# Case-Based Reasoning in the Health Sciences

Workshop at the
Twenty-Third International Conference on
Case-Based Reasoning
(ICCBR 2015)

Frankfurt, Germany
September 2015

Isabelle Bichindaritz, Cindy Marling, and Stefania Montani
(Editors)

## Program Chairs

| | |
|---|---|
| Isabelle Bichindaritz | State University of New York at Oswego, USA |
| Cindy Marling | Ohio University, USA |
| Stefania Montani | University of Piemonte Orientale, Italy |

## Program Committee

| | |
|---|---|
| Agnar Aamodt | NTNU, Norway |
| Juan Manuel Cortado | University of Salamanca, Spain |
| Peter Funk | Mälardalen University, Sweden |
| Jean Lieber | Loria, University of Nancy, France |
| Amedeo Napoli | Loria, University of Nancy, France |
| Lucia Sacchi | University of Pavia, Italy |
| Rainer Schmidt | Institute for Medical Informatics and Biometry, University of Rostock, Germany |

## Preface

The community working on health sciences applications of case-based reasoning (CBR) meets again at the International Conference on Case-based Reasoning (ICCBR) this year to share ideas and system descriptions collected in the proceedings of this workshop. This event is the tenth in a series of successful workshops, co- located with different ICCBR/ECCBR conferences. The first nine were held at ICCBR-03, in Trondheim, Norway, at ECCBR-04, in Madrid, Spain, at ICCBR-05, in Chicago, USA, at ECCBR-06 in Olüdeniz, Turkey, at ICCBR-07 in Belfast, Northern Ireland , at ECCBR-08 in Trier, Germany, at ICCBR-09 in Seattle, USA, at ICCBR- 2012 in Lyon, France, and at ICCBR-2013 in Saratoga Springs, USA.

Three papers and one invited speaker summary have been selected this year for presentation during the ICCBR workshops and inclusion in the Workshops Proceedings. The first paper in these proceedings deals with medical process management and presents a tool capable of predicting the evolution of a current process based on previous traces, which can be very useful to ensure compliance with clinical guidelines [Bottrighi et al.]. The second paper highlights how a case-based approach to data analysis can aid in integrating new fitness band data into machine learning models for blood glucose prediction [Marling et al.]. The third paper focuses on the importance and variety of data mining methods in case-based reasoning, in particular for health sciences applications [Bichindaritz]. Finally, the invited speaker summary discusses how hybrid case-based reasoning is likely to change the future of health science and healthcare [Funk].

These papers report on the research and experience of 11 authors working in three different countries on a wide range of problems and projects, and illustrate some of the major trends of current research in the area. Overall, they represent an excellent sample of the most recent advances of CBR in the health sciences, and promise very interesting discussions and interaction among the major contributors in this niche of CBR research.

September 2015
Frankfurt

Isabelle Bichindaritz
Cindy Marling
Stefania Montani

# Trace Retrieval as a Tool for Operational Support in Medical Process Management

Alessio Bottrighi (1), Luca Canensi (2), Giorgio Leonardi (1), Stefania Montani (1), Paolo Terenziani (1)

(1) DISIT, Computer Science Institute, Università del Piemonte Orientale, Alessandria, Italy
(2) Department of Computer Science, Università di Torino, Italy

**Abstract.** Operational support assists users while process instances are being executed, by making predictions about the instance completion, or recommending suitable actions, resources or routing decisions, on the basis of the already completed process traces. Operational support can be particularly useful is the case of medical processes, where a given process instance execution may differ from the indications of the existing reference clinical guideline. In this paper, we propose a Case Based Reasoning approach to medical process management operational support. The framework enables the user to retrieve past traces by submitting queries representing complex patterns exhibited by the current process instance. Information extracted from the retrieved traces can guide the medical expert in managing the current instance in real time. The tool relies on a tree structure, allowing fast retrieval from the available event log. Thanks to its characteristics and methodological solutions, the tool implements operational support tasks in a flexible, efficient and user friendly way.

## 1 Introduction

Operational support is a process management activity meant to assist users while process instances are being executed, by making predictions about the instance completion, or recommending suitable actions, resources or routing decisions, on the basis of the already completed instances [1]. Operational support can be particularly useful in the case of medical processes, where a given process instance execution may (significantly) differ from the indications of the existing reference clinical guideline. Indeed, specific patient characteristics (e.g., co-morbidities, allergies, etc.), or local resource constraints, may lead to deviations from the default behavior, which need to be managed in real time. Prediction and recommendation heavily rely on experiential knowledge, stored in the so-called "event log" in the form of past process traces. Case Based Reasoning (CBR) [2], and specifically the retrieval step in the CBR cycle, thus appears to be a very valuable methodology for implementing these operational support tasks. The percentage of retrieved traces that, e.g., were completed on time, can then be used to calculate the probability that the current instance will complete on time too. A

similar approach can be adopted to estimate costs, or predict problems. Moreover, the best actions to execute next can also be extracted from the retrieved traces.

In this paper we propose a case-based retrieval framework, where cases are traces of process execution, aimed at enabling prediction and recommendation in medical process operational support. In our framework, queries can be composed of several simple patterns (i.e., single actions, or direct sequences of actions), separated by delays (i.e., interleaved actions we do not care about). Delays can also be imprecise (i.e., the number of interleaved actions can be given as a range). To the best of our knowledge, an operational support facility like this is not available in the tools described in the literature. Our framework relies on a tree structure, called the *trace tree*, allowing fast retrieval, thus avoiding a flat search for all the traces in the log that match the input pattern. The trace tree is a sort of "model" of the traces, that we learn using a process mining technique we recently implemented [3], and built in such way that it can be used as an index[1].

The paper is organized as follows. In section 2 we illustrate our retrieval approach. In section 3 we discuss related work. In section 4 we present our concluding remarks and future work directions.

## 2 Trace retrieval

In our framework, trace retrieval relies on a tree structure, called the **trace tree**, in order to avoid a flat search for all the traces in the log that match the input query. In the following, we will first describe the trace tree semantics, and then introduce our query language and, finally, our retrieval procedure.

**Trace tree semantics.** In the trace tree, nodes represent actions, and arcs represent a precedence relation between them. More precisely, each node is represented as a pair $< P, T >$.

$P$ denotes a (possibly unary) set of actions; actions in the same node are in $AND$ relation, or, more properly, may occur in any order with respect to each other. Note that, in such a way, each path from the starting node of the tree to a given node $N$ denotes a set of possible process patterns (called *support patterns* of $N$ henceforth), obtained by following the order represented by the arcs in the path to visit the trace tree, and ordering in each possible way the actions in each node (for instance, the path $\{A, B\} \rightarrow \{C\}$ represents the support patterns "ABC" and "BAC").

$T$ represents a set of pointers to all and only those traces in the log whose prefixes exactly match one of the patterns in $P$ (called *support traces* henceforth). Specifically, prefixes correspond to the entire traces if the node at hand is a leaf. In the case of a node representing a set of actions to be executed in any order, $T$ is

[1] While the motivations for defining such a novel mining algorithm, and its advantages with respect to existing process mining literature contributions (e.g., ProM [4]), are extensively discussed elsewhere [3], in this work we focus on its usage to support case retrieval.

more precisely composed of several sets of support traces, each one corresponding to a possible action permutation.

Since all traces start with a dummy common action #, the root node contains the action #, paired to the pointers to all log traces.

**Query language.** In a tool implementing this framework, the user can issue a query, composed of one or more simple patterns to be searched for. In turn, simple patterns are defined as one or more actions in direct sequence. Multiple simple patterns can be combined in a complex pattern, by separating them by delays. A delay is a sequence of actions interleaved between two simple patterns; the semantics is that we do not care about these actions, so they will not be specified in the query. Instead, only their number will be provided, possibly in an imprecise way (i.e., we allow the user to express the number of interleaved actions as a range).

Formally, a query is represented in the following format:

$\langle (min_1, max_1)SP_1(min_2, max_2)SP_2...(min_k, max_k)SP_k(min_{k+1}, max_{k+1})\rangle$

where:

- $SP_j$ is a simple pattern (i.e. a sequence of letters, representing the actions we are looking for; these actions have to be in direct sequence);
- $(min_j, max_j)$ is the delay between two items (i.e., two simple patterns, or a simple pattern and the trace starting/ending point), expressed as a range in the number of interleaved actions.

As an example, the query
$\langle (0,0)B(0,1)E(2,2)Z(0,1)\rangle$
looks for action $B$, which has to start at the very beginning of the trace (just after the start action # - all traces start with a dummy common action # in our approach). This first simple pattern $B$ must be followed (with zero or a single interleaved action in between) by action $E$. $E$ must be followed by two actions, which we do not care about; after them, $Z$ is required. $Z$ can be the final action, or can be followed by one additional action we do not care about.

For instance, in the stroke management domain, where we will test our approach, actions $B$, $E$ and $Z$ could correspond to "Arrival at the emergency department", "Neurological examination", and "Chest X-ray" respectively. Looking for "Arrival at the emergency department" at the very beginning of the trace allows the exclusion of all those patients that were first stabilized at home or in the ambulance. The query then aims for searching for those situations where "Neurological examination" is executed early, and before "Chest X-ray"; in fact, this specific ordering is not mandatory, because "Chest X-ray" is a procedure common to many different disease management processes, and may be executed at different times, also depending on the availability of the X-ray machine. Similarly, in some cases "Neurological examination" might be delayed, if the neurologist is not available. The two actions between "Neurological examination" and "Chest X-ray" would typically correspond to "CTA" and "ECG", always obtained to every patient in the case of a suspected stroke (but not explicitly queried in the example).

It is worth noting that a query written as above corresponds to a whole set of queries, each one obtained by choosing a specific delay value and specific actions in each of the $(min_j, max_j)$ intervals. Every query in this set can be made partially explicit as a string, containing as many dummy symbols $*$ as needed, to cover the corresponding delay length (where the dummy symbol is chosen because we are not interested in the specific interleaved actions). For example, the query above would correspond to the following four partially explicit queries, whose length ranges from 6 to 8 actions (including #), where the dummy symbol $*$ has been properly inserted, according to the delay values information: $\#BE**Z$; $\#BE**Z*$; $\#B*E**Z$; $\#B*E**Z*$

Since each $*$ could be substituted by any of the $N$ types of actions recorded in the log and/or existing in the application domain, the example query corresponds to $N^2 + 2 * N^3 + N^4$ totally explicit queries.

The problem is obviously combinatorial, with respect to the possible delay ranges and action types. We thus believe that extensional approaches (in which only explicit queries can be issued) would not be feasible in many domains. Our query language, allowing for compact "intensional" queries, is therefore a significant move in the direction of implementing an efficient and user-friendly operational support tool.

**Trace retrieval.** In order to retrieve the log traces that match a query, we have implemented a multi-step procedure, articulated as follows: (1) automaton generation; (2) tree search; (3) filtering.

To generate the automaton, in turn, we implement the following procedure:

1. transform the query into a regular expression;
2. apply the Berry and Sethi [5] algorithm, to build a non-deterministic automaton that recognizes the regular expression above;
3. unfold the non-deterministic automaton;
4. transform the unfolded non-deterministic automaton into a deterministic automaton [6].

Steps (1) and (4) are trivial. As regards step (1) note that our query language is just a variation of regular expressions, useful to express delays and "do not care" (i.e., dummy) symbols in a compact way. The cost of step (1) is linear in the number of delays used in the query. Steps (2) and (3) use classical algorithms in the area of formal languages. The cost of step (2) is linear in the number of symbols in the query expressed as a regular expression (i.e., the output of step (1) [5]), and the cost of step (3) is the product between the number of dummy symbols in the query and the cardinality of the action symbols available in the log. Step (4) substitutes each arc labeled by the dummy symbol in the automaton with a set of arcs, one for each action in the event log. Although in the worst case step (4) is exponential with respect to the number of states in the automaton (i.e., the output of step (2)), note that the worst case is rare in practice [7].

Once the deterministic automaton has been obtained, it would be possible to exploit it in a classical way, by providing all event log traces in input to it, to verify which of them match the query. However, some of these traces may be identical, or share common prefixes of various length, so that the straightforward

approach would lead to repeated analyses of the common parts. In order to optimize efficiency, we have therefore proposed a novel approach, that provides the trace tree as an input to the automaton. Each path in the trace tree may index several identical support traces, that will be considered only once, thus speeding up retrieval with respect to a flat search into the event log. Moreover, in the tree common prefixes of different traces are represented just once, as common branches close to the root (different postfixes can then stem from the common branches, to reach the various leaves). These common parts will be executed on the automaton only once, without requiring repeated, identical checks.

It is worth noting that providing a tree as an input to the automaton represents a significantly novel contribution, since in the formal languages literature the input to be executed on the automaton is typically a string. The work in [8] represents an exception, but the tree it exploits (a Patricia tree) has very different semantics with respect to ours.

In detail, our approach operates as follows: the algorithm *Search_Process* (see algorithm 1) takes in input the trace tree $T$ and the deterministic automaton $A$, and provides as an output a set of pairs, composed of a trace tree leaf node and a corresponding string. Notably, there could be several pairs having the same leaf node. Each of the strings is an explicit instantiation of the query represented by the automaton, verified by (some of) the support traces in the leaf node. The output support traces are then provided as an input to the filtering step (see below).

Basically, *Search_Process* executes a breadth first visit of the trace tree; it exploits the variable *searching*, defined as a set of triples, composed of a trace tree node, an automaton state, and the string that has been recognized on the automaton so far. Initially (line 4), *searching* contains the root (with the dummy action #), paired to the initial state of the query automaton and to the empty string. The visit procedure (lines 7-35) extracts one triple at a time from the set *searching*. If the *node* in the triple contains a set of actions to be executed in any order (line 9), we simulate all the permutations on the automaton, and save the states we reach and the corresponding recognized strings into *new_states* set (line 12). If the node contains one single action, we simply simulate it on the automaton, and save the state we reach and the corresponding string into *new_states* set (line 17). In both cases, the string saved in *new_states* is the one in the input triple properly updated with the newly recognized symbols.

After the simulation, if the *node* at hand is a leaf (line 20), then for each item in *new_states* we check whether the state component is a final state (lines 22-24); if this is the case, *node* and the string paired to the final state are saved in the output variable *result* (line 23). Otherwise, if *node* is not a leaf, we pair its children to all the items in *new_states*, and save these objects into *searching* (lines 27-33). The visit terminates when *searching* is empty, i.e., all tree levels have been visited. The visit procedure is linear in the number of the trace tree nodes.

Referring to our example query, providing the trace tree in figure 1 as an input to the algorithm *Search_Process*, after examining the root (which is triv-

**ALGORITHM 1:** Pseudo-code of the procedure *Search_Process*.

```
1  Search_Process(T, A)
2  Output: set of < node, string >
3  result ← {}
4  searching ←< root(T), 0, empty >
5  repeat
6  |  tmp ← {}
7  |  foreach < node, state, string > ∈ searching do
8  |  |  new_states ← {}
9  |  |  if node is an any-order-node then
10 |  |  |  foreach Perm ∈ permutation(node) do
11 |  |  |  |  foreach act ∈ Perm do
12 |  |  |  |  |  new_states ← new_states ∪ simulate(A, act,state,string)
13 |  |  |  |  end
14 |  |  |  end
15 |  |  end
16 |  |  else
17 |  |  |  new_states ← simulate(A, action(node),state,string)
18 |  |  end
19 |  |  if new_states ≠ {} then
20 |  |  |  if node is a leaf then
21 |  |  |  |  foreach < state, string > ∈ new_states do
22 |  |  |  |  |  if final(state) then
23 |  |  |  |  |  |  result ← result ∪ < node, string >
24 |  |  |  |  |  end
25 |  |  |  |  end
26 |  |  |  end
27 |  |  |  else
28 |  |  |  |  foreach n ∈ sons(node) do
29 |  |  |  |  |  foreach < state, string > ∈ new_states do
30 |  |  |  |  |  |  tmp ← tmp ∪ < n, state, string >
31 |  |  |  |  |  end
32 |  |  |  |  end
33 |  |  |  end
34 |  |  end
35 |  end
36 |  searching ← tmp
37 until searching ≠ {}
38 return result
```

ial), *searching* contains the children of the root $A$, $B$, and $C$, paired with the state of the deterministic automaton, and with the string #. We simulate the actions $A$, $B$, and $C$ on the automaton. Only $B$ (i.e., "Arrival at the emergency department") is recognized, generating a state saved in *new_states* with the corresponding string $\#B$ (line 17). We then pair the children of node $B$ ($E$, $D$, $D-E$) to the item in *new_states* and save these triples into *searching* (lines 27-33). In the stroke management domain, $E$ corresponds to "Neurological examination" and $D$ to "CTA". Continuing the visit, particularly interesting is the case of node $D-E$, which requires consideration of all the possible permutations of actions $D$ and $E$. Both the permutations $DE$ and $ED$ are initially recognized. However, as the visit proceeds and node $P-Z$ is reached (with $P$ corresponding to "ECG" and $Z$ corresponding to "Chest X-ray"), it turns out that $DE$ must be followed by the permutation $PZ$ to match the query; on the other hand, if the choice $ED$ is made, it must be followed by $ZP$. Indeed, the query imposes some *constraints that cannot be checked only locally*, i.e., referring to a single node along the branch. After this step of the visit (depth 5 in the tree), the recognized partial strings paired to node $P-Z$ are $\#BDEOPZ$ and $\#BEDOZP$ (with $O$ corresponding to "NMR"). Notably, the patterns $\#BDEOZP$ and $\#BEDOPZ$ do not match the input query.

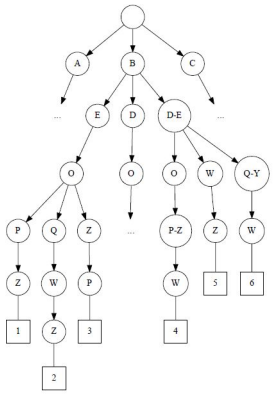


**Fig. 1.** Trace tree in the example.

If an output leaf node ends a branch which includes one or more nodes with actions to be executed in any order, it is possible that only some of the permutations of these actions are acceptable to answer the query. However, the trace tree leaf node indexes all the traces corresponding to the various support patterns (i.e., considering all possible permutations). Therefore, the support traces must be filtered.

To do so, without the need of operating directly on the input traces, we exploit the fact that, in each node with actions to be executed in any order, every permutation is explicitly stored, and each permutation indexes all and

only the support traces corresponding to it. Thus, the basic idea of our filtering step is simple: for each output pair ⟨ Node, String ⟩ of the tree search step, we navigate the trace tree from *Node* back to the root, maintaining, in each any order node, only the (pointers to) the traces corresponding to *String* (this can be easily done through the operation of intersection between sets of pointers).

The complexity of the filtering step is superiorly limited by the number of ⟨ Node, String ⟩ pairs identified as an output of the tree search step, multiplied by the tree depth.

Obviously, if the leaf node ends a branch that contains no nodes with actions to be executed in any order, the leaf support traces can be directly presented to the user, and the filtering step is not required.

## 3 Related work

Operational support techniques are implemented in the open source framework ProM [4] (developed at the Eindhoven University of Technology), which represents the state of the art in process mining research. In ProM, prediction and recommendation are typically supported by replaying log traces on the transition system [9], a state-based model that explicitly shows the states a process can be in, and all possible transitions between these states. The replay activity allows calculation of, e.g., the mean time to completion from a given state, or the most probable next action to be executed. In ProM's approach, statistics on event log traces are thus used for operational support, but the overall technique is very different from the one we propose in this paper, and no trace retrieval on the basis of complex pattern search is supported.

On the other hand, traces have been recently considered in the CBR literature, as sources for retrieving and reusing user's experience. As an example, at the International Conference on CBR in 2012, a specific workshop was devoted to this topic [10]. In 2013, Cordier et al. [11] proposed trace-based reasoning, a CBR approach where cases are not explicitly stored in a library, but are implicitly recorded as "episodes" within traces. The elaboration step, in which a case is extracted from a trace, is thus one of the most challenging parts of the reasoning process. Zarka et al. [12] extended that work, and defined a similarity measure to compare episodes extracted from traces. In these works, traces are typically intended as observations captured from users' interaction with a computer system. Trace-based reasoning was exploited in recommender systems [13, 14], and to support the annotation of digitalized cultural heritage documents [15]. Leake used execution traces recording provenance information to improve reasoning and explanation in CBR [16]. In the Phala system [17], the authors supported the generation and composition of scientific workflows by mining execution traces for recommendations to aid workflow authors. Finally, Lanz et al. used annotated traces recorded when a human user played video games in order to feed a case-based planner [18].

All these approaches implement forms of reasoning on traces. However, to the best of our knowledge, a tace-based CBR approach has never been exploited for operational support in Medical Process Management.

## 4 Conclusions

In this paper, we have introduced a novel framework for trace retrieval, designed to implement operational support tasks in a flexible, efficient and user-friendly way. With respect to existing operational support facilities, our tool is more flexible because it allows to search for traces that exhibit complex query patterns, identified in the input trace. The tool is also efficient and user-friendly, since:

- by allowing for the use of (imprecise) delays in the query language, it enables users to express a very large number of explicit queries in a compact way;
- by providing the trace tree as an input to the automaton:
  - it speeds up retrieval relative to a flat search into the event log;
  - it executes common prefixes of different traces only once on the automaton, avoiding repeated, identical checks.

In the future, we plan to extensively test the overall framework on real world traces, which log the actions executed during stroke patient management in a set of Northern Italy hospitals.

## References


1. W. Van der Aalst. *Process Mining. Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011.
2. A. Aamodt and E. Plaza. Case-based reasoning: foundational issues, methodological variations and systems approaches. *AI Communications*, 7:39–59, 1994.
3. L. Canensi, S. Montani, G. Leonardi, and P. Terenziani. Chapman: a context aware process miner. In *Workshop on Synergies between Case-Based Reasoning and Data Mining, International Conference on Case Based Reasoning (ICCBR)*, 2014.
4. B. van Dongen, A. Alves De Medeiros, H. Verbeek, A. Weijters, and W. Van der Aalst. The proM framework: a new era in process mining tool support. In G. Ciardo and P. Darondeau, editors, *Knowledge Mangement and its Integrative Elements*, pages 444–454. Springer, Berlin, 2005.
5. G. Berry and R. Sethi. From regular expressions to deterministic automata. *Theor. Comput. Sci.*, 48(3):117–126, 1986.
6. M. S. Lam, A. V. Aho, R. Sethi, and J. D. Ullman. *Compilers: Principles, Techniques, and Tools*. Addison-Wesley, 2006.
7. J. van Leeuwen. *Handbook of Theoretical Computer Science*. MIT Press, 1994.
8. Ricardo A. Baeza-Yates and Gaston H. Gonnet. Fast text searching for regular expressions or automaton searching on tries. *J. ACM*, 43(6):915–936, November 1996.
9. B. F. Van Dongen, N. Busi, and G. M. Pinna. An iterative algorithm for applying the theory of regions in process mining.

10. M. W. Floyd, B. Fuchs, P. Gonzalez-Calero, D. Leake, S. Ontanon, E. Plaza, and J. Rubin. *TRUE: Traces for Reusing User's Experiences Cases, Episodes, and Stories, International Conference on Case Based Reasoning (ICCBR)*. Lyon, 2012.
11. Amlie Cordier, Marie Lefevre, Pierre-Antoine Champin, Olivier Georgeon, and Alain Mille. Trace-Based Reasoning — Modeling interaction traces for reasoning on experiences. In *The 26th International FLAIRS Conference*, May 2013.
12. Raafat Zarka, Amlie Cordier, Elod Egyed-Zsigmond, Luc Lamontagne, and Alain Mille. Similarity Measures to Compare Episodes in Modeled Traces. In Springer, editor, *International Case-Based Reasoning Conference (ICCBR 2013)*, Lecture Notes in Computer Science, pages 358–372. Springer Berlin Heidelberg, July 2013.
13. Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, pages 335–336, New York, NY, USA, 2008. ACM.
14. Raafat Zarka, Amélie Cordier, Elöd Egyed-Zsigmond, and Alain Mille. Contextual trace-based video recommendations. In *Proceedings of the 21st International Conference Companion on World Wide Web*, WWW '12 Companion, pages 751–754, New York, NY, USA, 2012. ACM.
15. Reim Doumat, Elöd Egyed-Zsigmond, and Jean-Marie Pinon. User trace-based recommendation system for a digital archive. In *Proceedings of the 18th International Conference on Case-Based Reasoning Research and Development*, ICCBR'10, pages 360–374, Berlin, Heidelberg, 2010. Springer-Verlag.
16. David B. Leake. Case-based reasoning tomorrow: Provenance, the web, and cases in the future of intelligent information processing. In *Intelligent Information Processing V - 6th IFIP TC 12 International Conference, IIP 2010, Manchester, UK, October 13-16, 2010. Proceedings*, page 1, 2010.
17. D. B. Leake and J. Kendall-Morwick. Towards case-based support for e-science workflow generation by mining provenance. In K.D. Althoff, R. Bergmann, M. Minor, and A. Hanft, editors, *Proc. ECCBR 2008, Advances in Case-Based Reasoning*, volume 5239 of *Lecture Notes in Computer Science*, pages 269–283. Springer, 2008.
18. A. Lanz, B. Weber, and M. Reichert. Workflow time patterns for process-aware information systems. In *Proc. BMMDS/EMMSAD*, pages 94–107, 2010.

# Case-Based Reasoning as a Prelude to Big Data Analysis: A Case Study

Cindy Marling[1], Razvan Bunescu[1],
Babak Baradar-Bokaie[2] and Frank Schwartz[2]

[1] School of Electrical Engineering and Computer Science
Russ College of Engineering and Technology
Ohio University, Athens, Ohio 45701, USA
marling@ohio.edu, bunescu@ohio.edu

[2] Department of Specialty Medicine
Heritage College of Osteopathic Medicine
Ohio University, Athens, Ohio 45701, USA
babak.bokaie@gmail.com, schwartf@ohio.edu

**Abstract.** The 4 Diabetes Support System (4DSS) is a prototypical hybrid case-based reasoning (CBR) system that aims to help patients with type 1 diabetes on insulin pump therapy achieve and maintain good blood glucose control. The CBR cycle revolves around treating blood glucose control problems by retrieving and reusing therapeutic adjustments that have been effectively used to treat similar problems in the past. Other artificial intelligence (AI) approaches have been integrated primarily to aid in situation assessment: knowing when a patient has a blood glucose control problem and characterizing the type of problem that the patient has. Over the course of ten years, emphasis has shifted toward situation assessment and machine learning approaches for predicting blood glucose levels, as that is the area of greatest patient need. The goal has been to make large volumes of raw insulin, blood glucose and life-event data actionable. During the past year, newly available fitness bands have provided a potentially valuable source of additional data for controlling diabetes. Because it was initially unclear whether or how this new data might be leveraged, a case study was conducted, and CBR was once again called into play. This paper describes the case study and discusses the potential of CBR to serve as a prelude to big data analysis.

## 1 Introduction

The World Health Organization estimates that there are 347 million people living with diabetes [11]. From 5 to 10% of them have type 1 diabetes (T1D), the most severe form, in which the pancreas fails to produce insulin. T1D is neither curable nor preventable; however, it can be treated with insulin and effectively managed through blood glucose (BG) control. Good BG control helps to delay or prevent long-term diabetic complications, including blindness, amputations,

kidney failure, strokes, and heart attacks [4]. Therefore, it is important to rapidly identify and correct BG control problems.

The 4 Diabetes Support System (4DSS) is a prototypical hybrid case-based reasoning (CBR) system that detects and predicts BG control problems and suggests personalized therapeutic adjustments to correct them. The 4DSS has been extensively described in the literature [6, 7, 9]; a brief system overview is presented in the next section. A critical research thrust that grew out of work on the 4DSS is how to continuously, in real-time, predict that a BG control problem is about to occur. The key is to be able to accurately predict what the BG level will be in the next 30 to 60 minutes, which would allow enough time to intervene and prevent predicted problems. Large volumes of raw insulin and BG data are available for analysis. Machine learning algorithms for time series prediction have been efficaciously applied. Studies conducted on retrospective data show that our system predicts BG levels comparably to physicians specializing in diabetes care, but not yet well enough for use by patients in the real world [2, 8].

Recently, commercially available fitness bands and smart watches, such as the Basis Peak, Nike Fuelband, Fitbit, and Apple Watch, have made it practical to inexpensively and unobtrusively collect large quantities of physiological data. As this data may be indicative of patient activity impacting BG levels, it could potentially be used to improve BG level prediction. However, due to the complicated nature of the problem, it was not initially clear whether or how this data could be leveraged. Therefore, a case study was conducted in which a patient with T1D wore a fitness band in addition to his usual medical devices for two months. The data was consolidated and displayed via custom visualization software. The patient, his physician, and artificial intelligence (AI) researchers met weekly to review and interpret the data, using a protocol like that employed to build the 4DSS. This case-based focus shed light on how the new data could be integrated into machine learning models and leveraged to improve BG prediction. We posit that a case-based approach is especially useful in dealing with new data sources, new patients, and new medical conditions, and that early lessons learned through the CBR process can aid in later big data analysis.

## 2 Background

This section briefly describes the 4DSS and the work on machine learning models for blood glucose prediction prior to the new case study.

### 2.1 The 4 Diabetes Support System

A graphical overview of the prototypical hybrid CBR system is shown in Figure 1. The patient provides BG, insulin and life-event data to the system. BG and insulin data are uploaded from the patient's prescribed medical devices. The patient enters data about life events that impact BG levels, such as food, exercise, sleep, work, stress and illness, using a smart phone. The data is scanned by

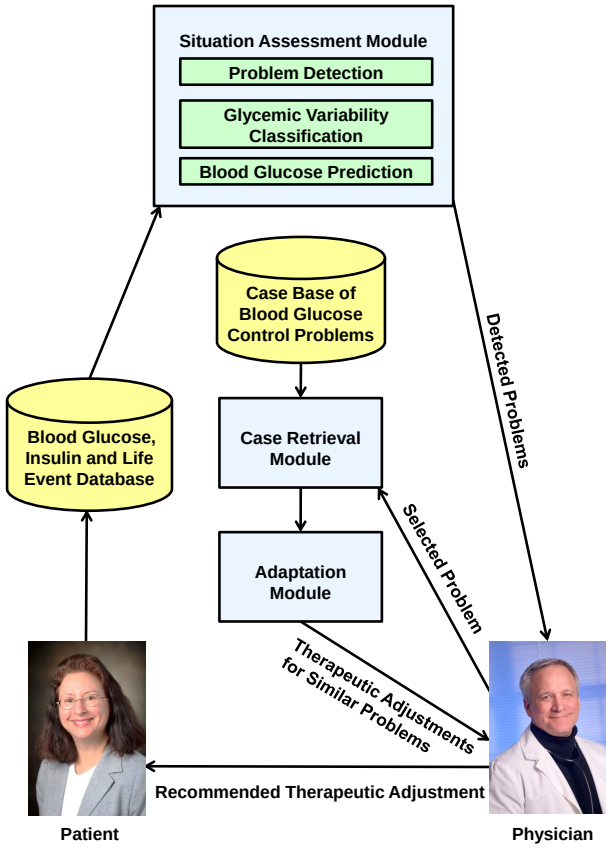


**Fig. 1.** Overview of the 4 Diabetes Support System, reproduced from [6]

the situation assessment module, which detects and predicts BG control problems. The most critical types of problems are: hyperglycemia, or high BG, which contributes to long-term diabetic complications; and hypoglycemia, or low BG, which may result in severe immediate reactions, including weakness, dizziness, seizure or coma. The situation assessment module reports detected problems to the physician. The physician selects a problem of interest, which is then used by the case retrieval module to obtain the most similar case or cases from the case base. Each retrieved case contains a specific BG control problem experienced by a T1D patient, a physician's recommended therapeutic adjustment, and the clinical outcome for the patient after making the therapeutic adjustment. Retrieved cases go to the adaptation module, which personalizes a retrieved solution to fit the specific needs of the current patient. A solution is a therapeutic adjustment comprising one or more actions that a patient can take. Adapted therapeutic adjustments are displayed to the physician as decision support. The physician

decides whether or not to recommend the therapeutic adjustments to the patient. Directly providing suggestions to the patient, while a long-term goal, would require regulatory approval.

### 2.2 Machine Learning Models for Blood Glucose Prediction

Originally conceived of as important for situation assessment, BG prediction has multiple potential applications that could enhance safety and quality of life for people with T1D if incorporated into medical devices. These applications include: alerts to warn of imminent problems; decision support for taking actions to prevent impending problems; "what if" analysis to project the effects of lifestyle choices on BG levels; and integration with closed-loop control algorithms for insulin pumps (aka the "artificial pancreas"). Predicting hypoglycemia is especially important, both for patient safety and because hypoglycemia is a limiting factor for intensive insulin therapy [3].

In our BG prediction approach, a generic physiological model is used to generate informative features for a Support Vector Regression (SVR) model that is trained on patient specific data. The physiological model characterizes the overall dynamics into three compartments: meal absorption, insulin dynamics, and glucose dynamics. The parameters of the physiological model are tuned to match published data and feedback from physicians. To account for the noise inherent in the data, the state transition equations underlying the continuous dynamic model are incorporated in an extended Kalman filter.

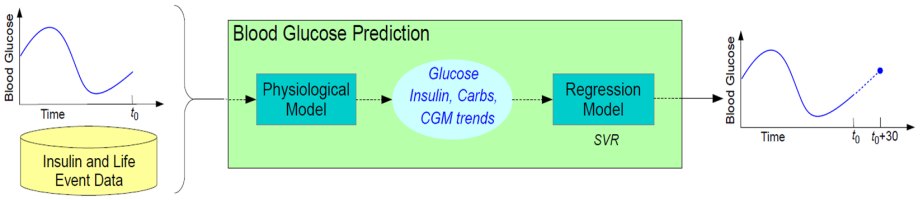


**Fig. 2.** Overview of the Blood Glucose Level Prediction Process

Figure 2 shows the overall BG level prediction process. A continuous dynamical system implementing the set of physiological equations is run in prediction mode for 30 and 60 minutes. Physiological model predictions are then used as features for an SVR model that is trained on the two weeks of data preceding the test point. Furthermore, an AutoRegressive Integrated Moving Average (ARIMA) model is trained on the same data and its predictions are used as additional features. The models are trained to minimize Root Mean Square Error (RMSE). SVR predictions are made at 30 and 60 minute intervals and compared to BG levels at prediction time ($t_0$), ARIMA predictions, and predictions made by physicians specializing in diabetes care. Results are shown in Figure 3.

| Horizon | $t_0$ | ARIMA | Physician | SVR |
|---|---|---|---|---|
| 30 Min | 27.5 | 22.9 | **19.8** | **19.5** |
| 60 Min | 43.8 | 42.2 | **38.4** | **35.7** |

**Fig. 3.** RMSE of the Best SVR Model vs. the $t_0$, ARIMA, and Physician Predictions

## 3 The Case Study

The recent proliferation of commercially available fitness bands provides an opportunity to exploit new data from inexpensive, unobtrusive, portable physiological sensors. These sensors provide signals indicative of activities that are known to impact BG levels, including sleep, exercise and stress. The hope is that, by incorporating these signals, we can obtain a more accurate picture of patient activity, while reducing or eliminating the need for the patient to self-report life events. We conducted an N-of-1 study in order to learn whether or how this influx of new data could be leveraged to advantage.

The subject was a middle-aged physician who has had T1D since childhood. For two months, he wore a fitness band along with his regularly prescribed medical devices and entered life-event data via his smart phone. The fitness band, a Basis Peak, provided data for galvanic skin response (GSR), heart rate, and skin and air temperatures. The medical devices, a Medtronic insulin pump and a Dexcom continuous glucose monitoring (CGM) system, provided insulin and BG data. All of this data was consolidated in the 4DSS database.

Once a week, the subject met with his physician and AI researchers to review and analyze the data. The consolidated data was displayed via custom-built visualization software called PhysioGraph. A screen shot from PhysioGraph, showing the different types of data, is shown in Figure 4. BG control problems identified by the 4DSS software, the subject, and/or his physician, were visualized and discussed. We looked for visual patterns in the fitness band data during the times when the problems occurred.

Preliminary findings based on these visualizations were encouraging. While even subtle patterns may be detected by machine learning algorithms, we were able to detect some marked patterns as humans. The most pronounced pattern was a rise in GSR with severe hypoglycemia. The most interesting pattern revolved around shoveling snow. The study was conducted during an unusually harsh winter in which the patient (and most of the rest of us) had to frequently shovel heavy snow. Shoveling snow is strenuous exercise, and exercise is known to lower BG levels. After shoveling for extended periods, the subject sometimes experienced hypoglycemia. This is a problem we would like to predict, because, if alerted, the patient could take action to prevent it. There was a discernable pattern in the fitness band data surrounding this problem. GSR and heart rate rose, while skin and air temperature dropped. While we do not yet know if this

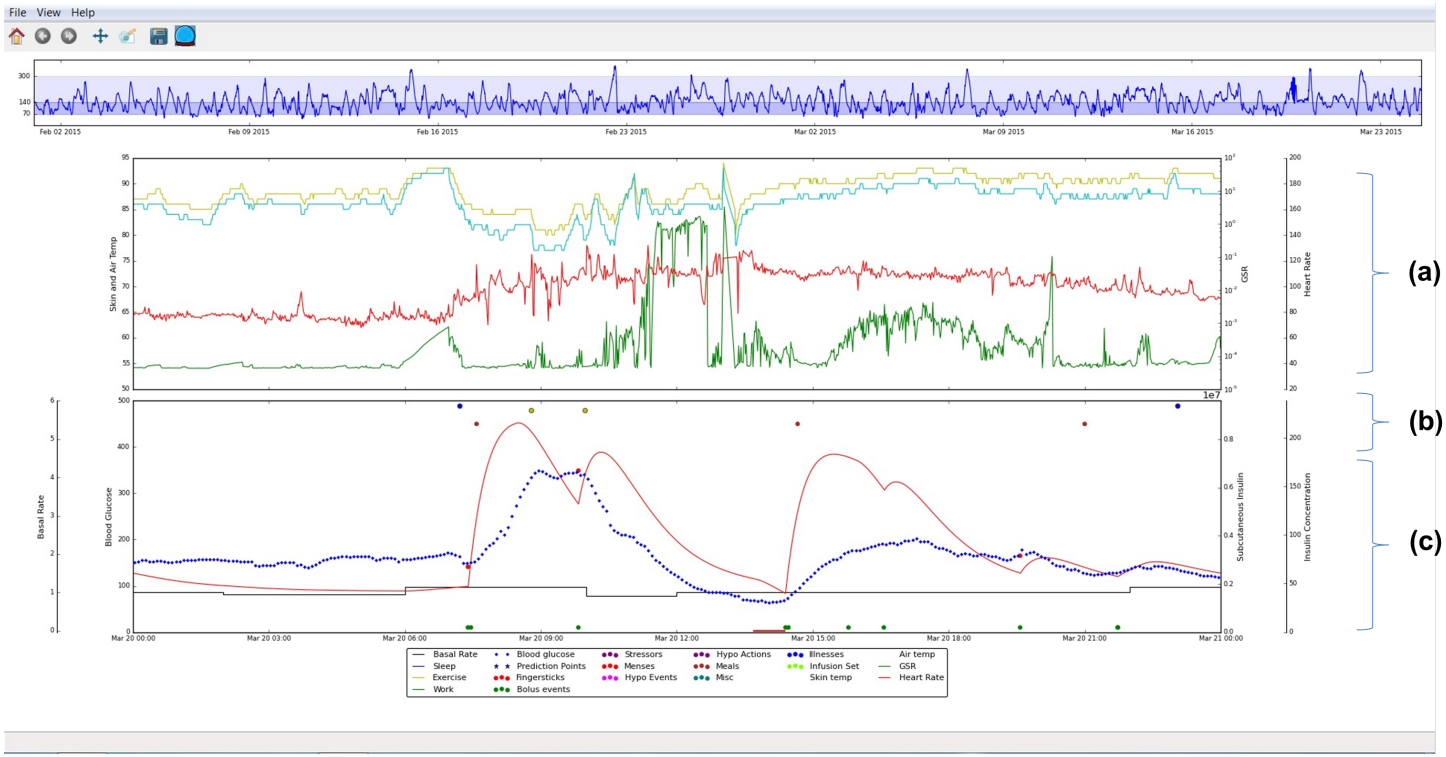


**Fig. 4.** A Screen Shot from PhysioGraph. Data from the fitness band is displayed in section (a). This includes GSR (green), heart rate (red), skin temperature (gold) and air temperature (cyan). Icons indicating life events appear in section (b). These icons are clickable in PhysioGraph to view event details. BG data and insulin data are shown in section (c). The dotted curve (blue) represents CGM data, while the solid curve (red) models the total insulin in the patient's system.

combination of signals will allow us to predict hypoglycemia or only detect it, we now have leads to follow.

## 4 Discussion

When work began on the 4DSS in 2004, we had little data and no structured cases. As we built our case base and collected data from over 50 T1D patients, we developed a database that enabled us to build machine learning models for purposes we did not envision in 2004. With more data, it becomes possible to leverage more AI approaches for more purposes. When data or knowledge is limited, however, CBR can be an enabling approach. As non-health-related examples, we can think of leveraging all of the information possible from a single oil spill, or of basing product recommendations for a customer with no purchase history on what similar customers have bought. In the medical arena, CBR has also proven useful for dealing with new situations. For example, CARE-PARTNER used CBR to determine appropriate follow-up care for the earliest stem cell transplant patients [1]. Once many patients had undergone stem cell transplantation and received follow-up care, their collective experiences were distilled into clinical practice guidelines that were used in lieu of CBR. Today, nearly 20 years later, big data tools like IBM Watson Health [5] may allow us to further evaluate, refine, and personalize treatment for these patients.

In the diabetes domain, big data is not yet publicly available; however, we anticipate its near-term future availability. Three non-technical factors contribute to this lack of data: (1) most diabetes patients do not yet wear devices or use systems that continuously collect data; (2) medical device manufacturers do not yet allow access to raw data in real-time, but require the use of their own proprietary software; and (3) patient privacy concerns inhibit data sharing, even when data exists. There has been a recent drive to collect and consolidate data from all T1D patients and all types of (currently incompatible) medical devices, spearheaded by the non-profit organization Tidepool [10]. A goal is to be able to analyze and leverage continuous data from hundreds of thousands of patients to improve diabetes care and outcomes for individuals. Our case engineering, based on the limited data we have already collected, could serve to jump start such efforts.

At the heart of any CBR system is the case. The case is a knowledge structure that, especially in complex medical domains, may embody more than a collection of readily available feature-value pairs. The design of a case for a CBR system begins with the analysis of real-world cases to identify problems, solutions and outcomes. It is necessary to understand and define the features that make cases similar to each other, reusable in different circumstances, and adaptable to the case at hand. Features engineered for cases may provide machine learning algorithms with better inputs than raw data or surface features.

In our case study, the fitness band provided physiological signals that were not a part of our original case design. There were 20 times as many data points per patient per day as we had previously collected. We did not know how the

new data could be used to anticipate or detect blood glucose control problems or, more fundamentally, how it related to BG levels in people with T1D. The inputs to our SVR models are not raw data points, but features that have been carefully engineered from the raw data based, in part, on insights gained from cases. Sometimes, simple data combinations suffice; for example, we could see during the case study that the difference between air and skin temperature was more relevant than either individual measurement. Other times, we have had to employ complex systems of equations; for example, a complex physiological model is needed to characterize the impact of insulin on BG levels. Even as we move toward more automated means of feature engineering and more reliance on machine learning techniques, early use of CBR can help to provide insight and intuition that may guide big data exploration and interpretation.

## 5 Summary and Conclusion

The 4DSS is a prototypical hybrid CBR system that aims to help T1D patients achieve and maintain good BG control. As cases and data have accumulated over ten years, the research emphasis has shifted toward using the accumulated data to build machine learning models for BG prediction. While these models have applicability to situation assessment within the 4DSS, their greater potential is in facilitating a wide range of practical applications that could enhance safety and quality of life for T1D patients. The recent proliferation of commercially available fitness bands has presented the opportunity to incorporate new types of data indicative of patient activity into the models to improve prediction accuracy. However, when it was initially unclear whether or how this new data might be leveraged, a case study was conducted, calling CBR back into play.

In the N-of-1 study, a T1D patient on insulin pump therapy with continuous glucose monitoring wore a fitness band and entered life-event data into the 4DSS database for two months. The aggregated data was displayed via custom-built visualization software and reviewed at weekly intervals by the patient, his physician, and AI researchers. BG control problems were analyzed with a focus on identifying patterns in the new data at the time the problems occurred. Some promising patterns could be visualized, including a marked rise in GSR with severe hypoglycemia. This case-based focus provided insight and intuition about how the new data relates to BG levels. Work on integrating the new data into our BG prediction models is currently underway.

## 6 Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 1117489. Additional support was provided by Medtronic and Ohio University. We gratefully acknowledge the contributions of diabetologist Jay Shubrook, DO, graduate research assistants Kevin Plis and Samson Xia, and Research Experience for Undergraduates (REU) participants Hannah Quillin and Charlie Murphy.

## References


1. Bichindaritz, I., Kansu, E., Sullivan, K.M.: Case-based reasoning in CAREPARTNER: Gathering evidence for evidence-based medical practice. In: Smyth, B., Cunningham, P. (eds.) Advances in Case-Based Reasoning: 4th European Workshop, Proceedings EWCBR-98. pp. 334–345. Springer-Verlag, Berlin (1998)
2. Bunescu, R., Struble, N., Marling, C., Shubrook, J., Schwartz, F.: Blood glucose level prediction using physiological models and support vector regression. In: Proceedings of the Twelfth International Conference on Machine Learning and Applications (ICMLA). pp. 135–140. IEEE Press (2013)
3. Cryer, P.E.: Hypoglycemia: Still the limiting factor in the glycemic management of diabetes. Endocrine Practice 14(6), 750–756 (2008)
4. Diabetes Control and Complications Trial Research Group: The effect of intensive treatment of diabetes on the development and progression of long-term complications in insulin-dependent diabetes mellitus. New England Journal of Medicine 329(14), 977–986 (1993)
5. IBM: IBM Watson Health (2015), http://www.ibm.com/smarterplanet/us/en/ibmwatson/health/, accessed July, 2015
6. Marling, C., Bunescu, R., Shubrook, J., Schwartz, F.: System overview: The 4 Diabetes Support System. In: Lamontagne, L., Recio-García, J.A. (eds.) Workshop Proceedings of the Twentieth International Conference on Case-Based Reasoning. pp. 81–86. Lyon, France (2012)
7. Marling, C., Wiley, M., Bunescu, R., Shubrook, J., Schwartz, F.: Emerging applications for intelligent diabetes management. AI Magazine 33(2), 67–78 (2012)
8. Plis, K., Bunescu, R., Marling, C., Shubrook, J., Schwartz, F.: A machine learning approach to predicting blood glucose levels for diabetes management. In: Modern Artificial Intelligence for Health Analytics: Papers Presented at the Twenty-Eighth AAAI Conference on Artificial Intelligence. pp. 35–39. AAAI Press (2014)
9. Schwartz, F.L., Shubrook, J.H., Marling, C.R.: Use of case-based reasoning to enhance intensive management of patients on insulin pump therapy. Journal of Diabetes Science and Technology 2(4), 603–611 (2008)
10. Tidepool: The Tidepool Platform: A home for diabetes data (2015), http://tidepool.org/platform/, accessed August, 2015
11. World Health Organization: 10 facts about diabetes (2014), http://www.who.int/features/factfiles/diabetes/facts/en/, accessed July, 2015

# Data Mining Methods for Case-Based Reasoning in Health Sciences

Isabelle Bichindaritz
Computer Science Department, State University of New York
Oswego, NY, USA

**Abstract.** Case-based reasoning (CBR) systems often refer to diverse data mining functionalities and algorithms. This article locates examples, many from health sciences domains, mapping data mining functionalities to CBR tasks and steps, such as case mining, memory organization, case base reduction, generalized case mining, indexing, and weight mining. Data mining in CBR focuses greatly on incremental mining for memory structures and organization with the goal of improving performance of retrieval, reuse, revise, and retain steps. Researchers are aiming at the ideal memory as described in the theory of the dynamic memory, which follows a cognitive model, while also improving performance and accuracy in retrieve, reuse, revise, and retain steps. Several areas of potential cross-fertilization between CBR and data mining are also proposed.

## 1 Introduction

Case-based reasoning (CBR) systems have tight connections with machine learning and data mining as exemplified by their description in data mining (Han et al. 2012) and machine learning (Mitchell 1997) textbooks. They have been tagged by machine learning researchers as *lazy* learners because they defer the decision of how to generalize beyond the training set until a target new case is encountered (Mitchell 1997), by opposition to most other learners, tagged as *eager*. Even though a large part of the inductive inferences are definitely performed at *Retrieval* time in CBR (Aha 1997), mostly through sophisticated similarity evaluation, most CBR systems also perform inductive inferences at *Retain* time. There is a long tradition within this research community to study what is a memory, and what its components and organization should be. Indeed CBR methodology focuses more on the memory part of its intelligent systems (Schank 1982) than any other artificial intelligence (AI) methodology, and this often entails learning declarative memory structures and organization. This article proposes to review the main data mining functionalities and how they are used in CBR systems by describing examples of systems using them and analyzing which roles they play in the CBR framework (Aamodt and Plaza 1994). The research question addressed is to de-

termine the extent to which data mining functionalities are being used in CBR systems, to enlighten possible future research collaborations between these two fields, particularly in health sciences applications. This paper is organized as follows. After the introduction, the second section highlights major concepts and techniques in data mining. The third section reviews the main CBR cycle and principles. The fourth section explains relationships between CBR and machine learning. The following sections dive into several major data mining functionalities and how they relate to CBR. The ninth section summarizes the findings and proposes future directions. It is followed by the conclusion.

## 2 Data Mining Functionalities and Methods

Data mining is the analysis of observational data sets to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful to the data owner (Hand et al. 2001). Traditionally described as a misnomer, knowledge discovery or knowledge discovery in databases is a preferred term. Some functionalities are clearly well defined and researched, among which (Han et al. 2012):

- **Classification / prediction:** classification is a supervised data mining method applied to datasets containing an expert labeling in the form of a categorical attribute, called a class; when the attribute is numeric, the method is called prediction. Examples of classifiers include neural networks, support vector machines (SVMs), naïve Bayes, and decision trees.
- **Association Mining**: association mining mines for frequent itemsets in a dataset, which can be represented as rules such as in market basket analysis. It is an unsupervised method. The most famous algorithm in this category is a priori algorithm.
- **Clustering:** clustering finds groups of similar objects in a dataset, which are also dissimilar from the objects in other clusters. In addition to the similarity-based methods like K Means, some methods use density-based algorithms or hierarchical algorithms.

Considerations for evaluating the mining results vary in these different methods, however a set of quality measurements are traditionally associated with each, for example accuracy or error rate for classification, and lift or confidence for association mining.

These core functionalities can be combined and applied to several data types, with extensions to the underlying algorithms or completely new methods.in addition to the classical nominal and numeric data types. Well researched data types are graphs, texts, images, time series, networks, streams, etc. We refer to these extensions as multimedia mining.

Other types of functionalities, generally combined with the core ones are for example feature selection, where the goal is to select a subset of features, sampling, where the goal is to select a subset of input rows, and characterization,

where the goal is to provide a summary representation of a set of rows, for example those contained in a cluster.

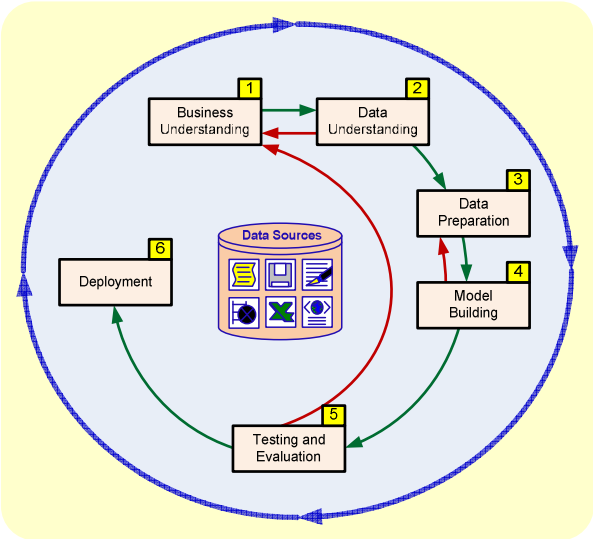


**Fig. 1.** CRISP-DM data mining process (Han et al. 2012)

Finally, the CRISP-DM methodology has been described to guide the data mining process (see Fig. 1) (Han et al. 2012). This methodology stresses the importance of stages preparing for and following the actual model building stage: data preparation, dealing with issues such as data consolidation, data cleaning, data transformation, and data reduction, which can require up to 85% of all the time dedicated to a project.

## 3 CBR Cycle and Methods

Case Based Reasoning is a problem solving methodology that aims at reusing previously solved and memorized problem situations, called cases. Traditionally, its reasoning cycle proceeds through steps (see Fig. 2). This article will refer to the major steps as Retrieve, Reuse, Revise, and Retain (Aamodt and Plaza 1994).

## 4 CBR and Machine Learning

CBR systems are generally classified as data mining systems because they can perform classification or prediction tasks. From a set of data – called cases in CBR – the classification or prediction achieved gives the case base a competency be-

yond what the data provide. If CBR systems are in par with data mining systems in such tasks as classification and prediction, there is, though an important difference. CBR systems start their reasoning from knowledge units, called cases, while data mining systems most often start from raw data. This is why case mining, which consists in mining raw data for these knowledge units called cases, is a data mining task often used in CBR. CBR systems also belong to instance based learning systems in the field of machine learning, defined as systems capable of automatically improving their performance over time. Although there is much commonality between data mining and machine learning, their definitions and goals are different. CBR systems are problem-solving systems following a reasoning cycle illustrated in Fig. 1. However as long as they learn new cases in their retain step, they are qualified as learning systems, thus belonging to machine learning system.

For this article, we will focus on identifying which data mining functionalities and methods are used in CBR, and what is their result in the CBR memory.

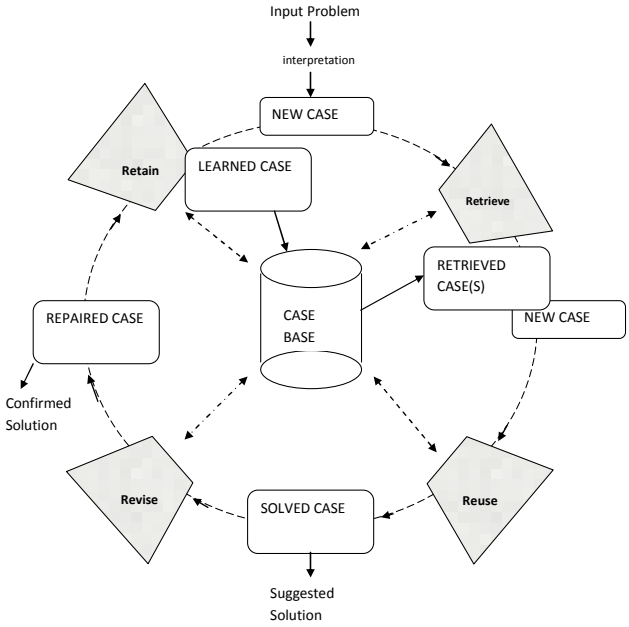


**Fig 2.** The classical CBR reasoning cycle (Aamodt and Plaza 1994)

First of all, since data mining emerged in the 90's from scaling up machine learning algorithms to large datasets, let us review what machine learning authors have been saying about CBR. They consider case-based reasoning systems as either analogical reasoning systems (Michalski 1993), or instance based learners (Mitchell 1997). Michalski (1993) presents the analogical inference, at the basis of case-based retrieval, as a dynamic induction performed during the matching process. Mitchell (1997) refers to CBR as a kind of instance based learner. This au-

thor labels these systems as *lazy* learners because they defer the decision about how to generalize beyond the training data until each new query instance is encountered. This allows CBR systems to not commit to a global approximation once and for all during the training phase of machine learning, but to generalize specifically for each target case, therefore to fit its approximation bias, or induction bias, to the case at hand. He points here to the drawback of overgeneralization that is well known for eager learners, from which instance based learners are exempt (Mitchell 1997).

These authors focus their analysis on the inferential aspects of learning in case-based reasoning. Historically CBR systems have evolved from the early work of Schank in the theory of the dynamic memory (Schank 1982), where this author proposes to design intelligent systems primarily by modeling their memory. Ever since Schank's precursory work on natural language understanding, one of the main goals of case-based reasoning has been to integrate as much as possible memory and inferences for the performance of intelligent tasks. Therefore focusing on studying how case-based reasoning systems learn, or mine, their memory structures and organization can prove at least as fruitful as studying and classifying them from an inference standpoint.

From a memory standpoint, learning in CBR consists in the creation and maintenance of the structures and organization in memory. It is often referred to as case base maintenance (Wilson and Leake 2001). In the general cycle of CBR, learning takes place within the reasoning cycle - see (Aamodt and Plaza 1994) for this classical cycle. It completely serves the reasoning, and therefore one of its characteristics is that it is an *incremental* type of mining. It is possible to fix it after a certain point, though; in certain types of applications, but it is not a tradition in CBR: *learning is an emergent behavior from normal functioning* (Kolodner 1993). When an external problem-solving source is available, CBR systems start reasoning from an empty memory, and their reasoning capabilities stem from their progressive learning from the cases they process. Aamodt and Plaza (1994) further state that *case-based reasoning favours learning from experience*. The decision to stop learning because the system is judged competent enough is not taken from definitive criteria. It is the consequence of individual decisions made about each case, to keep it or not in memory depending upon its potential contribution to the system. Thus often the decisions about each case, each structure in memory, allow the system to evolve progressively toward states as different as ongoing learning, in novice mode, and its termination, in expert mode. If reasoning and thus learning are directed from the memory, learning answers to a process of prediction of the conditions of cases recall (or retrieval). As the theory of the dynamic memory showed, recall and learning are closely linked (Schank 1982). Learning in case-based reasoning answers a disposition of the system to anticipate future situations: *the memory is directed toward the future* both to avoid situations having caused a problem and to reinforce the performance in success situations.

More precisely, learning in case-based reasoning, takes the following forms:

1. Adding a case to the memory: it is at the heart of CBR systems, traditionally one of the main phases in the reasoning cycle, and the last one: *Retain* (Aamodt

and Plaza 1994). It is the most primitive learning kind, also called learning by consolidation, or rote learning.
2. Explaining: the ability of a system to find explanations for its successes and failures, and by generalization the ability to anticipate.
3. Choosing the indices: it consists in anticipating *Retrieval*, the first reasoning step.
4. Learning memory structures: these may be learnt by generalization from cases or be provided from the start to hold the indices for example. These learnt memory structures can play additional roles, such as facilitating reuse or retrieval.
5. Organizing the memory: the memory comprises a network of cases, given memory structures, and learned memory structures, organized in efficient ways. Flat and hierarchical memories have been traditionally described.
6. Refining cases: cases may be updated, refined based upon the CBR result.
7. Discovering knowledge or metareasoning: the knowledge at the basis of the case-based reasoning can be refined, such as modifying the similarity measure (weight learning), or situation assessment refinement. For example d'Aquin et al. (2007) learn new adaptation rules through knowledge discovery.

## 5 Classification / Prediction and CBR

Since CBR is often used as a classifier, other classifiers are generally used in ensemble learning to combine the CBR expertise with other classification/prediction algorithms. Another type of combination of classifier is to use several CBR systems as input to another classifier, for example SVM, applied to the task of predicting business failure (Li and Sun 2009).

Another notable class of systems is composed of those performing *decision tree induction* to organize their memory. INRECA (Auriol et al. 1994) project studied how to integrate CBR and decision tree induction. They propose to preprocess the case base by an induction tree algorithm, namely a decision tree. Later refined into an *INRECA tree* (see Fig. 2), which is a hybrid between a decision tree and a k-d tree, this method allows both similarity based retrieval and decision tree retrieval, is incremental, and speeds up the retrieval. This system was used in biological domains among others.

## 6 Association Mining and CBR

Association mining, although not looking closely related to CBR, can be resorted in several scenarios. Main uses are for case mining and case base maintenance.

Wong et al. (2001) use fuzzy association rule mining to learn cases from a web log, for future reuse through CBR.

Liu et al. (2008) use frequent item sets mining to detect associations between cases, and thus detect cases candidate for removal from the case base and thus its reduction (Retain step).

## 7 Clustering and CBR

Memory structures in CBR are foremost cases. A case is defined as a contextualized piece of knowledge representing an experience that teaches a lesson fundamental to achieving the goals of a reasoner (Kolodner 1993). For many systems, cases are represented as truthfully as possible to the application domain. Additionally, data mining methods have been applied to cases themselves, features, and generalized cases. These techniques can be applied concurrently to the same problem, or selectively. If the trend is now to use them selectively, probably in the near future CBR systems will use these methods more and more concurrently.

### 7.1 Case mining

Case mining refers to the process of mining potentially large data sets for cases (Yang and Cheng 2003). Researchers have often noticed that cases simply do not exist in electronic format, that databases do not contain well-defined cases, and that the cases need to be created before CBR can be applied. Instead of starting CBR with an empty case base, when large databases are available, preprocessing these to learn cases for future CBR permits to capitalize on the experience dormant in these databases. Yang and Cheng (2003) propose to learn cases by linking several database tables through *clustering* and *Support Vector Machines (SVM)*. The approach can be applied to learning cases from electronic medical records (EMRs).

### 7.2 Generalized case mining

Generalized case mining refers to the process of mining databases for generalized and/or abstract cases. Generalized cases are named in varied ways, such as prototypical cases, abstract cases, prototypes, stereotypes, templates, classes, ossified cases, categories, concepts, and scripts – to name the main ones (Maximini et al. 2003). Although all these terms refer to slightly different concepts, they represent structures that have been abstracted or generalized from real cases either by the CBR system, or by an expert. When these prototypical cases are provided by a domain expert, this is a knowledge acquisition task. More frequently they are learnt from actual cases. In CBR, prototypical cases are often learnt to structure the memory. Therefore most of the prototypical cases presented here will also be listed in the section on structured memories.

In medical domains, many authors mine for *prototypes*, and simply refer to *induction* for learning these. CHROMA (Armengol and Plaza 1994) uses induction to learn prototypes corresponding to general cases. Bellazzi et al. organize their memory around prototypes (Bellazzi et al. 1998). The prototypes can either have been acquired from an expert, or induced from a large case base. Schmidt and Gierl (1998) point that prototypes are an essential knowledge structure to fill the gap between general knowledge and cases in medical domains. The main purpose of this prototype learning step is to guide the retrieval process and to decrease the amount of storage by erasing redundant cases. A generalization step becomes necessary to learn the knowledge contained in stored cases.

Others specifically refer to *generalization,* so that their prototypes correspond to generalized cases. For example Malek proposes to use a *neural network* to learn the prototypes in memory for a classification task, such as diagnosis (Malek 1995). Portinale and Torasso (1995) in ADAPTER organize their memory through E-MOPs (Kolodner 1993) learnt by generalization from cases for diagnostic problem-solving. Maximini et al. (2003) have studied the different structures induced from cases and point out that several different terms exist, such as generalized case, prototype, schema, script, and abstract case. The same terms do not always correspond to the same type of entity. They define three types of cases. A point case is what we refer to as a real or ground case. The values of all its attributes are known. A generalized case is an arbitrary subset of the attribute space.

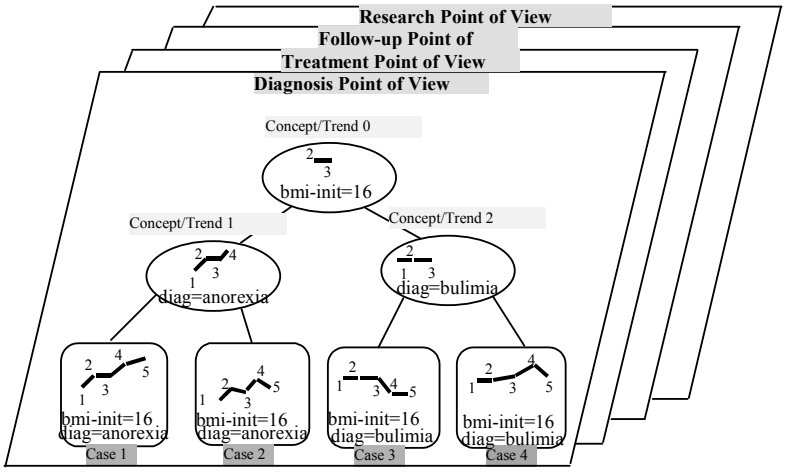


**Fig. 3.** Hierarchical memory organization in MNAOMIA: concepts are learnt during CBR for diagnosis, treatment, and/or follow-up, and can be reused by research task (Bichindaritz 1995)

There are two forms: the attribute independent generalized case, in which some attributes have been generalized (interval of values) or are unknown, and the attribute dependent generalized case, which cannot be defined from independent subsets of their attributes.

Finally, many authors learn *concepts* through *conceptual clustering*. MNAOMIA (Bichindaritz 1995) learns concepts and trends from cases through *conceptual clustering* (see Fig. 3). Perner learns a hierarchy of classes by *hierarchical conceptual clustering*, where the concepts represent clusters of prototypes (Perner 1998).

Dìaz-Agudo and Gonzàlez-Calero (2003) use *formal concept analysis* (FCA) – a mathematical method from data analysis - as another induction method for extracting knowledge from case bases, in the form of *concepts*. The authors point to one notable advantage of this method, during adaptation. The FCA structure induces dependencies among the attributes that guide the adaptation process (Dìaz-Agudo et al. 2003). Napoli (2010) stresses the important role FCA can play for classification purposes in CBR, through learning a case hierarchy, indexing, and information retrieval.

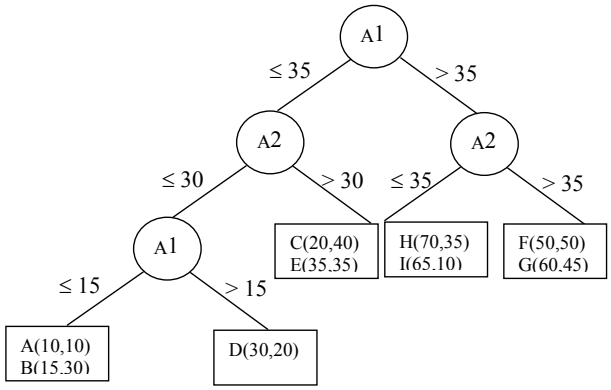


**Fig. 4.** Tree memory organization in INRECA using k-d trees (Auriol et al. 1994)

### 7.3 Mining for Memory Organization

Efficiency at case retrieval time is conditioned by a judicious memory organization. Two main classes of memory are presented here: unstructured – or flat – memories, and structured memories.

## Flat memories

Flat memories are memories in which all cases are organized at the same level. Retrieval in such memories processes all the cases in memory. Classical nearest neighbor (kNN) retrieval is a method of choice for retrieval in flat memories. Flat memories can also contain prototypes, but in this case the prototypical cases do not serve as indexing structures for the cases. They can simply replace a cluster of similar cases that has been deleted from the case base during case base maintenance activity. They can also have been acquired from experts. Flat memories are

the memories of predilection of kNN retrieval methods (Aha 1997) and of so-called memory-based systems.

### Structured memories

Among the different structured organizations, the accumulation of generalizations or abstractions facilitates the evaluation of the situation the control of indexation.

Structured memories, dynamic, present the advantage of being declarative. The important learning efforts in declarative learning are materialized in the structures and the dynamic organization of their memories. In medical imaging, Perner learns a hierarchy of classes by *hierarchical conceptual clustering*, where the concepts are clusters of prototypes (Perner 1998). She notes the advantages of this method: a more compact case base, and more robust (error-tolerant).

MNAOMIA (Bichindaritz 1995) proposes to use *incremental concept learning*, which is a form of hierarchical clustering, to organize the memory. This system integrates highly data mining with CBR because it reuses the learnt structures to answer higher level tasks such as generating hypotheses for clinical research (see Fig. 3), as a side effect of CBR for clinical diagnosis and treatment decision support. Therefore this system illustrates that by learning memory structures in the form of concepts, the classical CBR classification task improves, and at the same time the system extracts what it has learnt, thus adding a knowledge discovery dimension to the classification tasks performed.

Another important method, presented in CHROMA (Armengol and Plaza 1994), is to organize the memory like a hierarchy of objects, by *subsomption*. Retrieval is then a classification in a hierarchy of objects, and functions by substitution of values in slots. CHROMA uses its prototypes, induced from cases, to organize its memory. The retrieval step of CBR retrieves relevant prototypes by using subsomption in the object oriented language NOOS to find the matching prototypes.

Many systems use personalized memory organizations structured around several layers or *networks*, for example neural networks (Malek 1995).

Another type of memory organization is the *formal concept lattice*. Dìaz-Agudo and Gonzàlez-Calero (2003) organize through formal concept analysis (FCA) the case base around *Galois lattices*. Retrieval step is a classification in a concept hierarchy, as specified in the FCA methodology, which provides such algorithms (Napoli 2010). The concepts can be seen as an alternate form of indexing structure.

Yet other authors take advantage of the *B-tree structure* implementing databases and retrieve cases using database SQL query language over a large case base stored in a database (West and McDonald 2003).

## 8 Feature Selection and CBR

Feature mining refers to the process of mining data sets for features. Many CBR systems select the features for their cases, and/or generalize them. Wiratunga et al.

(2004) notice that transforming textual documents into cases requires dimension reduction and/or feature selection, and show that this preprocessing improves the classification in terms of CBR accuracy – and efficiency. These authors induce a kind of decision tree called boosted *decision stumps,* comprised of only one level, in order to select features, and *induce rules* to generalize the features. In biomedical domains, in particular when data vary continuously, the need to abstract features from streams of data is particularly prevalent. Other, and notable, examples include Montani et al., who reduce their cases time series dimensions through *Discrete Fourier Transform* (Montani et al. 2004), approach adopted by other authors for time series (Nilsson and Funk 2004). Niloofar and Jurisica propose an original method for generalizing features. Here the generalization is an abstraction that reduces the number of features stored in a case (Niloofar and Jurisica 2004). Applied to the bioinformatics domain of micro arrays, the system uses both *clustering* techniques to group the cases into clusters containing similar cases, and *feature selection* techniques.

**Table 1.** Data mining functionalities versus CBR steps map – methods italicized represent future directions

| | ***Classification / prediction*** | ***Association mining*** | ***Clustering*** | ***Feature selection*** |
|---|---|---|---|---|
| ***Data preparation / Metareasoning*** | **Ensemble learning** | **Case mining** | **Case mining** | |
| ***Retrieve*** | ***Opportunistic similarity knowledge mining*** | | | |
| ***Reuse*** | ***Opportunistic reuse knowledge mining*** | | | |
| ***Revise*** | ***Opportunistic revise knowledge mining*** | | | |
| ***Retain*** | **Memory organization** | **Case base reduction** | **Generalized case mining**<br><br>**Memory organization** | **Indexing**<br><br>**Weight learning** |

## 9 Discussion and Future Directions

In addition to the main functionalities listed above, multimedia mining extends the algorithms to the form taken by cases and the type of their features for the same kinds of applications previously listed.

In summary, if we map the different data mining functionalities and the CBR steps / tasks, we notice on Table 1 that the steps benefitting the most from data mining are Retain, Data preparation and Metareasoning. This is not surprising because these steps are the most involved in declarative knowledge learning or updating. However the processing intensive steps such as Retrieve, Reuse and Re-

vise do not seem to resort to data mining beside the dynamic induction mentioned in Section 4.

Interesting areas to explore could be feature selection functionality for case mining, data preparation, or metareasoning. Retrieve, Reuse, and Revise could also explore the use of data mining. For retrieval, in addition to weight learning already mentioned, learning similarity measures (Stahl 2005), or improving on an existing one, would be valuable. For reuse or revise, learning adaptation rules or revision rules or models would be highly pertinent – and some work has started in these areas (Badra et al. 2009). These synergies could take place during the Retain step, but also in an opportunistic fashion during the processing steps (see Table 1).

We can also foresee such synergies with Big Data for the processing of large datasets in distributed main memory that can make efficient use of data mining during processing on a larger scale. It is therefore very important for CBR researchers and professionals to gain expertise in data mining advances and their applicability to CBR.

CBR research focuses mostly on the model building stage of CRISP-DM. Other aspects of the CRISP-DM methodology would also be interesting for CBR synergies, for example aspects of data understanding, data preparation, testing, evaluation, and deployment in relationship with CBR to make this methodology more robust to fielded applications.

## 10 Conclusion

CBR systems make efficient use of most data mining tasks defined for descriptive modeling. We can list among the main ones encountered in biomedical domains, cluster analysis, rule induction, hierarchical cluster analysis, and decision tree induction. The motivations for performing an incremental type of data mining during CBR are several folds, and their efficiency has been measured to validate the approach. The main motivations are the following:

- Increase efficiency of retrieval mostly, but also of reuse, revise, and retain steps.
- Increase robustness, tolerance to noise.
- Increase reasoning accuracy and effectiveness.
- Improve storage needs.
- Follow a cognitive model.
- Add functionality, such as a synthetic task like generating new research hypotheses as a side effect of normal CBR functioning.
- Perform metareasoning, such as knowledge discovery to learn new adaptation rules.

The memory organization maps directly into the retrieval method used. For example, generalized cases and the like are used both as indexing structures, and organizational structures. We can see here a direct mapping with the theory of the

dynamic memory, which constantly influences the CBR approach. The general idea is that the learned memory structures and organizations condition what inferences will be performed, and how. This is a major difference with database approaches, which concentrate only on retrieval, and also with data mining approaches, which concentrate only on the structures learned, and not on how they will be used. Opportunistic use of data mining during the retrieval, reuse, and revise steps would bring a more robust dimension to CBR by learning when a need arises, instead of, or in addition to, systematically at Retain. The ideal CBR memory is one which at the same time speeds up the retrieval step, and improves effectiveness, efficiency, and robustness of the task performed by the reasoner, and particularly the reuse performed, influencing positively both the retrieval, the reuse and the other steps. Researchers do not want to settle for a faster retrieval at the expense of less accuracy due to an overgeneralization. And they succeed at it.

Future work involves revisiting these data mining techniques in the framework of the knowledge containers identified by Richter (2003) and constantly tracking novel methods used as they appear. The variety of approaches as well as the specific and complex purpose lead to thinking that there is space for future models and theories of CBR memories, in particular embracing metareasoning and opportunistic approaches more systematically, and where data mining will play a larger role.

## 11 References

Aamodt A, Plaza E (1994) Case-Based Reasoning: Foundational Issues, Methodologies Variations, and Systems Approaches. AI Communications, IOS Press, Vol. 7: 1:39-59

Aha DW (1997) Lazy Learning. Artificial Intelligence Review 11:7-10

Armengol E, Plaza E (1994) Integrating induction in a case-based reasoner. In: Keane M, Haton JP, Manago M (eds) Proceedings of EWCBR 94. Acknosoft Press, Paris, pp 243-251

Auriol E, Manago M, Althoff KD, Wess S, Dittrich S (1994) Integrating Induction and Case-Based Reasoning: Methodological Approach and First Evaluations. In: Keane M, Haton JP, Manago M (eds) Proceedings of EWCBR 94. Acknosoft Press, Paris, pp 145-155

Badra F, Cordier A, Lieber J (2009) Opportunistic Adaptation Knowledge Discovery. In: McGinty L, Wilson DC (eds) Proceedings of ICCBR 09. Springer-Verlag, Lecture Notes in Artificial Intelligence, Berlin, Heidelberg, New York, pp 60-74

Bellazzi R, Montani S, Portinale L (1998) Retrieval in a Prototype-Based Case Library: A Case Study in Diabetes Therapy Revision. In: Smyth B, Cunningham P (eds) Proceedings of ECCBR 98. Springer-Verlag, Lecture Notes in Artificial Intelligence 1488, Berlin, Heidelberg, New York, pp 64-75.

Bichindaritz I (1995) A case-based reasoner adaptive to several cognitive tasks. In: Veloso M., Aamodt A (eds) Proceedings of ICCBR 95. Springer-Verlag,

Lecture Notes in Artificial Intelligence 1010, Berlin, Heidelberg, New York, pp 391-400

d'Aquin M, Badra F, Lafrogne S, Lieber J, Napoli A, Szathmary L (2007) Case Base Mining for Adaptation Knowledge Acquisition. In : IJCAI. 7, pp. 750-755

Dìaz-Agudo B, Gervàz P, Gonzàlez-Calero P (2003) Adaptation Guided Retrieval Based on Formal Concept Analysis. In: Ashley K, Bridge DG (eds) Proceedings of ICCBR 03. Springer-Verlag, Lecture Notes in Artificial Intelligence 2689, Berlin, Heidelberg, New York, pp 131-145

Han J, Kamber M, Pei J (2012) Data Mining concepts and Techniques. Morgan Kaufmann, Waltham, Massachussetts

Hand D, Mannila H, Smyth P (2001) Principles of Data Mining. The MIT Press, Cambridge, Massachusetts

Kolodner JL (1993) Case-Based Reasoning. Morgan Kaufmann Publishers, San Mateo, California

Leake, DB, & Wilson, DC (1998). Categorizing Case-base Maintenance: Dimensions and Directions. In: Advances in Case-Based Reasoning. Springer Berlin Heidelberg, pp. 196-207

Li H, Sun J, (2009) Predicting business failure using multiple case-based reasoning combined with support vector machine, Expert Systems with Applications, Volume 36, Issue 6, pp 10085-10096

Liu C-H, Chen L-S, Hsu C-C (2008) An association-based case reduction technique for case-based reasoning, Information Sciences, Volume 178, Issue 17 pp. 3347-3355

Malek M (1995) A Connectionist Indexing Approach for CBR Systems. In: Veloso M, Aamodt A (eds) Proceedings of ICCBR 95. Springer-Verlag, Lecture Notes in Artificial Intelligence 1010, Berlin, Heidelberg, New York, pp 520-527

Maximini K, Maximini R, Bergmann R (2003) An Investigation of Generalized Cases. In: Ashley KD, Bridge DG (eds) Proceedings of ICCBR 03. Springer-Verlag, Lecture Notes in Artificial Intelligence 2689, Berlin, Heidelberg, New York, pp 261-275

Michalski RS (1993) Toward a Unified Theory of Learning. In: Buchanan BG, Wilkins DC (eds) Readings in knowledge acquisition and learning, automating the construction and improvement of expert systems. Morgan Kaufmann Publishers, San Mateo, California, pp 7-38

Mitchell TM (1997) Machine Learning. Mc Graw Hill, Boston, Massachusetts

Montani S, Portinale L, Bellazzi R, Leornardi G (2004) RHENE: A Case Retrieval System for Hemodialysis Cases with Dynamically Monitored Parameters. In: Funk P, Gonzàlez Calero P (eds) Proceedings of ECCBR 04. Springer-Verlag, Lecture Notes in Artificial Intelligence 3155, Berlin, Heidelberg, New York, pp 659-672

Napoli A (2010) Why and How Knowledge Discovery Can Be Useful for Solving Problems with CBR. In: Proceedings of ICCBR 10. Springer-Verlag, Lecture Notes in Artificial Intelligence, Berlin, Heidelberg, New York, pp. 12-19

Niloofar A, Jurisica I (2004) Maintaining Case-Based Reasoning Systems: A Machine Learning Approach. In: Funk P, Gonzàlez Calero P (eds) Proceedings of ECCBR 04. Springer-Verlag, Lecture Notes in Artificial Intelligence 3155, Berlin, Heidelberg, New York, pp 17-31

Nilsson M, Funk P (2004) A Case-Based Classification of Respiratory sinus Arrhythmia. In: Funk P, Gonzàlez Calero P (eds) Proceedings of ECCBR 04. Springer-Verlag, Lecture Notes in Artificial Intelligence 3155, Berlin, Heidelberg, New York, pp 673-685

Perner P (1998) Different Learning Strategies in a Case-Based Reasoning System for Image Interpretation. In: Smyth B, Cunningham P (eds) Proceedings of ECCBR 98. Springer-Verlag, Lecture Notes in Artificial Intelligence 1488, Berlin, Heidelberg, New York, pp 251-261

Portinale L, Torasso P (1995) ADAPTER: An Integrated Diagnostic System Combining Case-Based and Abductive Reasoning. In: Veloso M, Aamodt A (eds) Proceedings of ICCBR 95. Springer-Verlag, Lecture Notes in Artificial Intelligence 1010, Berlin, Heidelberg, New York, pp 277-288

Richter MM (2003). Knowledge containers. In: Readings in Case-Based Reasoning. Morgan Kaufmann Publishers

Schank RC (1982) Dynamic memory. A theory of reminding and learning in computers and people. Cambridge University Press, Cambridge

Schmidt R, Gierl L (1998) Experiences with Prototype Designs and Retrieval Methods in Medical Case-Based Reasoning Systems. In: Smyth B, Cunningham P (eds) Proceedings of ECCBR 98. Springer-Verlag, Lecture Notes in Artificial Intelligence 1488, Berlin, Heidelberg, New York, pp 370-381

Stahl A (2005) Learning Similarity Measures: A Formal View Based on a Generalized CBR Model. In: Munoz-Avila H, Ricci F (eds): Proceedings of ICCBR 05. Springer-Verlag, Lecture Notes in Artificial Intelligence 3620, Berlin, Heidelberg, New York, pp 507-521

West GM, McDonald JR (2003) An SQL-Based Approach to Similarity Assessment within a Relational Database. In: Ashley K, Bridge DG (eds) Proceedings of ICCBR 03. Springer-Verlag, Lecture Notes in Artificial Intelligence 2689, Berlin, Heidelberg, New York, pp 610-621

Wilson DC, Leake DB (2001) Maintaining Case-based Reasoners: Dimensions and Directions. Computational Intelligence Journal, Vo. 17, No. 2:196–213

Wiratunga N, Koychev I, Massie S (2004) Feature Selection and Generalisation for Retrieval of Textual Cases. In: Funk P, Gonzàlez Calero P (eds) Proceedings of ECCBR 04. Springer-Verlag, Lecture Notes in Artificial Intelligence 3155, Berlin, Heidelberg, New York, pp 806-820

Wong C, Shiu S, Pal S (2001) Mining fuzzy association rules for web access case adaptation. In *Workshop Proceedings of Soft Computing in Case-Based Reasoning Workshop, Vancouver, Canada*, pp. 213-220

Yang Q, Cheng H (2003) Case Mining from Large Databases. In: Ashley K, Bridge DG (eds) Proceedings of ICCBR 03. Springer-Verlag, Lecture Notes in Artificial Intelligence 2689, Berlin, Heidelberg, New York, pp 691-702

# Why hybrid Case-Based Reasoning will Change the Future of Health Science and Healthcare

Peter Funk

School of Innovation, Design and Engineering,
Mälardalen University, PO Box 883 SE-721 23, Västerås, Sweden
{firstname.lastname}@mdh.se

**Abstract,** The rapid development of the medical field makes it impossible even for experts in the field to keep up with new treatments and experience. Already in 2010 all medical knowledge doubled in 3,5 years, to keep up to date with all development even in a narrow field is today far beyond human capacity. The need for decision support is increasingly important to ensure optimal treatment of patients, especially if patients are not "standard patients" matching a gold standard treatment. By ensuring confidentiality and collecting structured cases on a large scale will enable clinical decision support far beyond what is possible today and will be a major leap in healthcare.

Already in 2010 all medical knowledge doubled every 3.5 years and is expected to double every 7 months in 2020 [1]. 20 years ago physicians met and discussed medical cases over a cup of coffee, an efficient way of sharing experience and disseminating knowledge. Times are changing; physicians say they don't have time for this any more. In a modern and efficient healthcare organisation there is no longer room for experience sharing and patients are treated according to guidelines. Many physicians I have discussed with admit that the consequence is that as much as 30% of patients don't receive optimal treatment. The amount of medical knowledge is already huge, so it often takes years for new results to spread and even specialists are not able to keep up to date with all developments in their own area. Also some physicians mentioned the use of "golden standard" having the consequence that not all patients get an optimal treatment on an individual level [2]. To illustrate this situation Fig. 1 shows what some physicians see as a problem.

The need for more individualized treatment is recognized today, but to make this come true is not easy for a number of reasons, one suggested reason given by a physician is the lack of support in hospitals for individualized treatments "No one questions your actions if you follow a *gold standard* and something goes wrong, but if you divert from it and something goes wrong, you are in a difficult situation". Sharing experience on patients

that do not fit the standard treatment is essential in order to reach a higher degree of individualization. And it is not always possible to wait for evidence. A valuable ability in humans is that we are able to learn from anecdotal cases and improve performance.

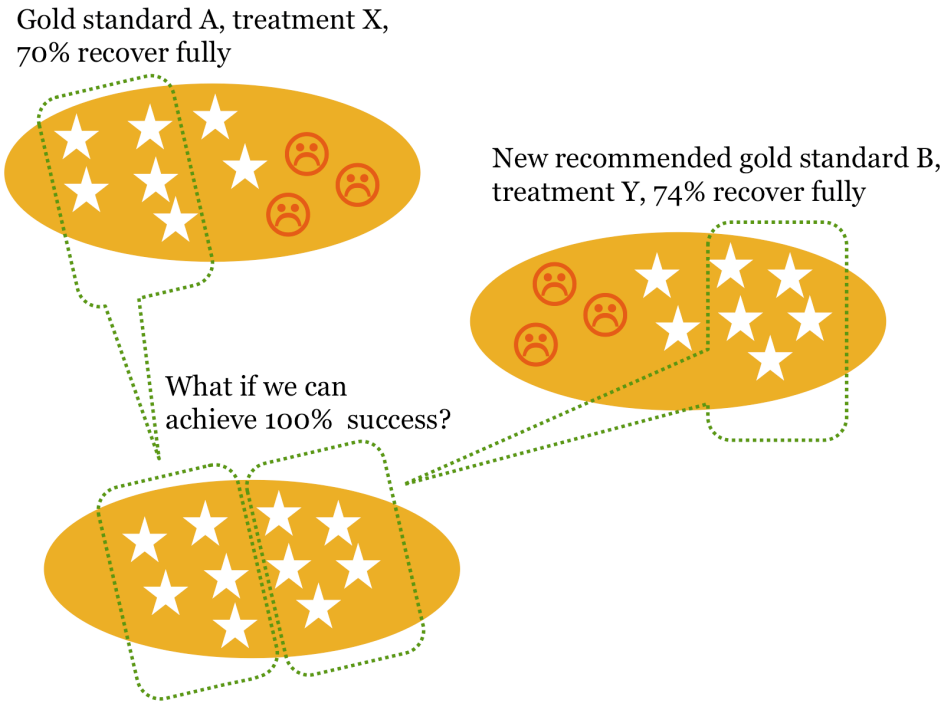


**Fig 1.** If treatment Y is better than treatment X, then it may be tempting to make treatment Y to a recommended gold standard. But what about the 26% which don't get the best treatment? If we can identify which individuals respond best on X and which respond best to Y, we are able to give every patient their optimal treatment.

Key problems to improve with high relevance in the medical area:

- Limited time to share experience among clinicians/physicians.
- Limited time to acquire relevant knowledge/experience related to patients
- Keeping up with all new medical knowledge
- Dissemination of new knowledge and experience at the point of need
- How to individualise treatment of patients so all get an optimal treatment

## 1. What can Case-based reasoning offer?

Imagine you have a patient in front of you, the CBR system immediately says "your patient's symptoms are very similar to 47 other patients in Europe, where there are 4 different treatments, patients with treatment C recovered within 2 weeks, twice as fast as with treatment A and B, there is no difference in treatment cost. Based on my experience (all my cases) a modification of treatment C is recommended (due to your patient having diabetes). In France there is an alternative treatment D (8 patients) with recovery time of less than 10 days, the cost for this treatment is 5 times higher". The system offers

- Advice at the point of care tailored for the patient and physician
- Dissemination of experience from new treatments/procedures
- Second opinion for an experienced clinician, transfer experience to a less experienced clinicians
- It can explain and justify all its conclusions and findings

We can provide all this with CBR and I cannot see how this can be solved without case-based clinical decision support systems. All the different foundational methods and techniques are already available in research, to mention some [3,4,5], but to achieve a transformation of the healthcare system we need a large scale approach since it requires a change in how patient cases are recorded and stored in order to preserve privacy and enable experience reuse.

To achieve this we need more elaborate case structures enabling hybrid case-based reasoning including experience sharing, knowledge discovery, data mining. Many approaches also address distributed knowledge sources [8] and under uncertainty [9] and case-based reasoning theory is today increasingly diverse and advanced able to address challenges preciously difficult to solve [10] and there is progress in integrating electronic patient record system with CBR [11]. One approach developed for medical application used in the Pain-Out project [5,6] is a two-layered case structure, see Fig. 2.

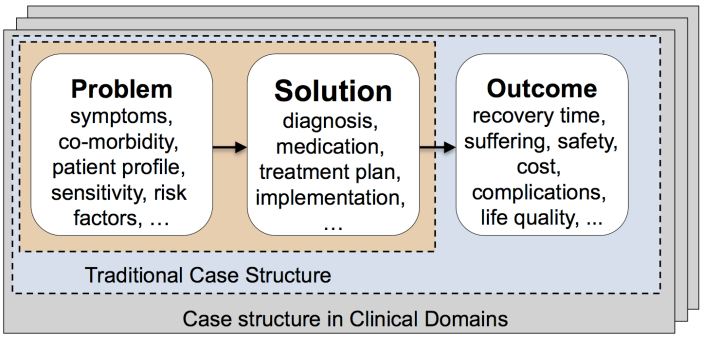


**Fig 2.** Extended case structure used in clinical application [3]

When explaining the concept of case-based reasoning for clinicians, the response is often "such a tool would dramatically change and improve my work and healthcare":

- Patient records become sources of experience and knowledge and provide supplementary information not currently accessible for diagnosis and treatment by clinicians at the point of care
- Clinicians will be able to easily and instantly share experience around specific case issues
- Dissemination of new clinical experience will be efficient and at the point of need.
- Patients will receive personalised and more informed diagnosis and care.

## 2. Example case

One project where we explored some of the issues is in the PAIN-OUT decision support tool. With over 40,000 cases as our "experience base" we developed a tool, giving clinicians relevant information specifically compiled for the patient at hand (comorbidity, age, weight and other factors taken into account). Similar patients are identified amongst the cases and the treatment and outcome is analyzed and presented.


**Fig 3.** Example of a clinical decision support tool that provides physician with personalised information tailored for the patient at hand.

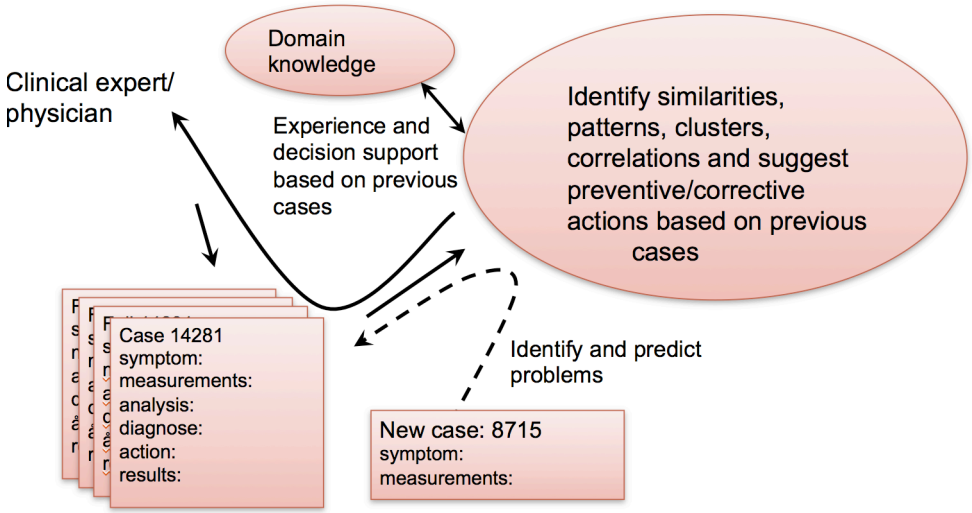


**Fig 4.** Example of how medical cases can be used to support clinicians.

## 3. Conclusions

We have summarized some important issues where case-based decision support can help.

Clinical case based reasoning enables:

- second opinion for an experienced clinician
- dissemination of experience from new treatments/procedures
- transfer experience to a less experienced clinician
- link to relevant research and clinical studies
- other clinicians experience (annotated cases)

The requirements are that cases are collected where symptoms, diagnosis and outcome of treatment is recorded. In many medical registries this is unfortunately not available, especially the outcome is rarely recorded and it is often difficult or impossible to reconstruct the cases.

## Reference


[1] Peter Densen, MD. Challenges and Opportunities Facing Medical Education. Trans Am Clin Climatol Assoc. 2011; 122: 48–58.PMCID: PMC3116346

[2] Timmermans, Stefan, and Marc Berg. *The gold standard: The challenge of evidence-based medicine and standardization in health care*. Temple University Press, 2010.

[3] Begum, Shahina, et al. "Case-based reasoning systems in the health sciences: a survey of recent trends and developments." *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 41.4 (2011): 421-434.

[4] C Marling, S Montani, I Bichindaritz, P Funk, Synergistic case-based reasoning in medical domains Expert systems with applications, 41. 2 (2014): 249-259.

[5] Ahmed M.U., Funk P., A Computer Aided System for Post-operative Pain Treatment Combining Knowledge Discovery and Case-Based Reasoning, In *Case-Based Reasoning Research and Development*, pp. 3-16. Springer Berlin Heidelberg, 2012.

[6] Rothaug et. al, Patients' perception of postoperative pain management: Validation of the International Pain Outcomes (IPO) Questionnaire, The Journal of Pain 14.11 (2013): 1361-1370, Churchill Livingstone.

[7] Ahmed, Mobyen Uddin, and Peter Funk. "Mining rare cases in post-operative pain by means of outlier detection." Signal Processing and Information Technology (ISSPIT), 2011 IEEE International Symposium on. IEEE, 2011.

[8] Reichle, Meike, Kerstin Bach, and Klaus-Dieter Althoff. "Knowledge engineering within the application-independent architecture SEASALT." *International Journal of Knowledge Engineering and Data Mining* 1.3 (2010): 202-215.

[9] Bruland, Tore, Agnar Aamodt, and Helge Langseth. "Architectures integrating case-based reasoning and bayesian networks for clinical decision support." *Intelligent Information Processing V*. Springer Berlin Heidelberg, 2010. 82-91.

[10] Richter, Michael M., and Rosina O. Weber. "Case-Based Reasoning." *A Textbook* (2013). ISBN 978-3-642-40166-4, Springer Verlag.

[11] van den Branden, M., Wiratunga, N., Burton, D., & Craw, S. (2011). Integrating case-based reasoning with an electronic patient record system. *Artificial Intelligence in Medicine*, *51*(2), 117-123.