Therefore this quality function cannot be directly used for problem solving and optimization algorithms cannot be applied.

Example

We now briefly introduce an illustrating example from the area of sales support. The goal is to sell a PC that fulfills the specific requirements of a customer. There are already several pre-configured PCs at a fixed price (these will become the cases) and additional components can be added or removed (adaptation). In this Scenario, P is the set of all possible combinations of requirements, e.g., $P = 2^{\{Textprocessing, Games, Music, \ldots\}}$, and S is the set of all possible PC configurations, i.e, sets of included components, e.g., $S = 2^{\{ASUS-Mainboard, 4.3GBHarddisk, \ldots\}}$. [2] We can now think of a quality function Q in this domain which basically reflects the price of the configured PC as follows:

$$Q(p,s) = \begin{cases} -cost & : \ s \text{ fulfills all requirements from } p \text{ and the price of } s \text{ is } cost \\ -\infty & : \ s \text{ does not fulfill all requirements from } p \end{cases}$$

A customer stating his problem (requirements) usually wants to have a solution with the highest possible quality, i.e., a working solution at the lowest price.

[2] 2^X denotes the power-set of the set X

In this scenario, classical optimization approaches are not applicable since Q is difficult to known completely. One might think about computing Q by summing up the price for all components, but this approach is not feasible if one wants to consider special offers, special prices for existing pre-configured PCs from the stock, or different prices from different distributors, etc.

3 Cases and Adaptation Knowledge

Cases

We now introduce the notion of a case. Cases encode certain knowledge about the quality function. Therefore, the usual definition of a case is extended by explicitly introducing the quality value for the particular problem-solution-pair. Consequently, a *case* is a triple $(p, s, q) \in P \times S \times \mathcal{R}$. Let $C = P \times S \times \mathcal{R}$ be the space of all cases, and the case base $CB \subset C$ is a finite set of cases.

A case representation of that kind was already suggested by Kolodner (Kolodner, 1993) (p. 147, 158ff.). Besides problem and solution, she also introduces a third component of a case called outcome. The outcome is the resulting state of the world when the solution is carried out. Her notion of outcome and our notion of quality are a feedback from the real world when trying the solution to the problem. This feedback can for example also be acquired in the revise phase of the CBR process.

Definition 1 (Soundness of case and case base) A case $c = (p, s, q)$ is *sound* w.r.t. a quality function Q iff $q = Q(p, s)$ holds. A case base CB is sound iff all its cases are sound.

So, even if the quality function is not known completely, sound cases capture the quality value for certain isolated points from the P-S-space. In many standard CBR scenarios, the quality of a problem-solution-pair is not explicitly noted as part of the case representation. It is assumed that all cases are of high quality, e.g., they represent correct solutions and consequently the quality is always 1 in case of a binary quality function.

Example (continued)

We can now consider existing pre-configured PCs (e.g. from a stock) to be cases in the case base. The quality value is known because the price for which the distributor sells the PC is known.

Adaptation Knowledge

Adaptation knowledge is also knowledge about the quality function Q. However, it is not knowledge about isolated points from Q, but knowledge about differences. Adaptation knowledge is knowledge about the gradients of Q when going into certain directions. In our model, adaptation knowledge comes in the form of *adaptation operators*. An adaptation operator is a partial function $\alpha : C \rightarrow C$, i.e., it transforms a case into some successor case. This view on adaptation knowledge extends the traditional view of transformational adaptation since we propose that the problem, the solution, and the quality value may get modified. Traditionally, only the solution of the retrieved case is modified. The adaptation knowledge container (Richter, 1995) A is a set of adaptation operators $A = \{\alpha_1, \alpha_2, \ldots\}$. We can now state what soundness of the adaptation knowledge means.

Definition 2 (Soundness of an adaptation operator, the adaptation container) An adaptation operator α is sound w.r.t. Q iff for all $c = (p, s, q) \in C$ and all $c' = (p', s', q') \in C$ holds: if $\alpha(c) = c'$ and $Q(p, s) = q$ then $Q(p', s') = q'$. The adaptation container A is sound w.r.t. to Q iff all its adaptation operators are sound.

Example (continued)

We can consider an adaptation operator for adding more hard-disk space to a PC. An operator stating that if you add a hard disk of 4.3 GB space, the PC will fulfill the requirements of database applications and the price will increase by 985 DM (increasing the price means reducing the quality) can be represented as follows:

$$\alpha_1(p, s, q) = (p \cup \{\text{DB-Applics.}\},\ s \cup \{\text{4.3 GB HardDisk}\},\ q - 985) \text{ if } q \neq -\infty$$

4 The Adaptation (Reuse) Process

Let $\hat{p}$ be the current problem. Then, the adaptation (reuse) step in CBR transforms a retrieved case $c = (p, s, q) \in CB$ into an adapted case $c' = (p', s', q') = \alpha_{i_1} \circ \cdots \circ \alpha_{i_m}(c)$, with $\alpha_{i_j} \in A$. For this adaptation process, the following simple lemma is obvious:

Lemma 1 (Soundness of adapted case) If the case base CB and the adaptation container A are sound w.r.t. Q and if the adapted case c' is computed by $c' = \alpha_{i_1} \circ \cdots \circ \alpha_{i_m}(c)$ for a $c \in CB$, then c' is also sound w.r.t. Q.

Please note that neither p nor p' must be identical to the current problem $\hat{p}$. However, the goal of the adaptation is to find a solution to the current problem $\hat{p}$. Hence, we are looking for a sequence of adaptation operators such that $p' = \hat{p}$ holds. If we can find such an adaptation sequence, then the previous lemma allows us to conclude that $Q(\hat{p}, s') = q'$; hence we know the quality of the computed solution. If we cannot achieve $p' = \hat{p}$ then we don't know anything about the solution of the problem p.

Illustration

We can think about the quality function as a plane in the 3-dimensional P-S-Q space (see Figure 1). This plane is not known to the system, but we have information about certain points on the plane (black dots indicating cases from the case base) and we have information about how we can move from one point on the plane to another point on the plane by applying an adaptation operator. Adaptation operators are shown as arrows. To solve the new problem $\hat{p}$, we start from a case $C1$ and follow the arrows until we have reached $\hat{p}$.

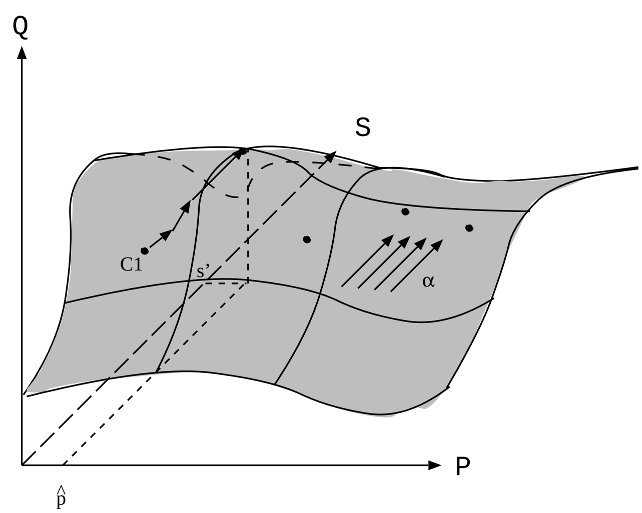

Figure 1: Quality function

If we take a look at the definition of the problem solving task as introduced before, it becomes clear that the goal of problem solving is to achieve a global optimal solution w.r.t. to the quality function Q. Since the CBR system does not have complete knowledge about Q it can not fulfill this task. If it has achieved a solution, it can never know whether there still is a different solution with a higher quality. However, a CBR system can nevertheless behave correct w.r.t. the knowledge (CB, A) it has. This leads to the following definitions.

Definition 3 (Correctness of a CBR system w.r.t. CB and A) A CBR system consisting of a sound case base CB and a sound adaptation container A is *correct* iff the following holds: if the system delivers solution s' for the problem $\hat{p}$, then there exists a retrieved case $c \in CB$ and a sequence of adaptation operators $\alpha_{i_1} \ldots \alpha_{i_m} \in A$, such that $(\hat{p}, s', q') = \alpha_{i_1} \circ \cdots \circ \alpha_{i_m}(c)$ and there does not exist $c'' \in CB$ and no other set of adaptation operators $\alpha'_{i_1} \ldots \alpha'_{i_m} \in A$, such that $(\hat{p}, s''', q''') = \alpha'_{i_1} \circ \cdots \circ \alpha'_{i_m}(c'')$ and $q''' > q'$.

So, a CBR system behaves correct, if it only computes solutions which are optimal w.r.t. the knowledge it has.

Definition 4 (Completeness of a CBR system w.r.t. CB and A) A CBR system consisting of a sound case base CB and a sound adaptation container A is *complete* iff the following holds: if, for a problem $\hat{p} \in P$ there exists a case $c \in CB$ and a sequence of adaptation operators $\alpha_{i_1} \ldots \alpha_{i_m} \in A$, such that $(\hat{p}, s', q') = \alpha_{i_1} \circ \cdots \circ \alpha_{i_m}(c)$ then the system delivers a solution (which can be different from s').

A CBR systems is complete, if it always computes a solution if one can be derived from the case base and the adaptation knowledge.

Correctness and completeness are desirable properties of CBR systems. However, it is obvious that it is very hard to achieve both properties in the general case, i.e., for arbitrary problem and solution spaces and for arbitrary adaptation operators. In the general case, one can even show that the problem of deciding whether a correct solution to a problem exist is undecidable. [3] Nevertheless, these two properties are important. We should try to find restrictions for the representation of cases and adaptation operators, such that we can guarantee correctness and completeness. We can also try to come up with some relaxed versions of these properties, e.g., in a more PAC-like manner.

5 Similarity

It might strike one's mind that we have not yet spoken about the concept of similarity, although usually considered the key concept in CBR.

[3] The undecidablity result for action planning (Bylander, 1991) can be used to show this property.

Similarity for retrieving cases

Similarity measures are usually used for selecting an appropriate case during retrieval. Because of the knowledge we have captured already in the adaptation container, a similarity measure which determines which case to retrieve does *not* encode any new domain knowledge about the quality function Q. However, it contains some kind of *control knowledge* which allows to simplify the CBR process, because we don't have to look for all cases, try to adapt them and compare the quality of the resulting solutions.

Definition 5 Correctness of similarity measure w.r.t. adaptation container A. A similarity measure $sim_R : P \times C \rightarrow [0..1]$ is correct w.r.t. the adaptation container A iff the following condition holds: $sim_R(\hat{p}, c) = f(q')$ for a bijective monotonous function $f : \mathcal{R} \rightarrow [0..1]$ if there exists a sequence of adaptation operators $\alpha_{i_1} \ldots \alpha_{i_m} \in A$, such that $(\hat{p}, s', q') = \alpha_{i_1} \circ \cdots \circ \alpha_{i_m}(c)$ and there does not exist another set of adaptation operators ${\alpha'}_{i_1} \ldots {\alpha'}_{i_m} \in A$, such that $(\hat{p}, s'', q'') = {\alpha'}_{i_1} \circ \cdots \circ {\alpha'}_{i_m}(c)$ and $q'' > q'$.

A correct similarity measure (w.r.t. A) applied to the current problem $\hat{p}$ and a case c delivers a similarity value which is a function of the best quality of the solution we can obtain by adapting the case c to the current problem $\hat{p}$. Hence, this similarity measures guarantees to retrieve adaptable cases (Smyth and Keane, 1994).

Retrieval-only systems

It might be surprising that in this model, similarity measures don't encode new domain knowledge since there are many pure retrieval systems in which similarity measures do in fact encode domain knowledge. If we think about pure retrieval systems, then we can look at the similarity measure of these systems as follows: A similarity measure $sim_1(\hat{p}, c)$ $(c = (p, s, q))$ is a measure of how good the solution s contained in c is for $\hat{p}$. Consequently, sim_1 is a function of the quality and hence $sim_1(\hat{p}, (p, s, q)) = f(Q(\hat{p}, s))$ for an arbitrary bijective monotonously function f. In our model, the same information can be encoded into an infinite number of adaptation operators as follows: $\alpha_{\hat{p}}((p, s, q) = (\hat{p}, s, Q(\hat{p}, s))$. These operators state that any case can be used to solve any problem without solution modification. However, the quality is changed. If we take these adaptation operators, then the similarity measure sim_1, which is usually implemented in a CBR system, is a correct similarity measure w.r.t. this infinite set of adaptation operators.

From these considerations we can see, that we can move all the domain knowledge that is usually captured in a similarity measure of pure case retrieval systems into adaptation operators of our more general knowledge. This is another instance of the observation by Richter (Richter, 1995) that in principle, every container of a CBR system can hold any knowledge. Here, we can see that we can move the knowlege from the similarity container to the adaptation container and vice versa.

6 Conclusions

We have presented a new view of looking at transformational adaptation which has a clearly defined semantics given by the quality function Q. We defined soundness of cases and adaptation knowledge w.r.t. Q and we showed that applying sound adaptation knowledge to a sound case leads to a sound solution. We also defined some ideal properties for a CBR system like completeness and correctness.

We were able to show, that in this view, the similarity measure does not encode domain knowledge but control knowledge. That means that if we have determined the adaptation knowledge, the ideal similarity measure is already implicitly specified. From that we can conclude that we should start thinking about the adaptation knowledge and only afterwards start to determine the similarity measure particularly suited for the present adaptation knowledge. This intuition was already present in the work by Smyth and Keane (Smyth and Keane, 1994), but we now have a formal justification for it.

From our considerations, case retrieval and adaptation can be viewed as a special kind of optimization problem, i.e., finding an optimal solution to a problem. Unlike classical optimization problems, the function to be optimized is only partially known. We only have knowledge about certain points of the function as well as knowledge about particular ways of jumping from a known point to a new point. We might now consider doing case retrieval and adaptation by completely searching the whole space of known points, i.e., starting from a case and applying arbitrary sequences of operators. Due to complexity reasons, this is of course not advisable.

The next step in this line of research should concentrate on examining special interesting cases of this general model. Special cases emerge when we consider

- particular ways of representing problems and solutions (e.g. as flat feature vectors with symbolic, numeric, or mixed kind of attributes),
- restrict the set of quality values $\mathcal{R}$ to a finite number,

- consider adaptation operators of a particular type, e.g. one operator only modifies only one problem attribute.

We expect that in some special cases we can escape the computational complexity of the general case.

Acknowledgements

Thanks to Ivo Vollrath and Juergen Schumacher for their valuable comments. Funding for this work has been provided by the Commission of the European Union (INRECA-II: Information and Knowledge Reengineering for Reasoning from Cases; Esprit contract no. 22196) to which the authors are greatly indebted. The partners of INRECA-II are AcknoSoft (prime contractor, France), Daimler Benz (Germany), TECINNO (Germany), Irish Multimedia Systems (Ireland), and the University of Kaiserslautern (Germany).

References

Bylander, T. (1991). Complexity results for planning. In Mylopoulos, J. and Reiter, R., editors, *Proceedings of the 12th International Conference on Artificial Intelligence IJCAI-91*, pages 274–279.

Hanney, K., Keane, M. T., Smyth, B., and Cunningham, P. (1995). Systems, tasks and adaptation knowledge: revealing some revealing dependencies. Lecture Notes in Artificial Intelligence, 1010, pages 461–470. Springer Verlag.

Kolodner, J. L. (1993). *Case-Based Reasoning*. Morgan Kaufmann, San Mateo.

Richter, M. M. (1995). The knowledge contained in similarity measures. Invited Talk on the ICCBR-95.
http://wwwagr.informatik.uni-kl.de/~lsa/CBR/Richtericcbr95remarks.html.

Smyth, B. and Keane, M. T. (1994). Retrieving adaptable cases. In Wess, S., Althoff, K.-D., and Richter, M. M., editors, *Topics in Case-Based Reasoning: First European Workshop, EWCBR-93, selected papers*, volume 837 of *Lecture Notes in Artificial Intelligence*, pages 209–220. Springer, Berlin.

Voß, A. (1996). Towards a methodology for case adaptation. In Wahlster, W., editor, *ECAI'96, 12th European Conference on Artificial Intelligence, Aug. 1996, Budapest*, pages 147–151. John Wiley and Sons, Chichester.

Wess, S. (1995). *Fallbasiertes Problemlösen in wissensbasierten Systemen zur Entscheidungsunterstützung und Diagnostik*. PhD thesis, Universität Kaiserslautern. Available as DISKI 126, infix Verlag.