Conceptual Analogy: Conceptual clustering for informed and efficient analogical reasoning *

Katy Börner and Heiko Wode
University of Freiburg
Centre for Cognitive Science
79098 Freiburg, FRG
katy | heiko@cognition.iig.uni-freiburg.de

Roland Faßauer
HTWK Leipzig
Karl-Liebknecht Straße 132
04251 Leipzig, FRG
roland@informatik.th-leipzig.de

Abstract

Conceptual analogy (CA) is a general approach that applies conceptual clustering and concept representations to facilitate the efficient use of past experiences (cases) during analogical reasoning (Börner 1995). The approach was developed and implemented in SYN* (see also (Börner 1994, Börner and Faßauer 1995)) to support the design of supply nets in building engineering.

This paper sketches the task; it outlines the nearest-neighbor-based agglomerative conceptual clustering applied in organizing large amounts of structured cases into case classes; it provides the concept representation used to characterize case classes and shows the analogous solution of new problems based on the concepts available. However, the main purpose of this paper is to evaluate CA in terms of its reasoning efficiency; its capability to derive solutions that go beyond the cases in the case base but still preserve the quality of cases.

1 Introduction

The application of CBR methods (Kolodner 1993, Wess, Althoff and Richter 1994) to support design tasks causes several problems. First of all, one needs to deal with large amounts of highly structured cases increasing the computational expense to retrieve, match, and adapt cases. But, short response times are crucial for the acceptance and practical usage of support systems. Secondly, knowledge about the similarity and adaptability of design cases is not available in general[1]. Without this knowledge only queries that resemble one of the cases stored in the case base could be solved. However, design problems and solutions are hardly ever identical.

Conceptual Analogy was developed to handle these problems. It permits the organization of large case bases into mutually exclusive case classes that are represented conceptually by the common features (to be used as an index to the case base) and distinctive features (hinting at possibilities for adaptation) of their instances. These concepts form the basis for efficient analogical reasoning, which leads to solutions conforming to the cases seen, i.e., preserving the quality of cases in the case base.

*This research was supported by the Federal Ministry of Education, Science, Research and Technology (BMFT) within the joint project FABEL under contract no. 413-4001-01IW104. Project partners in FABEL are German National Research Center of Computer Science (GMD), Sankt Augustin, BSR Consulting GmbH, München, Technical University of Dresden, HTWK Leipzig, University of Freiburg, and University of Karlsruhe.

[1] This may be due to the fact that the acquisition and modeling of these kinds of knowledge would require more effort than the actual system support could save.

2 The Design Task

In architectural design, past experiences (cases) correspond to arrangements of physical objects represented by CAD layouts and refer to parts in real buildings. Concentrating on the design of complex installation infrastructures for industrial buildings, cases correspond to pipe systems that connect a given set of outlets to the main access. Pipe systems for fresh and return air, electrical circuits, computer networks, phone cables, etc. are numerous and show varied topological structures.

Representing the main access by a square, outlets by circles, interconnecting points by circles of smaller size and pipes by line segments, Fig. 1 (left) illustrates five pipe systems. Each of them shows a tree like structure. The main access corresponds to the root (type R), outlets correspond to leaves (type L) and all other objects are represented by internal objects (type I). Pipes correspond to edges and equal the *connected_to* relation among objects. Fig. 1 (right) shows a problem (partial cases) that corresponds to a set of accesses and its solution.

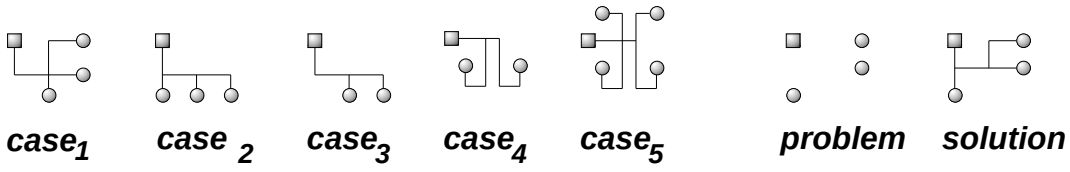

Figure 1: A case base, a problem and its solution

Cases, problems and solutions can be represented by a set of objects O and a set E of pairs of objects (or edges) so that the binary relation E holds for these pairs, i.e., $E := \{(o_i, o_j) \in O \times O \mid Eo_io_j\}$. In general, there is no operational information available about how to transform a problem into its solution.

A *case* is denoted by $c = (O^c, E^c)$. A case base CB is a finite set of cases. The set of all objects and edges represented by a $CB = \{c_1, c_2.., c_n\}$ is $O_{(CB)} = \bigcup_{c \in CB} O^c$ and $E_{(CB)} = \bigcup_{c \in CB} E^c$. A design *problem* $p = (O^p, E^p)$ is represented by the root object (main access) and by leaf objects (outlets). A *solution* to a given problem p and a CB is a case $c = (O^c, E^c)$ that contains the problem objects, eventually adds intermediate objects and provides all edges that are required to connect the root object with all leaf objects. Let C be the minimal set of cases that provide the objects and edges to solve the problem. Then $O^p \subseteq O^c \subseteq O \ \wedge \ E^p \subset E^c$ and $O^c \backslash O^p \subseteq O_{(C)} \ \wedge \ E^c \backslash E^p \subseteq E_{(C)}$ hold. In general, there can be several solutions $S_{CB,p} = \{c_i \mid c_i$ is solution of p with $CB\}$.

The *optimal* solution shares the largest interconnected structure with the cases in C and the edges added, $(E^s \backslash E^p)$, occurred most frequently in the cases of C, i.e., in $E_{(C)}$.

More formally, let *Size* characterize the number of edges of a case $Size(c) := |E|$. The prototype $PT(C) = (O^{pt}, E^{pt})$ of a set of cases C corresponds to the objects and edges that are common to all cases in C and are connected to the root object o_R. The *relative frequency* $H(E)$ of an edge (o_i, o_j) in $E_{(C)}$ corresponds to the number of case solutions containing this edge divided by the number of cases in C:

$$H_C((o_i, o_j)) := \frac{|\{c \in C \mid (o_i, o_j) \in E^c\}|}{|C|}.$$

The *average relative frequency* of a set of edges E in $E_{(C)}$ is

$$\bar{H}_C(E) := \frac{1}{|E|} \sum_{(o_i,o_j)\in E} H_C((o_i,o_j)).$$

Let $S_{CB,p} = \{c_1, c_2, ...\}$ be the set of solutions. Solutions that share a larger prototype with the cases in C are preferred ($\succeq_W$):

$$c_1 \succeq_W c_2 \Leftrightarrow Size(PT(C \cup c_1)) \geq Size(PT(C \cup c_2)).$$

Out of these, solutions with a higher average relative frequency of their new edges are preferred ($\succeq_H$):

$$c_1 \succeq_H c_2 \Leftrightarrow \bar{H}_C(E^{c_1} \backslash E^p) \geq \bar{H}_C(E^{c_2} \backslash E^p).$$

Now the design task may be characterized as follows. The *input* corresponds to (1) a finite set of *complete* cases CB; (2) *optimality criteria* that a solution has to meet; and (3) a *problem* (O^p, E^p).

Required *output* is an *optimal* solution (O^s, R^s). That is, the new solution contains the intermediate objects and edges that connect all leaf objects in O^p to the root object in O^p, offer the largest interconnected structure with the cases in C and the edges added occurred most frequently in C.

3 Conceptual Analogy

The approach of Conceptual Analogy (CA) was developed to solve the task outlined above in an efficient manner. The approach integrates *knowledge organization* based on past cases and *analogical reasoning* via *concept representations* of classes of cases sharing similar structure.

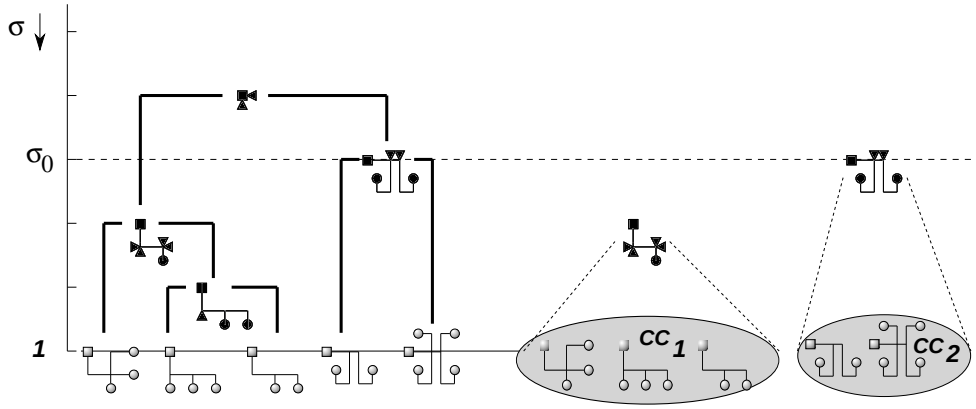

Figure 2: Conceptual clustering of case bases and the selection of one partition

Knowledge organization: CA uses unsupervised, nearest-neighbor-based, agglomerative, conceptual clustering to partition the case base into case classes. For the determination of

the nearest neighbor, tree-like case structures of different size need to be compared. The structural similarity σ of a set C of cases is defined as the quotient of the size of the prototype PT(C) and the average size of the cases in C to be compared:

$$\sigma(C) := \frac{Size(PT(C))}{\frac{1}{|C|}\sum_{c \in C} Size(c)} \in [0,1].$$

Given a case base, agglomerative (bottom-up) conceptual clustering is applied to create a case class hierarchy with partitions at multiple levels of generalization. It starts with a set of case classes, each containing a single case. Repeatedly the two most similar case classes (clusters) over the entire set are merged to form a new case class that covers both. This process continues until a single all-inclusive cluster remains. Result is an uniform, binary hierarchy which represents concrete cases by leaves. Its internal nodes represent concepts describing the cases below them by their common features (prototype) and their variations (instantiations). Denoting variables in prototypes by black triangles, Fig. 2 (left) depicts the hierarchy for the five cases represented in Fig. 1.

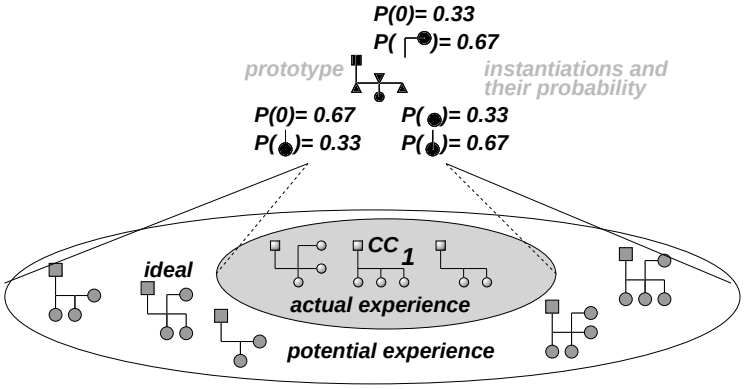

Figure 3: A concept representation as well as its actual and potential experience

A similarity threshold σ_0 is used to determine an *exhaustive partition of* CB (see Fig. 2 (right)) at which reasoning will be based. The similarity threshold value is selected using several criteria: minimal retrieval time, maximal potential knowledge and maximal conformity of proposed solutions. These criteria influence the performance of CA and are discussed in section 4. The resulting partition CCP of mutually exclusive case classes CC that exceed a similarity threshold σ_0 is defined by:

$$CCP = \{CC_i \mid \bigcup_i CC_i = CB \ \wedge \ \forall i \neq j (CC_i \cap CC_j = \emptyset) \wedge \forall i (\sigma(CC_i) \geq \sigma_0)\}.$$

Representation of case classes by concepts: Each case class is represented by its *prototype* $PT(CC)$, i.e., the most specific common structure of the cases in CC whereas differences are denoted by variables, the set of *substitutions* $IR(CC)$ that, when applied to the prototype, result in the concrete cases of CC as well as the *probability* $P(CC)$ for each substitution:

$$K(CC) = (PT(CC), IR(CC), P(CC)).$$

Note that prototypes as well as instantiation values are trees. The prototype contains the root object. Its leafs are either leaf objects or variables. Instantiations contain no variables.

The *actual experience* of a case class equals the set of concrete cases CC. The *potential experience* CC^{pot} of a case class refers to the set of cases that could be derived by all possible combinations of instantiations that replace each variable in the prototype by a concrete value. The *ideal* of a CC corresponds to the instantiation of the prototype with the maximum observed probability.

Denoting new combinations of instantiations by grey circles and squares Fig. 3 illustrates the concept representation as well as the potential experience and the ideal of CC_1 from Fig. 2. During subsequent reasoning only concepts are used. The concrete cases are stored to enable the dynamic update of concepts.

The most *efficient organization and representation* of a case base corresponds to few concepts that exactly cover the cases in CB, resulting in short answer times but allowing the derivation of past cases (solutions) only. Selecting the concept level with prototypes containing a maximum number of variables as well as a maximum number of possible instantiation values maximizes the number of potential solutions that are originally not covered by CB. The prototypes, however, may not represent enough structure to enable a meaningful similarity assessment. Aiming at *high solution conformity*, prototypes should be very specific.

Analogical Reasoning: Given a query (new problem), it is categorized into the most applicable concept. In order to do so, the problem needs to be structurally reformulated[2] in terms of the prototypes of CCP. Basically, the objects and edges of the prototype that connect leaf objects (outlets) to the problem root (main access) are transferred. The problem may be reformulated several times, applying different prototypes (see Fig. 4). The remaining unconnected problem objects are denoted by $r = O^r$.

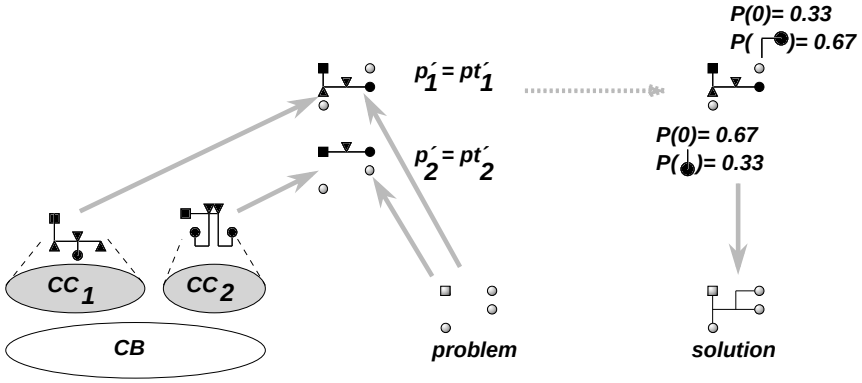

Figure 4: Categorization of a new problem and instantiation of the most applicable prototype

Retrieval for adaptation (here categorization for instantiation) requires the definition of similarity in terms of the adaptation knowledge available. The applicability α of a concept for a reformulated problem p' takes into account the unconnected problem objects O^r that are in the instantiation values of the concept thus hinting at the possibilities for adaptation[3] as well as the similarity of the *prototype* of the concept to the reformulated problem. If O^r is in the set

[2] The reformulation of the problem in terms of prior prototypes was influenced by the work of (O'Hara and Indurkhya 1994) on geometrical, proportional analogy problems.

[3] This is in the spirit of *adaptation guided retrieval* already proposed by (Paulokat, Präger and Weß 1992, Smyth and Keane 1994).

of objects of $IR(CC)$ then the applicability of $K(CC)$ is

$$\alpha(K(CC), p') = \sigma(\{PT(CC), p'\})$$

otherwise $\alpha = 0$. Given the most applicable concept, the prototype pt' of problem and $PT(CC)$ is instantiated (see Fig. 4 (right)). The range of instantiations is restricted by the set of instantiation values for each variable in the prototype. The adaptation process is controlled (and deterministic) in that the most important variable is selected and instantiated by the values with the highest probability. The scope of instantiation is a solution that relates all unconnected problem objects to the prototype.

Finally, the conformity μ of the resulting solution c is determined. The similarity of *prototype* and solution c are then considered:

$$\mu(K(CC), c) = \sigma(\{PT(\{PT(CC), p'\}), c\}).$$

The solution is presented to the user along with its *conformity*. The prototype applied can be used to explain the solution.

Update Given the new solution, the user is now in the position to accept or modify it or, if the solution is incomplete, to complete the solution or to reject it. Accepted solutions are stored in the case class applied, changing at least the probability of instantiations. Given that the solution had to be modified, completed or generated by the user its addition to the case base may lead to a complete re-organization and a conceptual re-representation of the entire case base. In such a way, the organization of the case base into case classes and the representation of case classes by concepts is dynamically changed to incorporate new cases.

4 Evaluation

As for the evaluation of *Conceptual Analogy* to standard CBR approaches we consider the *memory space*, the *reasoning efficiency*, the *reasoning performance*, and the *quality* of proposed solutions.

Memory space: The memory space, required by CBR approaches to store all cases in a flat CB, corresponds to the size of the cases. CA stores the cases itself, plus the concept representations of CC, i.e., their prototypes, instantiations and corresponding weights and requires approximately twice as much space.

Reasoning efficiency: Given a time limit[4], the reasoning efficiency is important for the success of a certain approach. The number EM of edges to compare, required by conventional CBR (for the serial search through all cases) corresponds to the size of cases in the CB:

$$EM_{CBR} = \sum_{c \in CB} Size(c).$$

As for CA, the selection of the most applicable concept of a certain CCP plus the serial instantiation of variables corresponds approximately to the size of prototypes of the CC in CCP plus the average size of instantiation values per each CC in CCP divided by the number of CC:

$$EM_{CA} := \sum_{CC \in CCP} Size(PT(CC)) + \frac{1}{\mid CCB \mid} \times \sum_{CC \in CCP} \sum_{x \in VAR(PT(CC))} Size(IR(CC, x)).$$

[4]That resembles the time the user is willing to wait for a solution before he returns to his do-it-yourself habit.

whereas $VAR(PT(CC))$ denotes the set of variables in $PT(CC)$ and $IR(CC, x)$ refers to the set of instantiation values for variable $x \in VAR(PT(CC))$. This allows CA reasoning to function as efficiently or even more efficiently than CBR:

$$EM_{CBR} \geq EM_{CA}.$$

Reasoning performance: Another distinctive advantage of CA is the derivation of potential experiences that conform to the cases seen, but which go beyond the set of actual experiences stored in the CB. While CBR methods have the cases in CB at their disposal exclusively, the set of potential experiences derivable from the conceptual representations of case classes exceeds or equals the set of cases in CB:

$$CB \subseteq \bigcup_{CC \in CCP} CC^{pot}.$$

Solution quality: A basic intention of case-based reasoning is the preservation of the case *quality* during memory organization as well as reasoning. Proposed solutions should conform to the cases in CB. The conformity of cases to themselves equals one. Thus, the average conformity of all cases that are *retrieved* unchanged by CBR equals one. The average conformity of the cases that are *derivable* by CA from the conceptual representations of CCP corresponds to the conformity of the potential solutions that are derivable from all concepts representing the CC's of a CB divided by the number of all potential solutions:

$$\overline{\mu}_{CA} := \frac{1}{|CCP|} \sum_{CC \in CCP} \frac{1}{|CC^{pot}|} \sum_{c \in CC^{pot}} \mu(K(CC), c).$$

Obviously, the conformity of the cases in CB to themselves (CBR) is equal to or higher than the average conformity of potential experiences derived by CA:

$$\overline{\mu}_{CBR} = 1 \geq \overline{\mu}_{CA}.$$

Optimal partition: The optimal partition of CB (and thus the optimal similarity threshold value σ_0) corresponds to a compromise between maximum reasoning efficiency (expressed by the minimum number of *comparisons* required for classification and instantiation), maximum number of *potential experiences* and maximum *conformity* of the potential set of solutions. Norming all three features to the values obtained for standard CBR the partition that minimizes

$$\frac{EM_{CA}}{EM_{CBR}} \times \frac{|CB|}{|\bigcup_{CC \in CCP} CC^{pot}|} \times \frac{1}{\overline{\mu}_{CA}}$$

is selected for reasoning. Depending on the application requirements or user preferences the three factors may be weighted differently.

5 Related Work and Discussion

Conceptual clustering techniques have been used in several systems to organize and index cases in an efficient way. Already UNIMEM (Lebowitz 1987) applied inductive learning methods to identify predictive features to be used as indices. Hierarchical structured case memories that group cases sharing similar attributes into classes and represent each case class by a concept

have been proposed. These concepts have been either described by *sets of common features* (MOP's) like in CYRUS (Kolodner 1983), or as *probabilistic concepts* as in COBWEB (Fisher 1987). To our knowledge, the concept representation used by CA is unique in that it divides the knowledge contained in the case class into a set of common features to be used as indices and a set of generic knowledge about possible adaptations for design.

Whereby most systems are restricted to attribute-value languages, LABIRINTH (Thompson and Langley 1991) extends COBWEB into structured domains. LABIRINTH learns unclassified objects that have relations and uses the component structure to constrain matching. It demonstrates a method for learning from hierarchically decomposed objects, using the results of component classification to guide object classification. However, like COBWEB, LABIRINTH is restricted to classification tasks.

BRIDGER (Reich 1991) was designed to support the design of cable-stayed bridges. It organizes attribute-value descriptions of bridges into a hierarchical classification tree. In synthesis, the set of specification properties, considered to be a partial description of a new bridge, is classified with the hierarchy. Once the description reaches a leaf node, the properties missing from its description are completed from that leaf as well as from several adjacent nodes. Each completion results in a solution candidate. There is no mechanism for selecting the optimal solution nor to handle interrelated attributes.

COBWEB (Fisher 1987) is interesting in its use of a principled evaluation function that favors clusters maximizing the potential for inferring information. In contrast to earlier unsupervised learners, which have been evaluated in terms of the *comprehensibility* of the formed concepts, COBWEB was explicitly evaluated in performance tasks - *missing attribute prediction*. CA is optimized and evaluated in terms of *reasoning efficiency*, maximum *potential experience vs. actual experience*, and maximum average *conformity* of proposed solutions.

There are also a number of CBR approaches that apply conceptual clustering techniques in organizing their case base. The *Prototype-Based Indexing System* (PBIS) proposed by (Malek and Amy 1994) uses an incremental prototype-based neural network to organize cases into groups of similar cases and to represent each group of cases by prototypes. The JANUS CBR Shell (Schiemann and Woltering 1995) applies a Cohonen network to automatically organize cases into disjoint case classes corresponding to similar attribute values and represents these classes by reference cases. Both systems do a two stage retrieval. Firstly, the prototype/reference case pointing to a case class is selected. Secondly, this case class is searched for the most similar concrete case. Its solution is presented as the actual classification. The systems are restricted to attribute-value representations of cases and to classification tasks.

CA's contributions are: The *efficient, structural* organization of cases into case classes. Its use of *performance criteria* to select an optimal partition of CB. The representation of CC's by concepts representing knowledge about the *similarity* (prototype) and *adaptability* (instantiations) of cases. During reasoning concepts allow for the *efficient* categorization of a new problem into the most *applicable* (instead of most similar) case class. Concepts allow also for the generation of *potential solutions* that go beyond past cases and the evaluation of new solutions for their *conformity*. Prototypes itself may be used to explain design solutions. Last but not least, CA requires an *extremely low knowledge acquisition and modeling effort* that is important for the acceptance of design support systems.

6 Acknowledgements

We would like to thank David W. Aha, Eberhard Pippig, and M. M. Richter for several discussions on this topic that helped to shape this research. Nonetheless, the paper reflects our personal view.

References

Börner, K. (1994). Structural similarity as guidance in case-based design, *in* Wess et al. (1994), pp. 197–208.

Börner, K. (1995). Conceptual analogy, *in* D. W. Aha and A. Ram (eds), *AAAI 1995 Fall Symposium Series: Adaptation of Knowledge for Reuse*, Boston, MA, pp. 5–11.

Börner, K. and Faßauer, R. (1995). Analogical Layout Design (Syn*), *in* K. Börner (ed.), *Modules for Design Support*, FABEL Report no. 35, GMD, Sankt Augustin.

Fisher, D. H. (1987). Knowledge acquisition via incremental conceptual clustering, *Machine Learning* **2**: 139–172.

Kolodner, J. L. (1983). Reconstructive memory: A computer model, *Cognitive Science* **7**: 281–328.

Kolodner, J. L. (1993). *Case-based reasoning*, Morgan Kaufmann Publishers, San Mateo, CA.

Lebowitz, M. (1987). Experiments with incremental concept formation: UNIMEM, *Machine Learning* **2**: 103–138.

Malek, M. and Amy, B. (1994). A pre-processing model for integrating CBR and prototype-based neural networks, *Technical report*, Working Paper TIMC-LIFIA-IMAG Grenoble.

O'Hara, S. E. and Indurkhya, B. (1994). Incorporating (re)-interpretation in case-based reasoning, *in* Wess et al. (1994), pp. 246–260.

Paulokat, J., Präger, R. and Weß, S. (1992). CAbPlan – fallbasierte Arbeitsplanung, *in* T. Messer and A. Winkelhofer (eds), *6. Workshop Planen und Konfigurieren*, pp. 169–176.

Reich, Y. (1991). *Building and improving design systems*, EDRC 02-16-91, Carnegie Mellon University.

Schiemann, I. and Woltering, A. (1995). Organisation großer Fallbasen in der TUB-JANUS Shell zum effizienten Retrieval geeigneter Fälle, *CBR-WS, XPS-95*, pp. 30–36.

Smyth, B. and Keane, M. T. (1994). Retrieving adaptable cases: The role of adaptation knowledge in case retrieval, *in* Wess et al. (1994), pp. 209–220.

Thompson, K. and Langley, P. (1991). Concept formation in structured domains, *in* D. Fisher, M. Pazzani and P. Langley (eds), *Concept formation: Knowledge and experience in unsupervised learning*, Morgan Kaufmann Publishers, chapter 5.

Wess, S., Althoff, K.-D. and Richter, M. M. (eds) (1994). *Topics in Case-Based Reasoning – Selected Papers from the First European Workshop on Case-Based Reasoning (EWCBR-93)*, Vol. 837 of *LNAI*, Springer Verlag.