In: *Artificial Intelligence in Medicine Journal*, Vol. 12, No. 1, Jan. 1998, 25-41

Case-Based Reasoning for Medical Decision Support Tasks: The INRECA Approach

Klaus-Dieter Althoff[1], Ralph Bergmann[1], Stefan Wess[1],
Michel Manago[2], Eric Auriol[2],
Oleg. I. Larichev[3], Alexander Bolotov[3], Yurii I. Zhuravlev[4], Serge I. Gurov[4]

[1] Centre for Learning Systems and Applications, Dept. of Computer Science,
University of Kaiserslautern, PO Box 3049,
D-67653 Kaiserslautern, Germany
Phone: +49 631 205 3360, Fax: +49 631 205 3357,
Email: {althoff,bergmann}@informatik.uni-kl.de

[2] AcknoSoft, 58a, rue du Dessous-des-Berges, F-75013 Paris, France
Phone: +33 1 44248800, Fax: +33 1 44248866,
Email: manago@ibpc.fr

[3] Institute for Systems Analysis, Russian Academy of Sciences, 9 pr. 60-let Octiabrya,
117312 Moscow, Russia
Phone: +7 095 135-8503, Fax: +7 095 938-2209,
Email: larichev@glas.apc.org

[4] Russian Academy of Sciences, Computing Center, ul. Vavilova, d. 40, 117967
Moscow, Russia
Phone: +7 095 135-6231, Fax: +7 095 135-6159,
Email: Gurov@ccas.ru

In: *Artificial Intelligence in Medicine Journal*, Vol. 12, No. 1, Jan. 1998, 25-41

Abstract

We describe an approach for developing knowledge-based medical decision support systems based on the rather new technology of case-based reasoning. This work is based on the results of the Inreca European project and preliminary results from the Inreca+ project which particularly deals with medical applications. One goal was to start from case-based reasoning technology for technical diagnosis, as it was available among the partners, and 'scale-up' to more general non-technical decision support tasks as typically given in medical domains. Inreca technology is used to build an initial decision support system at the Russian Toxicology Information and Advisory Center in Moscow for diagnosing poison cases that are caused by psychotropes.

Keywords

Case-based reasoning, induction, medical decision support, toxicology domain

In: *Artificial Intelligence in Medicine Journal*, Vol. 12, No. 1, Jan. 1998, 25-41

1. Introduction

In this article we describe an approach for developing knowledge-based medical decision support systems based on the rather new technology of *case-based reasoning* (CBR). CBR is an approach for solving problems based on solutions of similar past cases [1; 20]. A *case* consists at least of a problem description (e.g., symptoms) and a solution (e.g., a diagnosis or a therapy). Cases are stored in a database of cases called *case base*. To solve an actual problem a notion of *similarity* between problems is used to *retrieve* similar cases from the case base. The solutions of these found similar cases are used as starting points for solving the actual problem at hand.

While in technical domains CBR systems for decision support and diagnostic problem solving are already in daily use in a lot of companies [4], this is much less the case for medical decision support tasks. Well known examples from the literature include using CBR for expertise relocation in developing countries [30], using CBR for intelligent retrieval of radiology images [24], integrating various reasoning capacities of physicians like logical, deductive, uncertain and analogical reasoning [26], trading case-specific and model-based knowledge in bone healing [38], utilizing CBR for trend prognosing for medical problems [36], automated image interpretation of myocardial perfusion scintigrams [19], and using CBR for selecting of an antibiotics therapy [37]. The GS.52 system [17] is an example that is in practical use as a real-life application; it uses CBR to address the domain of dismorphic syndromes.

Van Bemmel [40] differentiated between clinical support systems and decision support systems. He pointed out that clinical support systems, e.g. imaging systems, are very successful, offering data the user needs for decision making. By contrast, medical decision support systems are not or only partially accepted. One reason is that developers of decision support systems do not always realize who the future users of their systems will be. According to van Bemmel [40] medical decision support systems could be successfully employed for enhancing primary data reliability, consistency, and completeness. They need to be integrated with clinical support systems and they need to be part of routine clinical practice. Of course, physicians will only accept decision support systems if they have proven to increase the quality of patient care.

The work reported here is based on the results of the Inreca ('Induction and Reasoning from Cases') Esprit project of the European Union1 and preliminary results from the Inreca+ ('Integrating Induction and Case-Based Reasoning for Diagnostic Problems with Focus on Medical Domains') project2. Both projects aim at developing

[1] The partners of the Inreca-project are AcknoSoft (France; prime contractor), tecInno (Germany), Irish Medical Systems (Ireland), and the University of Kaiserslautern (Germany).

[2] The partners of the Inreca+ project are AcknoSoft (France; prime contractor),

information technology for building systems for solving diagnosis and identification problems by using past history, where Inreca+ especially focuses on medical problems. In Inreca, the CBR system design and implementation was driven by a thorough analysis of the state of technology in CBR [4; 10].

In this article we address the topic of developing case-based decision support systems for diagnosing intoxications by drugs. The main goals are to reduce the time required to come to a decision, particularly, in an emergency case, to compensate unavoidable lacks of experience of young medical staff, and to distribute available experience to different sites.

Such a system would have potential use in the Russian Toxicology Information and Advisory Center in Moscow, in many Russian hospital ambulances, and a number of toxicology centers that still need to be built at central places in Russia. In addition, a case-based expert system could also be useful for urgent ambulance consultations in many European countries (a.o.). It is known that every year Russia has more intoxication cases than any other country in Europe. Therefore it is reasonable to use valuable experience of the best Russian toxicologists.

The next section presents the requirements of decision support tasks in medical domains and illustrates them using a concrete example from the toxicology domain developed at the Russian Academy of Sciences. Section three presents the Inreca CBR approach and its integration with inductive machine learning techniques. Section four evaluates this approach by discussing how the before mentioned requirements can be fulfilled and outlines a procedure for developing a medical decision support application. Finally, the results of an evaluation of two initial systems – one in the toxicology domain – are presented. Section five discusses related work on medical decision support systems and in section six an outlook for further work is given.

2. Requirements for a Medical Decision Support Systems

We will first describe a concrete subject area from the field of medical decision support applications. This domain will be used for illustrating the requirements derived afterwards.

University of Kaiserslautern, Institute of Mathematics (Moldova), Reliable Software Inc. (Belarus), All-Russian Institute for Scientific and Technical Information (Russia), and Russian Academy of Sciences (Russia).

2.1. A Decision Support System for Diagnosing Poison Cases Caused by Psychotropes

As a main subject domain we present a special problem of medical diagnosis: decision support for therapy selection in case of intoxication with drugs, in particular with psychotropes. This requires a fast diagnosis of a practical situation and the choice of a therapy. The reason for selecting this clinical discipline is that here the information problem comes to the forefront, where the major difficulty in treatment of patients with acute poisoning is usually poison identification in the patient's organism and circumstances of an exposure (dose, routes of administration, time upon injection, etc.). The goal is to create a decision support system which is useful for the following purposes:

- To be used by an ambulance physician in the cases of an intoxication by medicines.
- To be used by physicians whose specialty lies outside the domain of toxicology[3]. Having got the clinical symptoms of an intoxicated patient the system finds the substance taken by him and offers the necessary course of actions.

1	Ethanol
2	Barbiturates
3	Methanol
4	Amynotryptelene
5	Malathion
6	Acetic acid
7	Parathion
8	Dichloroethane

Table 1 – Types of drugs considered

There appears also to be an additional use even for domain experts in two directions. First a decision support system can confirm their decision while they work in the call-center (one expert usually processes 10-30 poison cases per day where 3-5 cases are

[3] The information problem is always present because physicians cannot learn all the types of poisons at the university, of which in general more than a million exist.

really difficult ones[4]), second the respective expert can extend his knowledge in the parts of the toxicology domain he is not familiar with (even the best expert does not cover the whole domain).

No	Attribute	No	Attribute
2	Age	279	Pain in epigastrium
3	Sex	282	Sickness
8	Beds/days	287	Vomiting
11	Outcome	291	Coffee-ground vomit
179	Respiration rate	293	Toxic gastroenteritis
230	Systolic BP	343	Clear consciousness
231	Diastolic AP	344	Impaired consciousness
232	Pulse AP	345	Mental confusion
238	Pulse on admition	346	Loss of consciousness
239	Tachycardia	347	General weakness
1366	Type of poison	348	Lethargy
13	Death	349	Retardeness
46	Satisfactory state	350	Flabbiness
47	Medium severity state	351	Excitement
48	Severe state	352	Psychic excitement
60	Pale dermal cover	353	Motor excitement
62	Dermal hyperemia	355	Inadequate consciousness
63	Dermal cianosys	356	Adynamia
64	Cyanosis of nasolabial triangle	358	Sopor
65	Dermal acrocyanosis	360	Coma
74	Cold ski	361	Deep coma
75	Sweating	362	Initial coma
106	Hyperemia of oral cavity	366	Coma duration
113	Edema of oral cavity	373	Blurred speech
146	Thorax muscle rigidit	374	Poorly responsive
149	Spontaneous myofibrillations	375	Nonresponsive
151	Hand tremor	376	Disoriented in space
153	Asthenia	377	Disoriented in time
165	Reflexes present	378	Noncritical
168	Disturbed respiration	379	Headache
169	Disturbed aspiration-obturation	380	Dizziness
173	Rigid respiration	400	Medium-size pupils
189	Rale	401	Meyosis
191	Dry rale	403	Medriasis
200	Bronchorrhea	415	Live photoreaction of pupils
205	Pulmonaty hperhydration	416	Zero photoreaction of pupils
217	Bronchopneumonia	420	Preserved pain reaction
237	Collaps (AP < 90 mm Hg)	421	Reduced pain reaction
252	Decompressed shock	422	Zero pain reaction
256	Mouth odour	423	Cough reflex preserved
257	Alcohol odour	426	Drunken man behaviour
267	Pain during esophagus palpation	1034	Torpor
268	Pain during swallowing	1061	Salivation

[4] The most difficult problems correspond to a combination of different types of poison and also for the background of alcohol.

Table 2 – Attributes considered in the toxicology domain

As part of the Inreca+ project, a concrete CBR application was built based on actual case data on acute poisoning collected by the Toxicology Information and Advisory Center of the Russian Federation Ministry of Health and Medical Industry. Table 1 shows the different 8 types of medicines that are considered in this application.

Table 2 shows a list of 86 parameters that have been identified to be important for this diagnosis task. We did not include into the clinical information list any sort of laboratory data analysis because, as a rule, when a toxicology laboratory is available for a physician and there is enough time to carry out the analysis it gives the right answer and the computer assistant is not needed at all.

2.2. Requirements

In the following we describe a number of requirements that arise for (medical) decision support problems in general and for a decision support system in the toxicology domain in particular.

- Short response time:

  One of the most important requirements for a (critical) medical decision support system is the response time. The system must be able to present a diagnosis/therapy based on the observed symptoms within less than a minute. Only a fast problem solution enables the physician to immediately start a therapy which might be crucial for saving the patient's life. This is particularly true for toxicological cases. The initial therapeutic action must be as adequate as possible which requires a quick and accurate prediction of the toxic cause without waiting for the toxicological analysis [25].

- Justifiability of results:

  A result (e.g., a selected therapy) presented by the system must be justified by the system in a way that a physician can validate the outcome and judge its accuracy. The justification presented by the system should be in a form the physician is familiar with in order to allow a fast decision process. Thus, the underlying problem solving process needs to be transparent.

- Dealing with incomplete information:

Incomplete information is a basic characteristic of medical domains. Often the value of certain attributes cannot be acquired because the information is simply not available at the time of diagnosis or the required tests would take too long, are too risky, or are too expensive. This is also particularly true in the toxicology domain. For example, a comatose patient is not able to provide a physician with any information such as past diseases. On the other hand, a comatose patient requires a very fast treatment and the acquisition of a broad number of additional parameters would usually take much too long.

- Dealing with vague relationships:

  In medical domains, many relationships between certain attribute values are of a vague nature. Often, different degrees of the occurrence of a certain symptom must be distinguished, but a clear definition of these degrees and their relationship is often not available. A typical example that occurs in the toxicology domain are the attributes describing the state of consciousness of the patient. There is usually no clear definition of the terms ‘clear consciousness’, ‘impaired consciousness’, or ‘mental confusion’. However, it is clear that the state ‘mental confusion’ is closer to ‘impaired consciousness’ than to ‘clear consciousness’. A decision support system should be able to make use of this vague information.

- Dealing with measured values and conceptual terms:

  In ‘medical domains’, the attributes that describe a patient usually contain values that are the result of some measurement (e.g., the systolic blood pressure) and values that describe some conceptual terms (e.g., ‘impaired consciousness’). While measured values are usually represented as numeric values (e.g., 120 mm Hg), conceptual terms are usually binary or n-ary features. Consequently, a generic decision support system must be able to efficiently deal with both kinds of attributes.

3. The Solution: The Inreca Approach

We now describe in more detail an approach to building decision support systems which reason from cases. The employed application areas demonstrate the requirements mentioned above.

Generally accepted characteristics of CBR systems already meet some of the above mentioned requirements. CBR systems are in principal able to deal with incomplete information (e.g., unknown attribute values), make use of vague relationships by means of similarity measures, as well as to allow numeric and symbolic attributes. However, CBR approaches that fulfill all these requirements together tend to lose the ability of efficient retrieval when the case base grows. The reason for this is that case-based systems typically *interpret* the specific knowledge contained in all the cases at run time, i.e., during the consultation of the system. The more cases arise, the more computational effort must be spent for their interpretation. This is a major problem when applying CBR to real-life applications and makes it particularly hard to achieve the requirement 'short response time'. This is a very critical aspect with respect to user acceptance.

The Inreca approach, presented in some detail in this section, presents a successful solution to this important problem. The Inreca system allows to *compile* some of the specific knowledge contained in the cases into more general rules that can be efficiently evaluated and which keep the consultation time of the system short. This approach can be viewed as an integration [9] between classical CBR and inductive machine learning approaches [29].

3.1. Cases and Similarity Measures

In a case-based decision support system, a case describes a past situation in which a particular decision was taken. In medical domains, a case contains a description of the symptoms observed during examination of a patient as well as the diagnosis or the treatment that was identified, e.g., by a physician. For example, in the toxicology domain, the symptoms that are recorded in a case are those shown already in Table 2. Each symptom appears as an *attribute* in the case representation, each of which has a particular type assigned, defining the value range of the attribute. For example, the symptom No. 232 from Table 2, named 'Pulse AP', can be represented as an integer attribute, while the symptom 'Sweating' can be represented as a Boolean attribute. The diagnosis that was identified in a particular case is also turned into an attribute (called *target attribute*) recorded as part of the case representation. In our domain, the possible values for this attribute (No. 1366 'Type of poison' in Table 2) have been already shown in Table 1.

When a new problem must be solved, e.g., a patient with an unknown intoxication must be diagnosed, some of the symptoms must be checked and noted as a new problem case. The CBR process than proceeds by searching for the most similar known case from the case base. For this purpose, the similarity between two cases (the problem case and the case in the case base) must be defined through the similarity of the attributes used in the case representation (except for the target attribute).

In general, different approaches to the definition of similarity are known in the CBR literature [41]. A very common approach is to define similarity through a *similarity measure $SIM(X,Y)$,* which is a function that assigns a pair of cases X,Y a real-valued number out of the range [0..1]. A high value represents a high similarity between X and Y. Such a similarity measure can be defined as a weighted sum of the results of *local similarity measures* available for each attribute. A local similarity measure $sim_a(x,y)$ compares two values of the single attribute a and returns its similarity in terms of a real-valued number out of the range [0..1]. The following formula summarizes this similarity definition:

$$SIM(X,Y) = \sum_{a \in A} w_a \times sim_a(X_a, Y_a)$$

Here, A denotes the set of all attributes that occur in the case representation and w_a is a weight for the attribute a.

3.2. The Inreca-Tree: A Data Structure for Indexing Cases

Now, we present the Inreca approach, which allows to find the most similar case(s) in an efficient way. The efficiency of case retrieval is of high importance and a serious problem if a case base has reached a considerable size. The core idea behind our approach is a new indexing structure we call *Inreca-Tree*. This indexing structure is based on the concept of the k-d tree [15], which is a multi-dimensional binary search tree. Such a tree – automatically constructed during system building, i.e., before the consultation is performed – structures the space of available cases based on their observed density. During retrieval, the tree focuses the search for the most similar case(s) and thereby avoids the investigation of all available cases.

3.2.1. Branches of an Inreca-Tree

The Inreca-Tree is an n-ary tree in which the branches represent constraints for certain attributes of the cases. Since we need to handle ordered and unordered value ranges as well as unknown attribute values, we introduce different kinds of branches. These branches are shown in Fig. 1.

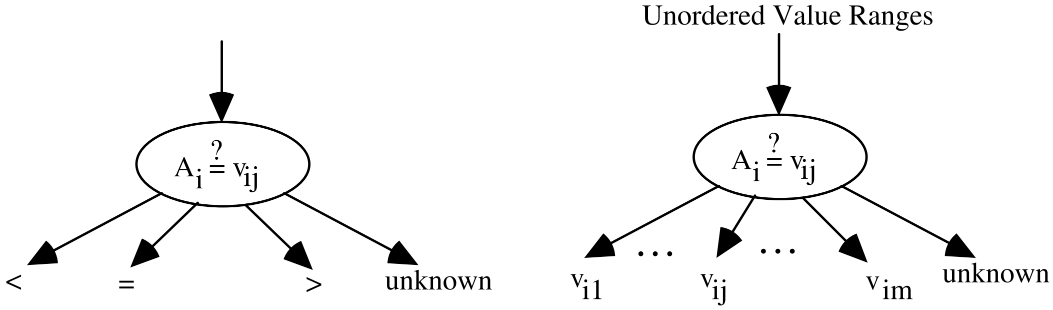

Figure 1 - Branches of the Inreca-Tree for ordered and unordered value ranges

Attributes with an ordered value range partition the set of cases with respect to a certain value v_{ij}. The cases are divided into sets of cases in which the respective attribute has a value which is smaller than v_{ij}, larger than v_{ij}, equal to v_{ij}, or unknown. Attributes with an unordered (finite) value range partition the cases into one set for each value contained in the value range and one additional set for cases in which the attribute value is unknown. The leaves of an Inreca-Tree (we call them *buckets* as in a k-d tree) contain all cases that fulfill all constraints that occur in the path from the root of the tree to the respective leaf.

Fig. 2 shows an example of an Inreca-Tree for the toxicology domain. The top of the tree shows a branch node for the attribute ‘Pulse’, which holds values from the ordered type ‘integer’. This node partitions the set of available cases into three subsets in which the patient‘s pulse is less than 40, equal to 40, higher than 40, or unknown. The next node partitions the set of cases with a pulse higher than 40 into three subsets, depending on the ‘Coma’ attribute. At the leaf nodes of the tree, some buckets are displayed which contain the respective cases. Each case in a bucket fulfills all the constraints recorded from the root of the tree to the bucket. For example, in case 3, the patient is known to have a pulse higher than 40, and coma and sweating is not observed.

3.2.2. Basic Procedure for Building an Inreca-Tree

The Inreca-Tree is built prior to the first consultation of the system. It is assumed that all available cases are already stored in the case base (CB) and are accessible. The basic recursive procedure for building an Inreca-Tree is quite simple and is described in Fig. 3. Every node within the tree represents a subset of the case base and the root node represents the whole case base. Every inner node partitions the represented case set into disjoint subsets.

The described algorithm uses the three subprocedures *SelectAttribute, SelectValue,* and *Split?.*

The *Split?* procedure determines the depth of the Inreca-Tree. It uses a criterion to determine whether a new sub-tree must be generated or whether the cases in the current partition are collected in a bucket (leaf node). Different termination criteria are possible, e.g., based on the size of the partition, based on the distribution of the diagnoses that occur in the cases contained in the partition, etc.

The *SelectAttribute* procedure determines the attribute that is used for partitioning the case base in the current branch. In case of an attribute with an ordered value range, the procedure *SelectValue* determines the exact value used for splitting the case base.

Different choice criteria for attributes and values can be used for constructing an Inreca-Tree [42]:

- The inter-quartile distance [21] used in a classical k-d tree computes the distance between the first and the third quartile. For a given attribute, quartiles divide the set of values that occur in the current set of cases into four sets of equal size. The first quartile contains those cases with the smallest values for the attribute, the second quartile contains those cases with the next higher values for the attribute and so forth.
- The average similarity measure estimates the dispersion of cases with respect to a given partition of the case base. The partitioning attribute is the one with the greatest average similarity for a chosen partition.
- The information gain measure [39] computes the difference of *entropy* between a case base and its partition built from a particular attribute. The entropy evaluates the impurity of a set of cases with respect to the target attribute (diagnosis). The procedure selects the attribute that provides the best information gain.

It has been shown that these different criteria have an impact on the retrieval efficiency [42]. An optimal criterion must be selected for the current application at hand.

3.3 Retrieval with the Inreca-Tree

We now explain how the Inreca-Tree can be used for efficiently retrieving the most similar case(s) for a given new problem case. The search is done via a recursive tree search procedure according to the global similarity measure *SIM(X,Y)*. During the search, the two test procedures '*Ball-Overlap-Bounds*" (BOB) and '*Ball-Within-Bounds*" (BWB) are used to focus on the relevant search region. These procedures are extensions of equivalent procedures known from k-d trees [15].

While the search is performed, a priority list is maintained which contains the k most similar cases known so far, together with their similarity to the problem case. This list is updated when new cases are visited. The recursive procedure (beginning with the root node) runs as follows:

- If the current node is an inner node, the procedure is first iterated on one of the child nodes. The procedure follows the branch whose constraint is fulfilled by the value of the respective attribute contained in the problem case.
- If the current node is a leaf one, the priority list is updated according to the similarity of the cases belonging to the bucket with the problem case. Then the BWB test checks whether it is guaranteed that all k-nearest neighbors have been found. If this is the case, the search is terminated. If this is not the case, the search backtracks to a parent node and considers to investigate an additional portion of the tree.

- If the current node is an inner node that is reached through backtracking from a child node, a test is executed to look whether it is necessary to inspect one of the other child nodes. This is done through the BOB test. If this test is false, the partition of the other child nodes cannot contain any k-nearest neighbors with respect to the query. Therefore, they are not examined further and the search backtracks to the parent of the current node. If this test is true, the procedure is iterated on these nodes, i.e., it continues the search in the respective sub-tree.

The BOB and BWB test procedures have relatively simple geometrical interpretations (Fig. 4 and 5):

For these tests, a m-dimensional ball is drawn around the current query. The radius of this ball is determined by the similarity of the least similar case which is currently in the priority list (*k*th most similar case). Every case that is outside this ball is less similar than the currently known k most similar cases and need consequently not to be visited.

In order to recognize whether a node is ‘of interest’ (it may contain some candidates), the geometrical bounds of the node are used to define a test point that is most similar to the current query but still lies within the geometrical bounds of the current node. If this test point is in the ball it means that the ball overlaps with the node and then there maybe a candidate to the priority list in this node. This search is conducted by the BOB test. Fig. 4 shows a two-dimensional BOB test, where the test point X_{min1} belonging to the node K_1 is in the ball (and therefore it may be of interest to explore this node), but where the test points (X_{min2} and X_{min3}) for K_2 and K_3 are not in the ball. Consequently, K_1 requires further exploration, while K_2 and K_3 need not to be visited.

The other question to answer is, whether the ball around the query X_q lies completely in the geometric bounds of already explored nodes (let us call this set the ‘bounding box’), or not. To this test, one verifies if the bounding box has no intersection with the ball. Fig. 5 shows a successful two-dimensional BWB test.

In the best case, when no backtracking is required, retrieval with the Inreca-Tree leads to a running time that is proportional to the depth of the tree; the retrieval effort is O(log(n)), with n being the number of cases in the case base. However, in the worst case in which backtracking is required for every node, the whole tree must be investigated, leading to a retrieval effort of O(n). Experiments conducted in several non-medical domains have clearly shown that in the average case important reductions in the retrieval speed compared to a linear retrieval approach (complexity O(n)) can be achieved [41].

3.4. Compiling Inductive Knowledge into the Similarity Assessment

We now describe an approach that allows to further speedup retrieval when the number of cases increases drastically. The idea behind is to move from an approach in which the knowledge contained in the cases is interpreted to an approach that compiles some of the knowledge contained in the cases into rules that further improve the retrieval efficiency. The generation of general rules out of cases (often called examples) has been intensively studied in the field of inductive machine learning [29]. We now briefly introduce a particular approach from this field, namely the top-down induction of decision trees (TDIDT: [33; 34]).

3.4.1. Induction of Decision Trees

A decision tree is an n-ary tree whose inner nodes are labeled with an attribute. The links that lead from a node to its child nodes are labeled with a value that belongs to the value range of the attribute the node is labeled with. Each leaf node of a decision tree is labeled with a *decision class*, e.g. the diagnosis referring to the kind of intoxication. Obviously, a decision tree is very similar to the previously described Inreca-Tree. The main difference in this data structure is that the leaf node of the Inreca-Tree contains a set of cases (bucket), while a leaf node of a decision tree contains a decision class[5]. However, the main difference between the Inreca-Tree and a decision tree lies in the way it is used during problem solving. A decision tree is only traversed once from the root node to a leaf node. At every node, the link is followed which matches the respective attribute value in the problem case. BOB and BWB tests are not used. The decision class noted at the leaf node is then returned as result of the decision tree consultation. However, problem solving with a decision tree can be compared with a retrieval using an Inreca-Tree in which no backtracking occurs.

The construction of a decision tree is also very similar to the construction an Inreca-Tree. The partitioning procedure attempts to find the most informative attribute in order to create the shortest tree possible. Traditionally, it uses a hill-climbing search strategy and a preference criterion often based on the information gain [39; 33], like the C4.5-system [34]. At each node in the decision tree, the criterion is evaluated for all the attributes which are relevant and the one is picked which yields the highest increase of the information gain measure.

The efficiency of decision tree consultation is one of its main advantages. However, decision tree consultation has also several drawbacks which we expect to lead to major problems [9] particularly in medical decision support tasks. One problem stems from the fact that a decision tree consultation completely ignores the concrete cases from which the decision tree was built. Information contained in the cases but not contained in the branches of the tree is not used for decision making. This is a major disadvantage if only a small number of cases is currently available.

[5] On a more technical level, their are some more differences, e.g. in the kind of branches that are allowed in a decision tree. However, these differenes are not important for the further discussion.

3.4.2. Seamless Integration of Case-Based Reasoning and Induction of Decision Trees.

The Inreca-Tree can be used to realize inductive reasoning as well as nearest-neighbor retrieval of cases. Now we explain how an intermediate approach between TDIDT and CBR can be reached. We start with pure CBR and move ‘more to the side of induction’ as more cases arise. This approach is therefore called *seamless integration* between CBR and induction [4]. With this procedure, we want to avoid the problem that the retrieval time increases if more and more cases are contained in the case base. The more cases are available, the more reliable is the classification by the induced decision tree, because the inductive hypothesis is based on a larger set of known examples. Additionally, traversing a decision tree is much faster than nearest-neighbor retrieval because no backtracking in the tree is required.

The general idea of how to realize this shift from CBR to induction is to include more and more nodes in the Inreca-Tree for which backtracking is not required. As an example consider the Inreca-Tree shown in Fig. 2. In this tree, backtracking may be eliminated in the ‘yes-branch’ of the ‘Coma’ node. If coma is observed in the current problem case, then all cases not describing comatose patients can be ignored, because some special treatment is always needed in this situation. All cases about patients who do not have coma can therefore be ignored.

We have developed an approach that allows to compile such general knowledge, extracted from the decision tree, into the local similarity measure [41; 2]. This procedure is reasonable only if a sufficient number of cases is available. However, we strongly propose that even in that case such induced knowledge is carefully validated by an expert before it is used.

Using this approach a new procedure for developing an application that integrates CBR and TDIDT in a seamless way can be realized as summarized in Fig. 6.

4. The Evaluation of Inreca Technology for the Toxicology Application

For legal and ethical reasons a medical system should not be introduced into clinical practice before it has been properly evaluated [40]. Evaluation should cover all stages of the development process of a decision support system. An important point is that evaluation is a process to be continued after the introduction of a decision support system into practice, as is also done in post-marketing surveillance studies of drugs.

We describe in detail how the Inreca system, and especially the integration described in section 3, meets the requirements for medical decision support systems as presented in section 2. Then we present a plan for introducing case-based decision support systems in medical environments. Finally, some first experimental results on the evaluation on two initial CBR prototype systems are summarized in section 4.3.

4.1. Meeting the Domain and Task Requirements

We now discuss how the Inreca approach fulfills the requirements of decision support tasks in medical domains.

- Short response times:

  Efficient retrieval of cases was one of the major motivations for the development of the Inreca-Tree and particularly the seamless integration described in section 3.4.2. First of all, the indexing of a large case base by an Inreca-Tree (see section 3.2) already allows a very efficient case retrieval. Experiments in several non-medical domains have shown a significant speedup compared with other approaches to retrieval [42; 41]. Further important improvements can be achieved through the compilation of inductively learned general knowledge into the similarity measure. Even if for small case bases this improvement of efficiency is not of great importance, large case bases, which are used in real-life applications [28], require such an efficient indexing approach to fulfill the tight constraints on acceptable response time.

- Justifiability of results:

  The traditional justification of a diagnosis achieved with a CBR approach is to present the complete information contained in the most similar case. The physician who uses the system can then himself validate the similarity between the new case and the retrieved case. With the Inreca approach an alternative kind of justification is also possible. If the Inreca-Tree is viewed not as an indexing tree but as a decision tree, the user can validate the decision path followed by the system. Physicians that prefer ‘thinking in rules’ will consider this information as very valuable [22].

- Dealing with incomplete information:

  One major advantage of CBR is its ability to cope with incomplete information, i.e., unknown attribute values. The Inreca-Tree explicitly considers situations in which some attribute values in the problem case or even in a case stored in the case base are unknown. Another advantage of the Inreca approach is that it offers several strategies for the acquisition of missing information. Particularly, the Inreca-Tree provides hints about which attribute values should be acquired, based on the attributes that occur in the branch of the tree currently followed during the consultation.

- Dealing with vague relationships:

  The Inreca approach allows to use local similarity measures for expressing vague relationships between the possible values of an attribute. The similarity between two values can be quantified and coded into the local similarity measure. If a vague relationship becomes clear afterwards due to induced general knowledge, such a local similarity measures can be improved.

- Dealing with measured values and conceptual terms:

  The Inreca approach can handle numeric and symbolic attributes. While numeric attributes are usually required for handling measured values, ordered or unordered symbolic attributes are required for expressing conceptual terms. The Inreca-Tree can use both kinds of attributes for efficient indexing.

4.2 Developing Case-Based Medical Decision Support Systems

Developing CBR applications generally requires defining the area of competence (e.g., cardiology, rheumatology, toxicology, etc.), the purpose of the system (call center support, ambulance support, education, etc.), and its intended users (students, less experienced physicians, expert physicians, etc.). The next step is to select an appropriate CBR shell. A CBR shell realizes already the basic mechanisms for case representation, similarity assessment, and retrieval [4]. It also provides interfaces to other software components (e.g., databases) and to the user. For the further discussion we assume that a CBR shell based on Inreca technology is considered, such as Kate tools (AcknoSoft, France) or CBR-Works (tecInno, Germany).

Basing on the above decisions, a number of complicated, interacting development tasks must be carried out:

- Defining an appropriate case representation, i.e., selecting relevant diagnostic signs to be used as attributes, determining an appropriate value range for each attribute including the definition of the respective decision classes.
- Defining the similarity assessment, i.e., the local similarity measures and the attribute weights.
- Collecting cases.

Due to the difficulty of these development tasks an incremental development strategy is currently considered to be most successful [11]. The underlying model is comparable to the ‘spiral model’ well-known in software engineering [12]. During the development process a sequence of incrementally improved CBR prototype systems is generated. Each prototype system must be validated in order to define the steps required to further improve the system. For the development of medical decision support systems we propose to differentiate at least the following phases.

During the initial system building phase, a first and simple case representation and similarity assessment is defined and an initial set of cases is collected. This development phase requires only a limited involvement of an expert physician. The resulting initial CBR prototype system must then be analyzed according to its classification behavior. Based on this analysis the initial CBR prototype system will be revised with respect to case representation and similarity assessment.

In the next phase, the case base will be further extended and a novice physician will validate the system using a number test cases collected by him/her. This validation step will lead to further revisions or improvements of the system. This process will be iterated with a number of novice physicians, experienced physicians as well as domain experts. While novice physicians can test the system using mainly standard cases, more experienced physicians can also use more complicated and unusual cases for testing.

If the last validation step has been successful, then a pilot CBR system can be installed and used by an expert physician (e.g., in a toxicology call center). An expert is required here because (s)he is able to interpret each case in the case base and, by this, can decide whether a suggestion of the CBR system is appropriate. If the system has successfully used for some time it is possible to extend the group of possible users to less experienced physicians and, finally, to students for educational purposes.

4.3. Evaluating Two Initial CBR Prototype Systems: First Experimental Results

Up to now two initial CBR prototype systems have been built and evaluated as part of the Inreca+ project. In the following sections we present some of the first experimental results. Kate-CBR was used as CBR shell for building these initial systems. We used the standard similarity assessment provided by this shell without any domain-specific optimizations.

For the purpose of evaluation we also built decision support systems based on alternative methods, namely

- Kate-Induction, which is a commercial inductive tool based on the decision tree algorithm,
- two classification systems based on the Bayes (BC) and the linear discriminant function approach (LDFC), and
- a specialized classification approach, called algebraic approach (AS), which consists of an optimized combination of four different classifiers as described by Zhuravlev [43].

4.3.1. Initial CBR System for the Cardiology Domain

We consider the following medical classification problem from the cardiology domain, where some experiments already carried out in the scope of the work by Bolotov and Larichev [13] could be reused: the differential diagnostic of pulmonary thromboembolism (PTE) and myocardial infarction (MI). Experts gave the following set of symptoms: past history, breathing, skin color, arterial blood pressure, ECG, and lung radiography. There are three decision classes (diagnoses): I for PTE, II for MI and III for PTE in conjunction with MI.

Training Sample No. of cases	LDFC Discrepancy (%)	BC Discrepancy (%)	Kate-Induction Discrepancy (%)	Kate-CBR Discrepancy (%)
4	-	-	35.0%	15.6%
16	40.6%	27.0%	20.0%	4.7%
24	26.6%	18.0%	17.0%	4.7%
32	18.5%	17.4%	14.0%	3.1%

Table 3 – Comparison of classification discrepancy between LDFC, BC, Kate-Induction, and Kate-CBR

An initial case base of 64 cases was developed. In Table 3 the precision of the classification is given depending on the respective sample size. The discrepancy is evaluated for each method (the average numbers are given for five random samples).

CBR appears to be much better for this particular domain than induction and than LDFC and BC. Even with a small size of the training sample, CBR leads to an acceptable classification accuracy.

4.3.2. Initial CBR System for the Toxicology Domain

We developed an initial CBR system for the toxicology domain, in particular for the task of poison recognition during acute poisoning. A case data set of 459 cases, based on the eight types of drugs shown in Table 1, was acquired. In several runs, the Kate-CBR, Kate-Induction, and the AS algorithm were used on the same data sets. Table 4 presents the results of this experiment.

Algorithm	**Range of classification accuracy for different training samples**
AS algorithm	92.0 - 96.0%
Kate-Induction	78.5 - 86.6%
Kate-CBR	89.3 - 93.8%

Table 4 – Comparison of classification accuracy between AS, Kate-Induction, and Kate-CBR in the toxicology domain

It appears that the CBR approach leads already to a very high classification accuracy which is only slightly worse than the accuracy of the AS algorithm, which is highly optimized for the toxicology domain. Thus, this result is particularly notable when considering the low development effort required for applying Kate-CBR or Kate-Induction compared to the effort for developing the AS algorithm. Based on these preliminary experimental results we are quite optimistic that a systematic approach to CBR system development will lead to valuable case-based toxicological decision support system. Further developments along the lines discussed in section 4.2 will, of course, be necessary.

5. Discussion

A detailed comparison of Inreca with other CBR/CBR-related approaches and approaches to the integration of CBR and induction is given by Althoff [2].

A success story for a knowledge-based medical decision support system in the toxicology field is given by Darmoni, Massari et al. [14] who report on the SETH approach at Rouen University. The domain was chosen because drug poisoning is a frequent problem there. The aim of SETH is to give end-users specific advice concerning treatment and monitoring of drug poisoning. It simulates expert reasoning, taking into account for each toxicological task delay, sign, and dose. It is in daily use by hospital residents as telephone response support since April 1992. It is also used as an educational tool for drug poisoning. More concrete, the purpose of SETH in case of a known intoxication is to give non-toxicologist physicians better advice for the treatment and monitoring of drug poisoning, in case of an unknown intoxication the purpose is to identify products according to clinical manifestations and context. SETH uses a database on drug information and a case database on all data being entered by an end-user. The functional evaluation in October 1994 was positive [14]. The SETH approach underlines that computer based decision support in the toxicology field is very helpful. We think that using a CBR approach would allow to achieve comparable results with less development effort.

Malek, Danel and Rialle [25] describe an interesting combination of CBR and neural nets for solving toxic comas diagnostic problems. The authors compared their approach with other approaches (k-nearest neighbors, decision trees). The authors do not state if it is intended to evaluate the system in a field test. We believe that approaches that cannot guarantee to find the most similar case(s) available, because of using heuristic generalization and/or retrieval techniques as it is described in this paper, are not appropriate for such critical decision support tasks.

Puppe, Ohmann et al. [32] compared four different techniques four building medical decision support systems on acute abdominal pain cases. They stated that the diagnostic performance of a knowledge-based system depends more on the amount and quality of knowledge exploited than on the problem solving method chosen. If some piece of knowledge or data, essential for making a certain decision, would be missing from the knowledge base or case description, no problem solving method can be expected to produce satisfactory results. They concluded that the building of medical decision support systems is still an art rather than a routine task of software engineering. One result reported is that if the main goal is ‘optimizing the overall performance’ then methods basing only on case-specific knowledge can be recommended. If the main goal would be ‘optimizing the consideration of rare and serious cases’ then methods basing on general knowledge can be recommended.

Goos and Schewe [18] describe a successful CBR application to clinical rheumatology. They compared the performance of their CBR approach against an expert system based on general knowledge only, as described by Gappa, Puppe and Schewe [16]. Goos and Schewe [18] report that the CBR approach required only one third of the development effort of the earlier (general) knowledge-based system. However, the results were worse than those of the (general) knowledge-based system because the used case base was too small to allow to find all combinations of diagnoses occurring in real-life situations.

One possible résumé here could be that the CBR approach appears to have some advantages concerning system development if compared with other knowledge-based methods [7]. However, describing the similarity assessment mechanism adequately is a detailed knowledge engineering task that possibly ‘consumes’ part of the effort ‘saved’ by using previous cases.

For medical experts CBR is not more natural than other reasoning methods like rule-based reasoning. In addition, it is obvious that medical experts do not reason from cases only [22; 31; 23]. So, why do we need CBR for medical decision support systems? We believe that there are the following reasons.

CBR explicitly represents, memorizes, and reasons about cases, which are very important entities in medical contexts (partially already available electronically). Thereby, CBR inherently combines problem solving and learning. By this the system development process is supported by automatic learning techniques as well as the update/maintenance process after the initial system has already been constructed. The observation that experts of long standing use a ‘compiled form of knowledge’ [22] could be simulated by a CBR system by using induction as a means for compiling cases into general knowledge, as for instance exemplified in section 3.4.

CBR also offers a very natural approach to differential diagnostics: Physicians readily admit that the crucial point in making a diagnosis involves excluding diseases with very similar symptoms [22]. As a consequence, the actual test selection strategy, used during real problem solving, can focus on such similar cases and discriminate between these cases based on the user's answers [8; 35].

A CBR system can be used by an expert in the field for completing his/her knowledge on unusual cases. CBR technology also offers certain degrees of flexibility with respect to broadening the scope of its usage, for instance towards more general information retrieval problems (integration into clinical information systems, similarity based retrieval in databases).

6. Outlook

Ongoing activities are concerned with developing practical guidelines for users of CBR technology [6; 27; 5; 3], which include computer-assisted support for the development and maintenance of case-based applications. Interesting topics along this direction are dealing with evolving domains and a methodology for modeling similarity. The integration of general knowledge within the Inreca system in general and within the similarity assessment process in particular appears to be a step into the right direction. While such knowledge can be represented and used if available, induction can be used here to support application development by means of (inductive) analysis of the domain and the application task.

Currently the introduction of a medical decision support system at the Russian Toxicology Information and Advisory Center in Moscow is planned.

7. Acknowledgment

Funding for Inreca has been provided by the Commission of the European Union (Esprit contract no. 6322), to which the authors are greatly indebted. The partners of Inreca are AcknoSoft (prime contractor, France), tecInno (Germany), Irish Medical Systems (Ireland), the University of Kaiserslautern (Germany). For focusing on medical applications funding for ‘Inreca+’ is provided by INTAS, the international association for the promotion of cooperation with scientists from the independent states of the former Soviet Union, under contract no. 94-4040. The partners of the Inreca+ project are AcknoSoft (prime contractor, France), University of Kaiserslautern (Germany), Institute of Mathematics (Moldova), Reliable Software Inc. (Belarus), All-Russian Institute for Scientific and Technical Information (Russia), and Russian Academy of

Sciences (Russia). From the standpoint of developing a methodology for CBR applications partial funding has also been provided by the Stiftung Rheinland-Pfalz für Innovation (WiMo: „Knowledge Acquisition and Modeling for Case-Based Learning“).

8. References

[1] A. Aamodt and E. Plaza, Relating Case-Based Reasoning: Foundational Issues, Methodological Variations and System Approaches, AI Communications 7, 1 (1994) 39-59.

[2] K.-D. Althoff, Evaluating Case-Based Reasoning Systems: The Inreca Case Study, (Habilitationsschrift, University of Kaiserslautern, 1996).

[3] K.-D. Althoff and A. Aamodt, Relating Case-Based Problem Solving and Learning Methods to Task and Domain Characteristics: Towards an Analytic Framework, AI Communications 9 (1996) 1-8.

[4] K.-D. Althoff, E. Auriol, R. Barletta and M. Manago, A Review of Industrial Case-Based Reasoning Tools (AI Intelligence, Oxford, 1995).

[5] K.-D. Althoff and B. Bartsch-Spörl, Decision Support for Case-Based Applications, Wirtschaftsinformatik 38 (1996) 8-16.

[6] K.-D. Althoff, R. Bergmann, F. Maurer, M.M. Richter, R. Traphöner and W. Wilke, Wissensmodellierung und -akquisition für Case-Based Learning, Project proposal, Stiftung Rheinland-Pfalz für Innovation.

[7] K.-D. Althoff, M.M. Richter and W. Wilke, Case-Based Reasoning: A New Technology for Experience Based Construction of Knowledge Systems, forthcoming.

[8] K.-D. Althoff and S. Wess, Case-Based Knowledge Acquisition, Learning and Problem Solving in Diagnostic Real World Tasks, in: M. Linster and B. Gaines, eds., Proc. of the fifth European Knowledge Acquisition for Knowledge-Based Systems Workshop: EKAW-91 (GMD-Studien No. 211, Sankt Augustin, 1992) 48-67.

[9] K.-D. Althoff, S. Wess, R. Bergmann, F. Maurer, M. Manago, E. Auriol, N. Conruyt, R. Traphöner, M. Bräuer and S. Dittrich, Induction and Case-Based Reasoning for Classification Tasks, H.H. Bock, W. Lenski and M.M. Richter, eds., Information Systems and Data Analysis, Prospects-Foundations-Applications (Springer Verlag, Heidelberg, 1994) 3-16.

[10] K.-D. Althoff, S. Wess, K.-H. Weis jun., E. Auriol, R. Bergmann, H. Holz, R. Johnston, M. Manago, A. Meissonnier, C. Priebisch, R. Traphöner and W. Wilke, An Evaluation of the Final Integrated Inreca System, Inreca Esprit Project No. 6322, Deliverable D6, University of Kaiserslautern, Oct. 1995.

[11] B. Bartsch-Spörl, How to Make CBR Systems Work in Practice, in: H.D. Burkhard and M. Lenz, eds., 4th German Workshop on Case-Based Reasoning – System Development and Evaluation – (Informatik-Bericht No. 55, Humboldt University Berlin, 1996) 36-42.

[12] B.W. Boehm, A Spiral Model of Software Development and Enhancement, Computer 21, 5 (1988) 61-72.

[13] A.A. Bolotov and O.I. Larichev, Comparison of Pattern Recognition Methods by Precision of Approximating the Separating Hyperplanes, Automation and Remote Control 56 (1995), 1004-1010.

[14] S.J. Darmoni, P. Massari, J.-M. Droy, T. Blanc and J. Leroy, Functional Evalutation of SETH: An Expert System in Clinical Toxicology, in: P. Barahona, M. Stefanelli and J. Wyatt, eds., AI in Medicine – Proc. AIME '95 (Springer Verlag, Heidelberg, 1995) 231-238.

[15] J.H. Friedman, J.L. Bentley and R.A. Finkel, An Algorithm for Finding Best Matches in Logarithmic Expected Time, ACM Trans. Math. Software 3 (1977) 209-226.

[16] U. Gappa, F. Puppe and S. Schewe, Graphical Knowledge Acquisition for Medical Diagnostic Expert Systems, Artificial Intelligence in Medicine 5 (1993) 185-211.

[17] L. Gierl and S. Stengel-Rutkowski, Integrating Consultation and Semiautomatic Knowledge Acquisition in a Prototype-Based Architecture: Experiences with Dysmorphic Syndromes, Artificial Intelligence in Medicine 6 (1994) 29-49.

[18] K. Goos and S. Schewe, Case-Based Reasoning in Clinical Evaluation, in: S. Andreassen, R. Engelbrecht and J. Wyatt, eds., AI in Medicine – Proc. AIME '93 (IOS Press, Amsterdam, 1993) 445-448.

[19] M. Haddad, D. Mörtl and G. Porenta, SCINA: A Case-Based Reasoning System for the Interpretation of Myocardial Perfusion Scintigrams, in: Proc. of Computers in Cardiology (1995).

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

[21] L.H. Koopmans, Introduction to Contemporary Statistical Methods (Duxbury, Boston, Second Edition, 1987).

[22] O.I. Larichev, A Study on the Internal Organization of Expert Knowledge, Pattern Recognition and Image Analysis 5 (1995), 57-63.

[23] O.I. Larichev, H. Moshkovich, E. Furems, A. Mechitov and V. Morgoev, Knowledge Acquisition for the Construction of Full and Contradiction Free Knowledge Bases (Programma, Groningen, 1991).

[24] R.T. Macura and K.J. Macura, MacRad: Radiology image resource with a case-based retrieval system, in: M. Veloso and A. Aamodt, eds., Case-Based Reasoning Research and Development (Springer Verlag, Heidelberg, 1995) 43-54.

[25] M. Malek, V. Danel and V. Rialle, A Hybrid Case-Based Reasoning System Applied to Toxic Comas Diagnosis, Technical Report (1996).

[26] M. Malek and V. Rialle, A Case-Based Reasoning System Applied to Neuropathy Diagnosis, in: M. Keane, J.P. Haton and M. Manago, eds., EWCBR-94 – Second European Workshop on Case-Based Reasoning (AcknoSoft Press, Paris, 1994) 329-336.

[27] M. Manago, K.-D. Althoff, E. Auriol, R. Bergmann, S. Breen, M.M. Richter, J. Stehr, R. Traphöner and W. Wilke, INRECA II: Information and Knowledge Reengineering for Reasoning from Cases, Project proposal, Esprit IV.

[28] M. Manago and E. Auriol, Integrating Induction and Case-Based Reasoning for Troubleshooting CFM-56 Aircraft Engines, in: B. Bartsch-Spörl, D. Janetzko and S. Wess, eds., Fallbasiertes Schließen – Grundlagen und Anwendungen (Centre for Learning Systems and Application, University of Kaiserslautern, LSA-Report 95-02, 1995) 73-80.

[29] R.S. Michalski, A Theory and Methodology of Inductive Learning, Artificial Intelligence 20, 2 (1983) 111-161.

[30] E.T.O. Opiyo, Case-Based Reasoning for expertise relocation in support or rural health workers in development countries, in: M. Veloso and A. Aamodt, eds., Case-Based Reasoning Research and Development (Springer Verlag, Heidelberg, 1995) 77-87.

[31] B. Puppe, Building a Medical Knowledge Base: Tricks Facilitating the Simulation of the Expert's Reasoning, in: S. Andreassen, R. Engelbrecht and J. Wyatt, eds., AI in Medicine – Proc. AIME '93 (IOS Press, Amsterdam, 1993) 168-171.

[32] B. Puppe, C. Ohmann, K. Goos, F. Puppe and O. Mootz, Evaluating 4 Diagnostic Methods with Acute Abdominal Pain Cases, Technical Report (University of Würzburg, 1994).

[33] J.R. Quinlan, Induction of Decision Trees, Machine Learning 1 (1986) 81-106.

[34] J.R. Quinlan, C4.5: Programs for machine learning, (San Mateo, Morgan-Kaufmann, 1993).

[35] M.M. Richter and S. Wess, Similarity, Uncertainty and Case-Based Reasoning in Patdex, in: R.S. Boyer, ed., Automated Reasoning (Kluwer Academic Publishers, 1991) 249-265.

[36] R. Schmidt, B. Heindl, B. Pollwein and L. Gierl, Prognoses of Multiparametric Time Course Abstractions in a Case-Based Reasoning System, in: H.D. Burkhard and M. Lenz, eds., 4th German Workshop on Case-Based Reasoning – System Development and Evaluation – (Informatik-Bericht No. 55, Humboldt University Berlin, 1996) 170-177.

[37] R. Schmidt, B. Pollwein, L. Boscher, G. Schmid and L. Gierl, Der fallbasierte Konsiliarius ICONS für die Antibiotika-Therapie, in: B. Bartsch-Spörl, D. Janetzko and S. Wess, eds., 3rd German Workshop on Case-Based Reasoning – Foundations and Applications – (Centre for Learning Syst ems and Applications, University of Kaiserslautern, LSA-95-02,1995) 54-62.

[38] A. Seitz and A. Uhrmacher, Cases versus Model-Based Knowledge – An Application in the Area of Bone Healing, in: H.D. Burkhard and M. Lenz, eds., 4th German Workshop on Case-Based Reasoning – System Development and Evaluation – (Informatik-Bericht No. 55, Humboldt University Berlin, 1996) 178-185.

[39] Shannon and Weaver, The Mathematical Theory of Computation, (University of Illinois Press, 1947).

[40] J. Van Bemmel, Criteria for the Acceptance of Decision Support Systems by Clinicians, in: S. Andreassen, R. Engelbrecht and J. Wyatt, eds., AI in Medicine – Proc. AIME '93 (IOS Press, Amsterdam, 1993) 7-12.

[41] S. Wess, Fallbasiertes Schließen in wissensbasierten Systemen zur Entscheidungsunterstützung und Diagnostik, (Doctoral Dissertation, University of Kaiserslautern, 1995; also: Infix Verlag, Sankt Augustin, Germany).

[42] S. Wess, K.-D. Althoff and G. Derwand, Using k-d Trees to Improve the Retrieval Step in Case-Based Reasoning. in: S. Wess, K.-D. Althoff and M.M. Richter, eds., Topics in Case-Based Reasoning (Springer Verlag, Heidelberg, 1994) 167-181.

[43] Y.I. Zhuravlev, On an algebraic approach to the problems of pattern recognition and classification, Probl. Kibern. 33 (1978) 5-58 [in Russian].