

Case-Based Reasoning

- An Introduction

Agnar Aamodt

University of Trondheim, Department of Informatics
N-7055 Dragvoll, Norway.

E-mail: agnar.aamodt@ifi.unit.no, Fax: +47 73 591733, Phone: +47 73 591838

1. Introduction

Case-based reasoning is a recent approach to knowledge-based problem solving and decision support: A new problem is solved by remembering a previous similar situation and by reusing information and knowledge of that situation. Let us illustrate this by looking at some typical problem solving situations:

A physician is examining a patient in his office. He gets a reminding to a patient that he treated two weeks ago. Assuming that the reminding was caused by a similarity of important symptoms, the physician uses the diagnosis and treatment of the previous patient to determine the disease and treatment for the patient in front of him.

A financial consultant working on a difficult credit decision task uses a reminding to a previous case, which involved a company in similar trouble as the current one, to recommend that the loan application should be refused.

A drilling engineer has experienced several dramatic blow out situations. He is quickly reminded of one of these situations when the combination of critical measurements during drilling matches those of a blow out case. In particular, he may get a reminding to a mistake he made during a previous blow-out, and use this to avoid repeating the error once again.

Reasoning by reusing past cases is a powerful and frequently applied way to solve problems for humans. This claim is supported by results from cognitive psychological research, and a part of the foundation for the case-based approach is its psychological plausibility [Ross-89]. Case-based reasoning is a problem solving paradigm that in many respects is fundamentally different from other major AI approaches. Instead of relying solely on general knowledge of a problem domain, or making associations along generalized relationships between problem descriptors and conclusions, CBR is able to utilize the *specific* knowledge of previously experienced, concrete problem situations (cases). A new problem is solved by finding a similar past case, and reusing it in the new problem situation. A second important difference is that CBR also is an approach to incremental, sustained learning, since a new experience is retained each time a problem has been solved, making it immediately available for future problems. Case-based reasoning can be considered a form of analogical reasoning, where the analogs typically are within the same application domain. However, as I will get back to later, the main body of research on analogical reasoning has a different focus, namely analogies across domains.

In CBR terminology, a *case* usually denotes a *problem situation* including its interpretation, solution, and possible annotations. A case is previously experienced situation, which has been captured and learned in such way that it can be reused in the solving of future problems.

CBR is a combined approach to problem solving and machine learning, and a strong driving force behind case-based methods has come from the machine learning community. This makes CBR methods particularly interesting from a decision support point of view, since learning abilities in such systems is something that is urgently needed, but so far missing. Learning in CBR occurs as a natural by-product of problem solving. When a problem is successfully solved, the experience is retained in order to solve similar problems in the future. When an attempt to solve a problem fails, the reason for the failure is identified and remembered in order to avoid the same mistake in the future.

It seems clear that human problem solving and learning in general are processes that involve the representation and utilization of several types of knowledge, and the combination of several reasoning methods. If cognitive plausibility is a guiding principle, an architecture for knowledge-based systems where the reuse of cases is at the centre, should also incorporate other and more general types of knowledge in one form or another. This is an issue of current concern in CBR research.

2. Historical points

The first system that might be called a case-based reasoner was the CYRUS system, developed at Yale University [Kolodner-83]. CYRUS was based on the work by Roger Schank on dynamic memory [Schank-82], and was basically a question-answering system (with knowledge of the various travels and meetings of former US Secretary of State Cyrus Vance). The case memory model developed for this system has later served as basis for several well-known case-based reasoning systems (e.g. MEDIATOR [Simpson-85], PERSUADER [Sycara-88], CHEF [Hammond-89], JULIA [Hinrichs-92], CASEY [Koton-89]). Another basis for CBR, and another set of models, were developed at the University of Texas, Austin [Porter-86]. This work initially addressed the machine learning problem of concept learning for classification tasks. This led to the development of the PROTOS system [Bareiss-89], which emphasized integrating general domain knowledge and specific case knowledge into a unified representation structure. Another significant contribution to the CBR field in the early days was the work done at the University of Massachusetts, Amherst. With several law scientists in the group, they were interested in the role of precedence reasoning in legal judgements [Rissland-83]. Cases (precedents) are here not used to produce a single answer, but to interpret a situation in court, and to produce and assess arguments for both parties. At MIT they studied the use of case-based reasoning to optimize performance in an existing knowledge based system, where the domain (heart failure diagnosis) was described by a deep, causal model. This resulted in the CASEY system [Koton-89], in which case-based and deep model-based reasoning was combined.

In Europe, research on CBR was taken up a little later than in the US. The CBR work seems to have been stronger coupled to expert systems development and knowledge acquisition research than in the US. Among the earliest results was the work on CBR for complex technical diagnosis done at the University of Kaiserslautern [Althoff-89]. This led to the PATDEX system [Richter-91]. At IIIA in Blanes a case-based learning apprentice system for medical diagnosis [Plaza-90], and the use of case-based methods for strategy-level reasoning [López-90] was investigated. At King's College, Aberdeen, they studied the use of cases for knowledge base refinement, and an early result was the REFINER system [Sharma-88]. At the University of Trondheim the topic of study was the use of CBR in the context of knowledge acquisition and maintenance, where the combined use of cases and general domain knowledge was focused [Aamodt-89]. This led to the development of the CREEK system and integration framework [Aamodt-91]. On the cognitive science side, early work was done on analogical reasoning at Trinity College, Dublin, [Keane-88], and at the University of Freiburg, where the role of episodic knowledge in cognitive models was investigated [Strube-90].

Currently, the CBR activities in the United States as well as in Europe are spreading out (e.g. [CBR-91,EWCBR-93,IEEE-92]). In Japan and other Asian countries, for example in India [Venkatamaran-93], there are also activity points. In Japan the interest is to a large extent focused on the parallel computation approach to CBR [Kitano-93]. A particular emphasis in European CBR seems to be the integration of case-specific and general knowledge, i.e. viewing the CBR technology as an extension and strengthening of previous knowledge-based approaches - both by providing reuse of case-specific knowledge and by enabling sustained learning through experience. This perspective is currently explored within the Esprit-III project INRECA [Manago-93], where cases are the basis for integrating a CBR reasoner and learner with an inductive learning method and a decision tree reasoner. The project will demonstrate its result on practical real-world applications.

3. A characterisation of CBR methods

Central tasks that all case-based reasoning methods have to deal with are to identify the current problem situation, find a past case similar to the new one, use that case to suggest a solution to the current problem, evaluate the proposed solution, and update the system by learning from this experience. How this is done, what part of the process is focused, what type of problems drives the methods, etc. varies considerably, however.

The CBR paradigm covers a range of different methods for organizing, retrieving, utilizing and indexing the knowledge retained in past cases. Cases may be kept as concrete experiences, or a set of similar cases may form a generalized case. Cases may be stored as separate knowledge units, or splitted up into subunits and distributed within the knowledge structure. Cases may be indexed by a prefixed or open vocabulary, and within a flat or hierarchical index structure. The solution from a previous case may be directly applied to the present problem, or modified according to differences between the two cases. The matching of cases, adaptation of solutions, and learning from an experience

may be guided and supported by a deep model of general domain knowledge, by more shallow and compiled knowledge, or be based on an apparent, syntactic similarity only. CBR methods may be purely self-contained and automatic, or they may interact heavily with the user for support and guidance of its choices. Some CBR method assume a rather large amount of widely distributed cases in its case base, while others are based on a more limited set of typical ones. Past cases may be retrieved and evaluated sequentially or in parallel. Actually, "case-based reasoning" is just one of a set of terms used to refer to systems of this kind. This has lead to some confusions, particularly since case-based reasoning is a term used both as a generic term for several types of more specific approaches, as well as for one such approach. To some extent, this can also be said for analogy reasoning. An attempt to clarify these notions is given in the following, by describing the main types of CBR approaches.

Instance-based reasoning. This characterises a highly syntactic CBR-approach. To compensate for lack of guidance from general background knowledge, a relatively large number of instances are needed in order to close in on a concept definition. The representation of the instances are usually simple (e.g. feature vectors), since the major focus is on studying automated learning with no user in the loop. Instance-based reasoning labels work by Kibler and Aha and colleagues [Aha-91]. Basically, it is a non-generalization approach to the concept learning problem addressed by classical, inductive machine learning methods.

Memory-based reasoning. This approach emphasizes a collection of cases as a large memory, and reasoning as a process of accessing and searching in this memory. Memory organization and access is a focus of these methods. The utilization of parallel processing techniques is a characteristic of these methods, and distinguishes this approach from the others. The access and storage methods may rely on purely syntactic criteria, as in the MBR-Talk system [Stanfill-88], or they may attempt to utilize general domain knowledge, as ongoing work in Japan on massive parallel memories [Kitano-93].

Case-based reasoning (in the typical sense). Although case-based reasoning is used as a generic term in this paper, the typical case-based reasoning methods [Kolodner-93] have some characteristics that distinguish them from the other approaches listed here. First, a typical case is usually assumed to have a certain degree of richness of information contained in it, and a certain complexity with respect to its internal organization. That is, a feature vector holding some values and a corresponding class is not what we would call a typical case description. What is referred to as typical case-based methods also has another characteristic property: They are able to modify, or adapt, a retrieved solution when applied in a different problem solving context. Paradigmatic case-based methods also utilizes general background knowledge - although its richness, degree of explicit representation, and role within the CBR processes varies. Core methods of typical CBR systems borrow a lot from cognitive psychology theories.

Analogy-based reasoning. This term is sometimes used, as a synonym to case-based reasoning, to describe the typical case-based approach just described [Veloso-93].

However, it is also used to characterize methods that solve new problems based on past cases from a different domain, while typical case-based methods focus on indexing and matching strategies for single-domain cases. Research on analogy reasoning is therefore a subfield concerned with mechanisms for identification and utilization of cross-domain analogies [Kedar-Cabelli-88]. The major focus of these methods has been on the reuse of a past case, what is called the mapping problem: Finding a way to transfer, or map, the solution of an identified analogue (called source or base) to the present problem (called target).

Throughout the paper I will continue to use the term case-based reasoning in the generic sense, although our examples, elaborations, and discussions will lean towards CBR in the typical sense.

4. The CBR process

At the highest level of generality, a general CBR cycle may be described by the following four processes:

1. RETRIEVE the most similar case or cases
2. REUSE the information and knowledge in that case to solve the problem
3. REVISE the proposed solution
4. RETAIN the parts of this experience likely to be useful for future problem solving

A new problem is solved by *retrieving* one or more previously experienced cases, *reusing* the case in one way or another, *revising* the solution based on reusing a previous case, and *retaining* the new experience by incorporating it into the existing knowledge-base (case-base). The four processes each involve a number of more specific steps, which will be described in the task model. In Fig. 1, this cycle is illustrated. An initial description of a problem (top of Fig. 1) defines a *new case*. This new case is used to RETRIEVE a case from the collection of *previous cases*. The *retrieved case* is combined with the new case - through REUSE - into a *solved case*, i.e. a proposed solution to the initial problem. Through the REVISE process this solution is tested for success, e.g. by being applied to the real world environment or evaluated by a teacher, and repaired if failed. During RETAIN, useful experience is retained for future reuse, and the case base is updated by a new *learned case*, or by modification of some existing cases. As indicated in the figure, general knowledge usually plays a part in this cycle, by supporting the CBR processes. This support may range from very weak (or none) to very strong, depending on the type of CBR method. By general knowledge I here mean general domain-dependent knowledge, as opposed to specific knowledge embodied by cases. For example, in diagnosing a patient by retrieving and reusing the case of a previous patient, a model of anatomy together with causal relationships between pathological states may constitute the general knowledge used by a CBR system. A set of rules may have the same role.

In the following four sections, an overview of the four processes - or tasks - are described, and exemplified by some existing systems. A more extensive discussion, from which the description below to some extent has been excerpted, is given in [Aamodt&Plaza-94]. The

retrieval process is described most extensively, since here is where most of the research has been concentrated so far, and where the methods are most mature.

5. Retrieving a best matching case

The Retrieve task starts with a (partial) problem description, and ends when a best matching previous case has been found. Its subtasks are to identify relevant features, find an initial set of matches, and select the most promising one. The identification task basically comes up with a set of relevant problem descriptors, the goal of the matching task is to return a set of cases that are sufficiently similar to the new case - given a similarity threshold of some kind, and the selection task works on this set of cases and chooses the best match (or at least a first case to try out). While some case-based approaches retrieve a previous case largely based on superficial, *syntactical similarities* among problem descriptors (e.g. the CYRUS system, and PATDEX-1 systems), some approaches attempt to retrieve cases based on features that have deeper, *semantical similarities* (e.g. the PROTOS, CASEY, and CREEK systems). Ongoing work in the FABEL project, aimed to develop a decision support system for architects, explores various methods for combined reasoning and mutual support of different knowledge types [FABEL -93]. In order to match cases based on semantic similarities and relative importance of features, an extensive body of general domain knowledge is needed to produce an explanation of why two cases match and how strong the match is. Syntactic similarity assessment - sometimes referred to as a "knowledge-poor" approach - has its advantage in domains where general domain knowledge is very difficult or impossible to acquire. On the other hand, semantical oriented approaches - referred to as "knowledge-intensive" - are able to use the contextual meaning of a problem description in its matching, for domains where general domain knowledge is available.

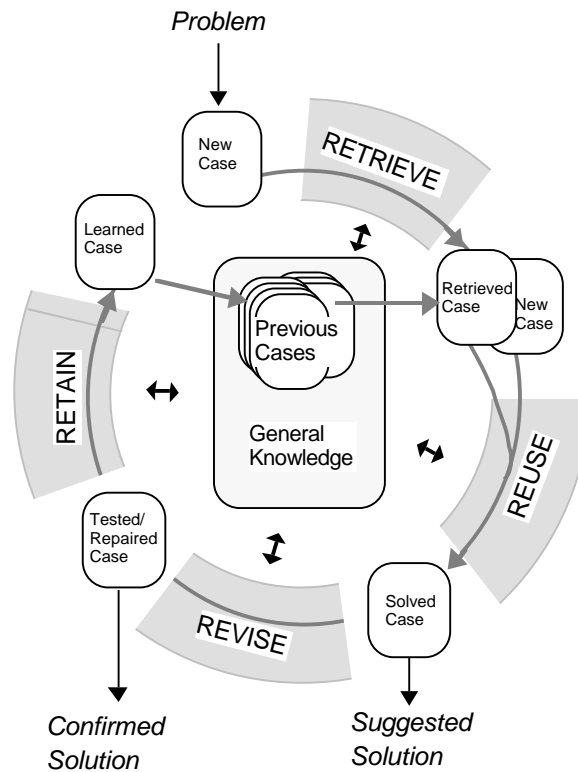


Fig. 1. The CBR Cycle

A question that should be asked when deciding on a retrieval strategy, is the purpose of the retrieval task. If the purpose is to retrieve a case which is to be adapted for reuse, this can be accounted for in the retrieval method.

Finding a set of matching cases is done by applying the problem descriptors (input features) as indexes to the case memory in a direct or indirect way. There are in principle three ways of retrieving a case or a set of cases: By following direct index pointers from problem features, by searching an index structure, or by searching in a model of general domain knowledge. PATDEX implements the first strategy for its diagnostic reasoning, and the second for test selection. A domain-dependent, but global similarity metric is used to assess similarity based on surface match. Dynamic memory based systems takes the second approach, but general domain knowledge may be used in combination with search in the discrimination network. PROTOS and CREEK combines one and three, since direct pointers are used to hypothesize a candidate set which in turn is justified as plausible matches by use of general knowledge to explain the match.

Cases may be retrieved solely from input features, but also from features inferred from the input. Cases that match all input features are, of course, good candidates for matching, but - depending on the strategy - cases that match a given fraction of the problem features (input or inferred) may also be retrieved. PATDEX uses a global similarity metric, with several parameters that are set as part of the domain analysis. Some tests for relevance of

a retrieved case is often executed, particularly if cases are retrieved on the basis of a subset of features. For example, a simple relevance test may be to check if a retrieved solution conforms with the expected solution type of the new problem. A way to assess the degree of similarity is needed, and several 'similarity metrics' have been proposed, based on surface similarities of problem and case features.

Similarity assessment may also be more knowledge-intensive, for example by trying to understand the problem more deeply, and using the goals, constraints, etc. from this elaboration process to guide the matching [Aamodt-94]. Another option is to weigh the problem descriptors according to their importance for characterizing the problem, during the learning phase. In PROTOS, for example, each feature in a stored case has assigned to it a degree of importance for the solution of the case. A similar mechanism is adopted by CREEK, which stores both the predictive strength (discriminatory value) of a feature with respect to the set of cases, as well as a feature's criticality, i.e. what influence the lack of a feature has on the case solution.

From the set of similar cases, a best match is chosen. This may have been done during the initial match process, but more often a set of cases are returned from that task. The best matching case is usually determined by evaluating the degree of initial match more closely. This is done by an attempt to generate explanations to justify non-identical features, based on the knowledge in the semantic network. If a match turns out not to be strong enough, an attempt to find a better match by following difference links to closely related cases is made. This subtask is usually a more elaborate one than the retrieval task, although the distinction between retrieval and elaborate matching is not distinct in all systems. The selection process typically generate consequences and expectations from each retrieved case, and attempts to evaluate consequences and justify expectations. This may be done by using the system's own model of general domain knowledge, or by asking the user for confirmation and additional information. The cases are eventually ranked according to some metric or ranking criteria. Knowledge-intensive selection methods typically generate explanations that support this ranking process, and the case that has the strongest explanation for being similar to the new problem is chosen. Other properties of a case that are considered in some CBR systems include relative importance and discriminatory strengths of features, prototypicality of a case within its assigned class, and difference links to related cases.

6. Reusing a past case

The reuse of the retrieved case solution in the context of the new case focuses on two aspects: (a) the differences among the past and the current case and (b) what part of retrieved case can be transferred to the new case. In simple classification tasks the differences are abstracted away (they are considered non relevant while similarities are relevant) and the solution class of the retrieved case is transferred to the new case as its solution class. This is a trivial type of reuse. However, other systems have to take into account differences in (a) and thus the reused part (b) cannot be directly transferred to the new case but requires an *adaptation* process that takes into account those differences.

There are two main ways to adapt a past case for a new problem. One is based on reusing the past case solution, the other on reusing the past method that constructed the solution. The first type of reuse does not look at how a problem is solved but focuses on the equivalence of solutions, and this usually requires a domain-dependent model in the form of transformational operators. An example is the CASEY system, where a new causal explanation is built from the old causal explanations by rules where the condition-part indexes differences and with a transformational operator at the action part of the rule.

Solution method reuse looks at *how* the problem was solved in the retrieved case. The retrieved case holds information about the method used for solving the retrieved problem including a justification of the operators used, subgoals considered, alternatives generated, failed search paths, etc. Then the retrieved method is reinstantiated and "replayed" within the new context. During the replay successful alternatives, operators, and paths will be explored first while failed paths will be avoided; new subgoals are pursued based on the old ones and old subplans can be recursively retrieved for them. An example of derivational reuse is the Analogy/Prodigy system [Veloso-93] that reuses past plans guided by commonalities of goals and initial situations, and resumes a means-ends planning regime if the retrieved plan fails or is not found.

7. Revising a solution

When a case solution generated by the reuse phase is not correct, an opportunity for learning from failure arises. This phase is called case revision and consists of two tasks: (1) evaluate the case solution generated by reuse. If successful, learn from the success (case retainment, see next section), (2) otherwise repair the case solution using domain-specific knowledge or user input.

The evaluation task takes the result from applying the solution in the real environment (asking a teacher or performing the task in the real world). This is usually a step outside the CBR system, since it - at least for a system in normal operation - involves the application of a suggested solution to the real problem. The results from applying the solution may take some time to appear, depending on the type of application. In a medical decision support system, the success or failure of a treatment may take from a few hours up to several months. The case may still be learned, and be available in the case base in the intermediate period, but it has to be marked as a non-evaluated case. A solution may also be applied to a simulation program that is able to generate a correct solution. This is used in CHEF, where a solution (i.e. a cooking recipe) is applied to an internal model assumed to be strong enough to give the necessary feedback for solution repair.

Case repair involves detecting the errors of the current solution and retrieving or generating explanations for them. The best example is the CHEF system, where causal knowledge is used to generate an explanation of why certain goals of the solution plan were not achieved. CHEF learns the general situations that will cause the failures using an explanation-based learning technique. This is included into a failure memory that is used in the reuse phase to predict possible shortcomings of plans. This form of learning moves

detection of errors in a pot hoc fashion to the elaboration plan phase were errors can be predicted, handled and avoided. A second step of the revision phase uses the failure explanations to modify the solution in such a way that failures do not occur. For instance, the failed plan in the CHEF system is modified by a repair module that adds steps to the plan that will assure that the causes of the errors will not occur. The repair module possesses general causal knowledge and domain knowledge about how to disable or compensate causes of errors in the domain. The revised plan can then be retained directly (if the revision phase assures its correctness) or it can be evaluated and repaired again.

8. Retaining a case for learning

This is the process of incorporating what is useful to retain from the new problem solving episode into the existing knowledge. The learning from success or failure of the proposed solution is triggered by the outcome of the evaluation and possible repair. It involves selecting which information from the case to retain, in what form to retain it, how to index the case for later retrieval from similar problems, and how to integrate the new case in the memory structure.

In CBR the case base is updated no matter how the problem was solved. If it was solved by use of a previous case, a new case may be built or the old case may be generalized to subsume the present case as well. If the problem was solved by other methods, including asking the user, an entirely new case will have to be constructed. In any case, a decision need to be made about what to use as the source of learning. Relevant problem descriptors and problem solutions are obvious candidates. But an explanation or another form of justification of why a solution is a solution to the problem may also be marked for inclusion in a new case. In CASEY and CREEK, for example, explanations are included in retained cases, and reused in later modification of the solution. CASEY uses the previous explanation structure to search for other states in the diagnostic model which explains the input data of the new case, and to look for causes of these states as answers to the new problem. This focuses and speeds up the explanation process, compared to a search in the entire domain model. The last type of structure that may be extracted for learning is the problem solving method, i.e. the strategic reasoning path, making the system suitable for derivational reuse.

Failures, i.e. information from the Revise task, may also be extracted and retained, either as separate failure cases or within total-problem cases. When a failure is encountered, the system can then get a reminding to a previous similar failure, and use the failure case to improve its understanding of - and correct - the present failure.

The 'indexing problem' is a central and much focused problem in case-based reasoning. It amounts to deciding what type of indexes to use for future retrieval, and how to structure the search space of indexes. Direct indexes, as previously mentioned, skips the latter step, but there is still the problem of identifying what type of indexes to use. This is actually a knowledge acquisition problem, and should be analyzed as part of the domain knowledge analysis and modeling step. A trivial solution to the problem is of course to use all input features as indices. This is the approach of syntax-based methods within instance-based

and memory-based reasoning. In the memory-based method of CBR-Talk [Stanfill-88], for example, relevant features are determined by matching, in parallel, all cases in the case-base, and filtering out features that belong to cases with few features in common with the problem case.

In CASEY, a two-step indexing method is used. Primary index features are - as referred to in the section on representation - general causal states in the heart failure model that are part of the explanation of the case. When a new problem enters, the features are propagated in the heart failure model, and the states that explain the features are used as indices to the case memory. The observed features themselves are used as secondary features only.

This is the final step of updating the knowledge base with new case knowledge. If no new case and index set has been constructed, it is the main step of Retain. By modifying the indexing of existing cases, CBR systems learn to become better similarity assessors. The tuning of existing indexes is an important part of CBR learning. Index strengths or importances for a particular case or solution are adjusted due to the success or failure of using the case to solve the input problem. For features that have been judged relevant for retrieving a successful case, the association with the case is strengthened, while it is weakened for features that lead to unsuccessful cases being retrieved. In this way, the index structure has a role of tuning and adapting the case memory to its use. PATDEX has a special way to learn feature relevance: A relevance matrix links possible features to the diagnosis for which they are relevant, and assign a weight to each such link. The weights are updated, based on feedback of success or failure, by a connectionist method.

In knowledge-intensive approaches to CBR, learning may also take place within the general conceptual knowledge model, for example by other machine learning methods (see next section) or through interaction with the user. Thus, with a proper interface to the user (whether a competent end user or an expert) a system may incrementally extend and refine its general knowledge model, as well as its memory of past cases, in the normal course of problem solving. This is an inherent method in the PROTOS system, for example. All general knowledge in PROTOS is assumed to be acquired in such a bottom-up interaction with a competent user.

The case just learned may finally be tested by re-entering the initial problem and see whether the system behaves as wanted.

9. Application types

As already indicated, the most developed part of the CBR process is the retrieval part. Current applications and development tools are therefore mainly implementing the retrieval step, although there are also examples of successful applications that include adaptation as well as the learning component. The major application type that has emerged from the CBR arena, however, is "help desk" support systems [Allen-94, Simoudis-92], where the system's job is to retrieve a case corresponding to a previous similar situation, and let the case reuse and possible adaptation be done by the human user. Such systems may also be a sound step towards a more full-fledged CBR system. Below, two applications are described, which illustrate relevant application types other than help desks, and also issues

to be aware of when developing and evaluating CBR systems.

At Lockheed, Palo Alto, the first fielded CBR system was developed. The problem domain is optimization of autoclave loading for heat treatment of composite materials [Hennessy-92]. The autoclave is a large convection oven, where airplane parts are treated in order to get the right properties. Different material types need different heating and cooling procedures, and the task is to load the autoclave for optimized throughput, i.e. to select the parts that can be treated together, and distribute them in the oven so that their required heating profiles are taken care of. There are always more parts to be cured than the autoclave can take in one load. The knowledge needed to perform this task reasonably well used to reside in the head of a just a few experienced people. There is no theory and very few generally applicable schemes for doing this job, so to build up experience in the form of previously successful and unsuccessful situations is important. The motivation for developing this application was to be able to remember the relevant earlier situations. Further, a decision support system would enable other people than the experts to do the job, and to help training new personnel. The development of the system started in 1987, and it has been in regular use since the fall 1990. The results so far are very positive. The current system handles the configuration of one loading operation in isolation, and an extended system to handle the sequencing of several loads is under testing. The development strategy of the application has been to hold a low-risk profile, and to include more advanced functionalities and solutions as experience with the system has been gained over some time.

The second application has been developed at General Dynamics, Electric Boat Division [Brown-91]. During construction of ships, a frequently re-occurring problem is the selection of the most appropriate mechanical equipment, and to fit it to its use. Most of these problems can be handled by fairly standard procedures, but some problems are harder and occur less frequently. These type of problems - referred to as "non-conformances" - also repeat over time, and because regular procedures are missing, they consume a lot of resources to get solved . General Dynamics wanted to see whether a knowledge-based decision support tool could reduce the cost of these problems. The application domain chosen was the selection and adjustment of valves for on-board pipeline systems. The development of the first system started in 1986, using a rule-based systems approach. The testing of the system on real problems initially gave positive results, but problems of brittleness and knowledge maintenance soon became apparent. In 1988 a feasibility study was made of the use of case-based reasoning methods instead of rules, and a prototype CBR system was developed. The tests gave optimistic results, and an operational system was fielded in 1990. The rule-base was taken advantage of in structuring the case knowledge and filling the initial case base. IN the fall of 1991 the system was continually used in three out of four departments involved with mechanical construction. A quantitative estimate of cost reductions has been made: The rule-based system took 5 man-years to develop, and the same for the CBR system (2 man-years of studies and experimental development and 3 man-years for the prototype and operational system). This amounts to \$750.000 in total costs. In the period December 90 - September 91 20.000 non-

conformances were handled. The cost reduction, compared to previous costs of manual procedures, was about 10%, which amounts to a saving of \$240.000 in less than one year. There are many any other applications in test use or more or less regular use. A rapidly growing application type is "help desk systems" [Simoudis-92], where basically case-based indexing and retrieval methods are used to retrieve cases, which then are viewed as information chunks for the user, instead of sources of knowledge for reasoning.

10. Tools

Several commercial companies offer shells for building CBR systems. Just as for rule-based systems shells, they enable you to quickly develop applications, but at the expense of flexibility of representation, reasoning approach and learning methods. In [Harmon-92] four such shells are reviewed: ReMind from Cognitive Systems Inc., CBR Express/ART-IM from Inference Corporation, Esteem from Esteem Software Inc., and Induce-it (later renamed to CasePower) from Inductive Solutions, Inc. The first three of these were reviewed more thoroughly in the German AI journal [Schult-92]. The example of CBR Express and ART-IM is typical, since many vendors offer CBR extensions to an existing tool. On the European scene Acknosoft in Paris offers the shell KATE-CBR as part of their CaseCraft Toolbox, Isoft, also in Paris, has a shell called ReCall. TechInno in Kaiserslautern has S3-Case, a PATDEX-derived tool that is part of their S3 environment for technical systems maintenance.

As an example of functionality, the ReMind shell offers an interactive environment for acquisition of cases, domain vocabulary, indexes and prototypes. The user may define hierarchical relations among attributes and a similarity measure based on them. Indexing is done inductively by building a decision tree and allowing the user to graphically edit the importance of attributes. Several retrieval methods are supported: (1) inductive retrieval matching the most specific prototype in a prototype hierarchy, (2) nearest neighbour retrieval, and (3) SQL-like template retrieval. Case adaptation is based on formulas that adjust values based on retrieved vs. new case differences. ReMind also has the capability of representing causal relationships using a qualitative model. The first commercial products appeared in 1991 including Help-Desk systems, technical diagnosis, classification and prediction, control and monitoring, planning, and design applications. ReMind is a trade mark of Cognitive Systems Inc. and was developed with the DARPA support.

ReCall is a CBR system trademark of ISoft, a Paris based AI company, and applications include help desk systems, fault diagnosis, bank loan analysis, control and monitoring. Retrieval methods are a combination of methods (1) and (2) in ReMind, but offer standard adaptation mechanisms such as vote and analogy, and a library of adaptation methods.

The KATE-CBR tool, named CaseWork, integrates an instance-based CBR approach within a tool for inductive learning of, and problem solving from, decision trees. The inductive and case-based methods can be used separately, or integrated into a single combined method. There are editor facilities to graphically build parts of the case/index structure, and to generate user dialogues. The tool has incorporated initial results on integration of case-

based and inductive methods from the INRECA project [Manago-93]. Some academic CBR tools are freely available, e.g. by anonymous ftp, or via contacting the developers.

11. Conclusions and Future trends

Summarizing the paper, we can say that case-based reasoning (CBR) puts forward a paradigmatic way to attack AI issues, namely problem solving, learning, usage of general and specific knowledge, combining different reasoning methods, etc. In particular we have seen that CBR emphasizes problem solving and learning as two sides of the same coin: problem solving uses the results of past learning episodes while problem solving provides the backbone of the experience from which learning advances. The current state of the art in Europe regarding CBR is characterized by a strong influence of the USA ideas and CBR systems, although Europe is catching up and provides a somewhat different approach to CBR, particularly in its many activities related to integration of CBR and other approaches and by its movement toward the development of application-oriented CBR systems.

The development trends of CBR *methods* can be grouped around four main topics: Integration with other learning methods, integration with other reasoning components, incorporation into massive parallel processing, and method advances by focusing on new cognitive aspects. The first trend, integration of other learning methods into CBR, forms part of the current trend in ML research toward *multistrategy learning* systems. This research aims at achieving an integration of different learning methods (for instance case-based learning and induction as is done in the MMA and INRECA systems) into a coherent framework, where each learning method fulfils a specific and distinct role in the system. The second trend, integration of several reasoning methods aims at using the different sources of knowledge in a more thorough, principled way, like what is done in the CASEY system with the use of causal knowledge. This trend emphasizes the increasing importance of knowledge acquisition issues and techniques in the development of knowledge-intensive CBR systems, and the European Workshop on CBR showed a strong European commitment towards the utilization of knowledge level modeling in CBR systems design.

The massive memory parallelism trend applies case-based reasoning to domains suitable for shallow, instance-based retrieval methods on a very large amount of data. This direction may also benefit from integration with neural network methods, as several Japanese projects currently are investigating [32]. By the fourth trend, method advances from focusing on the cognitive aspects, what I particularly have in mind is the follow-up of work initiated on creativity (e.g. [Schank-89]) as a new focus for CBR methods. It is not just an 'application type', but a way to view CBR in general, which may have significant impact on our methods.

The trends of CBR *applications* are clearly that we initially will see a lot of help desk applications around. This type of systems may open up for a more general coupling of CBR - and AI in general - to information systems. The use of cases for human browsing and decision making, is also likely to lead to increased interest in intelligent computer-aided learning, training, and teaching. The strong role of user interaction, of flexible user control, and the drive towards total interactiveness of systems (of 'situatedness', if you like) favours a case-based approach to intelligent computer assistance, since CBR systems are able to continually learn from, and evolve through, the capturing and retainment of past experi-

ences.

Case-based reasoning has blown a fresh wind and a well justified degree of optimism into AI in general and knowledge based decision support systems in particular. The growing amount of ongoing CBR research - within an AI community that has learned from its previous experiences - has the potential of leading to significant breakthroughs of AI methods and applications.

References

- Aamodt, A., (1989) Towards robust expert systems that learn from experience - an architectural framework. In John Boose, Brian Gaines, Jean-Gabriel Ganascia (eds.): *EKAW-89; Third European Knowledge Acquisition for Knowledge-Based Systems Workshop*, Paris, July 1989. pp 311-326.
- Aamodt, A. (1991). *A knowledge-intensive approach to problem solving and sustained learning*, Ph.D. dissertation, University of Trondheim, Norwegian Institute of Technology, May 1991. (University Microfilms PUB 92-08460)
- Aamodt, A., Explanation-driven case-based reasoning, in: S. Wess, K. Althoff, M. Richter (eds.): *Topics in Case-based Reasoning*. Springer Verlag, 1994, pp 274-288.
- Aamodt, A. and Plaza, E. Case-Based Reasoning: Foundational issues, methodological variations, and system approaches, *AI Communications*, Vol.7, No.1, March (1994) 39-59.
- Allen, B.P, Case-based reasoning: Business applications, *Communication of the ACM* 37 (3), 1994, 40-44.
- Althoff, K.D (1989). Knowledge acquisition in the domain of CNC machine centers; the MOLTKE approach. In John Boose, Brian Gaines, Jean-Gabriel Ganascia (eds.): *EKAW-89; Third European Workshop on Knowledge-Based Systems*, Paris, July 1989. pp 180-195.
- Bareiss, R. (1989). *Exemplar-based knowledge acquisition: A unified approach to concept representation, classification, and learning*. Boston, Academic Press.
- Brown, B. and Lewis, L. (1991): A case-based reasoning solution to the problem of redundant resolutions of non-conformances in large scale manufacturing. In: R. Smith, C. Scott (eds.): *Innovative Applications for Artificial Intelligence 3*. MIT Press.
- CBR-91, *Proceedings from the Case-Based Reasoning Workshop*, Washington D.C., May 8-10, 1991. Sponsored by DARPA. Morgan Kaufmann.
- EWCBR-93, *First European Workshop on Case-based Reasoning, Posters and Presentations*, 1-5 November 1993. Vol. I-II. University of Kaiserslautern.
- The FABEL Consortium (1993): *Survey of FABEL*. FABEL Report No. 2, GMD, Sankt Augustin.
- Hammond, K.J (1989): *Case-based planning*. Academic Press.
- Harmon, P. (1992): Case-based reasoning III, *Intelligent Software Strategies*, VIII (1).
- Hennessy, D. and Hinkle, D. (1992). Applying case-based reasoning to autoclave loading. *IEEE Expert* 7(5), pp. 21-26.
- Hinrichs, T.R. (1992): *Problem solving in open worlds*. Lawrence Erlbaum Associates.
- IEEE Expert* (1992): 7(5), Special issue on case-based reasoning. October 1992
- Keane, M. (1988): Where's the Beef? The Absence of Pragmatic Factors in Pragmatic Theories of Analogy In: *Proc. ECAI-88*, pp. 327-332
- Kedar-Cabelli, S. (1988): Analogy - from a unified perspective. In: D.H. Helman (ed.), *Analogical reasoning*. Kluwer Academic, 1988. pp 65-103.
- Kitano, H. (1993): Challenges for massive parallelism. *IJCAI-93, Proceedings of the Thirteenth International Conference on Artificial Intelligence*, Chambéry, France, 1993. Morgan Kaufman 1993. pp. 813-834.
- Kolodner, J. (1983a): Maintaining organization in a dynamic long-term memory. *Cognitive Science*, Vol.7, s.243-280.
- Koton, P. (1989): *Using experience in learning and problem solving*. Massachusetts Institute of Technology, Laboratory of Computer Science (Ph.D. diss, October 1988). MIT/LCS/TR-441. 1989.
- López, B. and Plaza, E. (1990): Case-based learning of strategic knowledge. Centre d'Estudis Avançats de Blanes, CSIC, Report de Recerca GRIAL 90/14. Blanes, Spain, October 1990.
- Manago, M., Althoff, K-D. and Traphöner, R. (1993): Induction and reasoning from cases. In: *ECML - European Conference on Machine Learning, Workshop on Intelligent Learning Architectures*. Vienna, April 1993.
- Plaza, E. and López de Mántaras, R (1990): A case-based apprentice that learns from fuzzy examples. *Proceedings, ISMIS, Knoxville, Tennessee*, 1990. pp 420-427.

- Porter, B. and Bareiss, R. (1986): PROTOS: An experiment in knowledge acquisition for heuristic classification tasks. In: *Proceedings of the First International Meeting on Advances in Learning (IMAL)*, Les Arcs, France, pp. 159-174.
- Richter, A.M. and Weiss, S. (1991): Similarity, uncertainty and case-based reasoning in PATDEX. In R.S. Boyer (ed.): *Automated reasoning, essays in honour of Woody Bledsoe*. Kluwer, pp. 249-265.
- Rissland, E. (1983): Examples in legal reasoning: Legal hypotheticals. In: *Proceedings of the Eighth International Joint Conference on Artificial Intelligence, IJCAI*, Karlsruhe.
- Ross, B.H. (1989): Some psychological results on case-based reasoning. *Case-Based Reasoning Workshop*, DARPA 1989. Pensacola Beach. Morgan Kaufmann. pp. 144-147).
- Schank, R. (1982): *Dynamic memory; a theory of reminding and learning in computers and people*. Cambridge University Press.
- Schank, R. and Leake, D. (1989): Creativity and learning in a case-based explainer. *Artificial Intelligence*, Vol. 40, no 1-3. pp 353-385.
- Schult, T. (1992): Werkzeuge für fallbasierte systeme. *Künstliche Intelligenz* 3(92).
- Sharma, S., Sleeman, D (1988): REFINER; a case-based differential diagnosis aide for knowledge acquisition and knowledge refinement. In: *EWSL 88; Proceedings of the Third European Working Session on Learning*, Pitman. pp 201-210.
- Simoudis, E. (1992): Using case-based reasoning for customer technical support. *IEEE Expert* 7(5), pp. 7-13.
- Simpson, R.L. (1985): A computer model of case-based reasoning in problem solving: An investigation in the domain of dispute mediation. Technical Report GIT-ICS-85/18, Georgia Institute of Technology.
- Stanfill, C and Waltz, D. (1988): The memory based reasoning paradigm. In: *Case based reasoning. Proceedings from a workshop*, Clearwater Beach, Florida, May 1988. Morgan Kaufmann Publ. pp.414-424.
- Strube, G. and Janetzko, D. (1990): Episodisches Wissen und Fallbasierte Schliessen: Aufgabe für die Wissensdiagnostik und die Wissenspsychologie. *Schweizerische Zeitschrift für Psychologie*, 49, 211-221.
- Sycara, K. (1988): Using case-based reasoning for plan adaptation and repair. *Proceedings Case-Based Reasoning Workshop, DARPA*. Clearwater Beach, Florida. Morgan Kaufmann, pp. 425-434.
- Veloso, M.M. and Carbonell, J. (1993): Derivational analogy in PRODIGY. In *Machine Learning* 10(3), pp. 249-278.
- Venkatamaran, S., Krishnan, R. and Rao, K.K. (1993): A rule-rule-case based system for image analysis. In: *First European Workshop on Case-based Reasoning, Posters and Presentations*, 1-5 November 1993. Vol. II. University of Kaiserslautern, pp. 410-415.