

A NEW APPROACH TO APPLICABLE MEMORY-BASED REASONING.

Ketil Bø

*Dynamic Imaging AS
P.O. Box 2865
N-7002 Trondheim, Norway*

Agnar Aamodt

*Norwegian University of Science and Technology,
Department of Informatics
N-7034 Trondheim, Norway*

Abstract

To appropriately utilize the rapidly growing amount of data and information is a big challenge for people and organizations. Standard information retrieval methods, using sequential processing combined with syntax-based indexing and access methods, have not been able to adequately handle this problem. We are currently investigating a different approach, based on a combination of massive parallel processing with case-based (memory-based) reasoning methods. Given the problems of purely syntax-based retrieval methods, we suggest ways of incorporating general domain knowledge into memory-based reasoning. Our approach is related to the properties of the parallel processing microchip MS160, particularly targeted at fast information retrieval from very large data sets. Within this framework different memory-based methods are studied, differing in the type and representation of cases, and in the way that the retrieval methods are supported by explicit general domain knowledge. Cases can be explicitly stored information retrieval episodes, virtually stored abstractions linked to document records, or merely the document records themselves. General domain knowledge can be a multi-relational semantic network, a set of term dependencies and relevances, or compiled into a global similarity metric. This paper presents the general framework, discusses the core issues involved, and describes three different methods illustrated by examples from the domain of medical diagnosis.

1. Introduction

More and more people find it difficult to cope with the rapidly growing amount of information continually produced and distributed. An increasing amount of information that could have been useful is never used, at least not at the right place and time. In companies and other corporations huge bodies of data and information are stored as the result of data logging and general information gathering. To an increasing degree, the contents of such stores are kept in computerized data bases, registers, personnel files, manuals, information banks, etc. They are distributed via local networks or global media (e.g. Internet) as electronic mail, hyper-network browsing systems (e.g. WWW), bibliographical retrieval services, news groups, etc., etc. However, the ease of production and distribution of information that is characteristic of our information society has not been met by a corresponding development of effective methods to help in the appropriate utilization of this information. Improved methods for information capturing - including active filtering and focusing of information, searching for specific information, and guidance on its utilization - are needed in order to benefit from the information 'explosion', rather than drown in it. This is a commonly recognized problem [1, 2]. Computer systems targeted at this problem need to address two main tasks: Finding relevant information for a user, and helping the user utilize the information found. The two tasks are highly interrelated. People want information in order to solve problems or to achieve goals. A human documentalist

or information servant will always try to find information that is useful for the user, i.e. that is *relevant*. A documentalist who is also competent in the topic of the search will also be able to actively filter and organize the information retrieved, pose new or modified search requests, present the results of the search according to its intended use, and give advice about the possible utilization of the results for the user. An intelligent computerized information servant should be able to do the same.

We can identify two different problems related to the finding and utilization of information. One is how to get hold of potentially useful information on a general basis, the other is how to find relevant information for a particular task at hand. We will refer to these as the *information filtering problem*, and the *information retrieval problem*, respectively. They both need to be addressed, and there are clear interdependencies between them, in the sense that the way one of them is handled, for example within a particular company, will have impact on how the other is handled. In this paper we describe a method that primarily addresses the information retrieval (IR) problem.

Standard methods for information retrieval include direct keyword indexing of documents¹, clustering methods for automatically deriving indexes from frequency analyses [3, 4], as well as free text search via inverted files or direct document scanning. These methods are characterised by a weak notion of search term relevancy, and no ability to reason about a problem situation in order to improve the retrieval. Case-based reasoning (CBR) methods, on the other hand, are knowledge-based methods designed to reason about a particular problem, and to retrieve a matching case from a "case base" of past problem-solving episodes. In this process a CBR method uses a similarity metric or some other means of similarity assessment, to retrieve cases that best matches the problem descriptors. Further, by retaining a problem just solved, a CBR system is able to continually learn by experience, and therefore adapt to its environment. However, a CBR system relies on a set of cases deliberately represented and stored in a case base, which is a limitation compared to, for example, a full-text IR system. Hence we have two technologies that each has its strengths and limitations (see also [5]). This calls for an investigation into ways of combining them.

CBR methods cover a wide scope of particular approaches, ranging from knowledge-intensive methods that require a substantial domain knowledge base to support the case-based processes, to methods where such knowledge is compiled into global, static similarity metrics. Some methods also utilize parallelism to gain performance efficiency. The research reported here describes a parallel processing approach to case-based reasoning - often referred to as *memory-based reasoning* (MBR) [6]. MBR has up to now focused on exploring parallelism on large amounts of raw data, focusing on the "brute force" capabilities of massively parallel processing. Limitations of these approaches have been pointed out by several authors, identifying the needs for more semantic oriented, knowledge-based, approaches [7, 14].

We are exploring different ways of combining parallelism and MBR with some degree of knowledge-based support, for the IR task. There are several ways of combining the two, where two extremes are a high degree of brute force parallelism and low emphasis on the explicit modeling of domain knowledge, on the one hand, and a high utilization of explicit knowledge models in the reasoning from past situations and a somewhat lower emphasis on the raw power of parallelism, on the other.

Cases may be used in two fundamentally different ways in combined MBR+IR systems. The case base may either be a separate, additional store of past problem solving episodes (here:

¹The term document is used in a general sense, as any information containing item.

information retrieval episodes), or it may simply be the collection of the documents themselves. We present a framework that covers both these approaches, and a set of methods that starts in the syntactic and rather simple end and ends in the more complex semantic one. While standard IR methods combine sequential processing with syntax-driven retrieval techniques, we are exploring a radically different approach: Combining massive parallelism with semantical retrieval methods, based on the meaning of terms within the context of the search. Our goal is to develop and appropriately integrate these methods into an advanced tool for information filtering and retrieval, combining knowledge-based and parallel processing techniques.

In the next section we give a brief overview of the areas of case-based reasoning, information retrieval and parallel processing, relevant to the rest of the paper. The MS160 microchip [8] is central to our tool, and its architecture and accompanying data retrieval mechanism is described in a separate subsection. In section 3 we discuss the main issues of case-based reasoning that need to be dealt with in our context and present a memory-based reasoning approach based on the MS160 architecture. The concluding section summarizes the paper and points to our continuing research.

2. Methodological Basis

2.1. Case-based reasoning

Case-based reasoning is a broad term, covering many particular types of methods. The main method types are summarized in the following, distinguished by their different solutions to core CBR problems such as case representation, integration of general knowledge, reasoning, and learning.

Typical case-based reasoning combines cases with general domain knowledge of some kind. The notion of a case, i.e. what constitutes a case, differs from more syntax-oriented methods, as described below. For example, a typical case has a certain degree of richness of information, and a certain complexity with respect to its internal organization. This is CBR in the original, paradigmatic sense [9, 10], and where most of CBR research is taking place. These type of CBR methods are generally able to modify, or adapt, a retrieved solution when applied in a different problem solving context.

Analogy-based reasoning is sometimes used as a synonym to the typical case-based approach just described [10], but most often [11] as a characterization of methods that solve new problems based on past cases from a *different domain*. The major focus of study has been on the *reuse* of a past case, what is called the mapping problem, i.e. to map the solution of an identified analogue (called source or base) to the present problem (called target).

Instance-based reasoning is a highly syntactic CBR-approach, which does not utilize explicit domain knowledge. The lack of guidance from general domain knowledge is compensated for by a large number of instances. This is a non-generalization approach to the concept learning problem addressed by classical, inductive machine learning methods [13].

Memory-based reasoning focuses memory organization and access within a large memory of cases. The utilization of *parallel processing* techniques is characteristic, and distinguishes this approach from the others. The access and storage methods may rely on purely syntactic criteria, as in the MBR-Talk system [7], or they may attempt to utilize general domain knowledge, as the work done in Japan on massive parallel memories [14].

In this paper we describe a method - or set of methods - that to various degrees combine the

memory-based and the more typical CBR approaches. The methods are based on a common framework, with two parts. One - the general CBR framework [15] - is summarized below. The other - a common integration framework - is the topic of chapter 3.

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

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

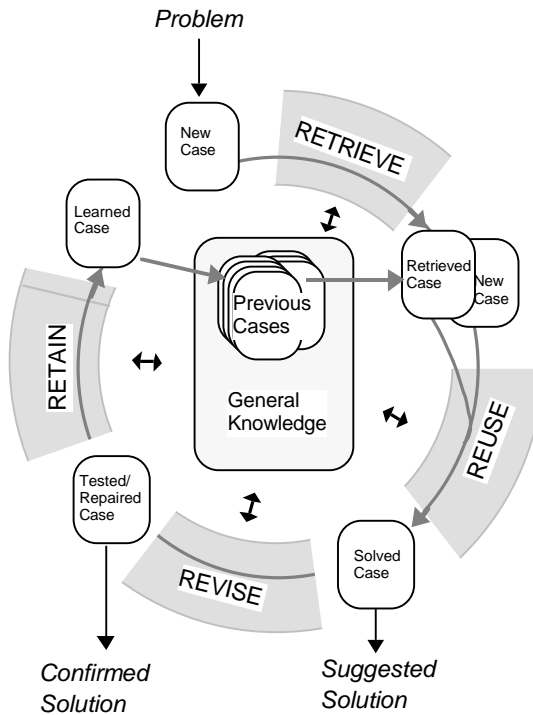


Figure 1. The CBR cycle

The cycle is illustrated in Figure 1. An initial description of a problem (top of figure) defines a new case. This new case is used to RETRIEVE a case from the collection of previous cases. While some case-based approaches retrieve a previous case largely based on superficial, syntactical similarities among problem descriptors (e.g. the CYRUS system [16], one of the first CBR systems), some approaches attempt to retrieve cases based on features that have deeper, semantical similarities (e.g. the PROTOS [17] and CREEK [18] systems). Syntactic similarity assessment - sometimes referred to as a "knowledge-poor" approach - has its advantage in domains where general domain knowledge is very difficult or impossible to acquire beforehand. Semantically oriented approaches on the other hand, often referred to as "knowledge-intensive"², are able to use the contextual meaning of a problem description in its matching, for domains where general domain knowledge is available.

The retrieved case is combined with the new case into a solved case, i.e. a proposed solution to the initial problem. REUSE of the retrieved case solution in the context of the new case focuses on

²Note that syntactic oriented methods may also contain a lot of general domain knowledge, implicit in their matching methods. The distinction between knowledge-poor and knowledge-intensive is therefore related to *explicitly represented* domain knowledge. Further, it refers to *generalized* domain knowledge, since cases also contain explicit knowledge, but this is *specialized* (specific) domain knowledge.

two aspects: (a) the differences among the past and the current case and (b) what part of a retrieved case can be transferred to the new case. In simple classification tasks the differences are abstracted away and the solution class of the retrieved case is transferred to the new case as its solution class. This is a trivial type of reuse. More typically, systems have to take into account differences in (a) and thus the reused part (b) cannot be directly transferred to the new case but requires an adaptation process that takes into account those differences. There are two main ways to reuse past cases: Reuse of the past case solution (transformational reuse), and reuse of the past method that constructed the solution (derivational reuse) [10].

Through the REVISE process this solution is tested for success, e.g. by being applied to the real world environment or evaluated by a teacher, and repaired if failed. Case revision consists of two tasks: Evaluate the case solution generated by reuse - and if successful, learn from the success. Otherwise repair the case solution using domain-specific knowledge. The evaluation task takes the result from applying the solution in the real environment (asking a teacher or performing the task in the real world). This is usually a step outside the CBR system, since it - at least for a system in normal operation - involves the application of a suggested solution to the real problem. The results from applying the solution may take some time to appear, depending on the type of application. In a medical decision support system, the success or failure of a treatment may take from a few hours up to several months. The case may still be learned, and be available in the case base in the intermediate period, but it has to be marked as a non-evaluated case.

During RETAIN, the system learns from its current problem solving experience, and the case base is updated by including the new case, or by modification of existing cases. This involves selecting what information from the experience to retain, in what form it should be retained, whether a new case should be constructed, how a new case should be indexed for later retrieval, and how it should be integrated in the memory structure and knowledge base in general. A new case may be built, or the old case may be generalized or strengthened to subsume the present case as well. If the problem was solved by other methods, including asking the user, a new case is constructed. The 'indexing problem' is a central and much focused problem in case-based reasoning. It amounts to deciding what type of indexes to use for future retrieval, and how to structure the search space of indexes. With the MS160 parallel search technology, as we will see, the "indexing problem" may be radically reduced or disappear altogether.

2.2. *Information retrieval*

The majority of information in databases is in the form of text, and even if the amount of digitized images, video and audio is growing, text will continue to be the dominant medium for information retrieval (IR). Much of IR research has concentrated on different representation techniques and retrieval algorithms. The document representation process in IR is commonly referred to as indexing, and the retrieval algorithms usually constitute a comparison process. However, if the query and an indexed document are to be compared, they must be described using the same view of the world.

If we assume that an information problem is stated in near natural language, the corresponding representation is the raw text itself. Another significant observation is that human IR is not a one-shot process, but a dynamic process of homing in on the target. These facts call for very fast searching mechanisms directly on the raw text itself.

Besides the direct matching, most IR systems has traditionally used statistical approach for the retrieval process. The knowledge based approach is less well understood even if natural language processing, knowledge-directed search, and learning techniques has been used with some success during the past few years. A comparison of statistical and knowledge based IR is given in the

following table [24]:

STATISTICAL	KNOWLEDGE-BASED
Text representation based on counts of single words	Text representation based on syntactic and semantic analysis
Retrieval based on a similarity function	Retrieval based on inference
Domain independent	Potentially domain knowledge bases
Feedback based on statistical models	Learning based on symbolic and connectionist models
User independent	Models of individuals and classes of users.
Efficient	Currently expensive to implement
Effective	Effective in narrow/specific domains

The typical effectiveness measure in Information Retrieval is the ratio between *precision* and *recall*. Precision expresses the coherence on the result of the searching and what is searched for. Recall express to what extent everything searched for is found. The problem is that we normally will be able to get either high precision or low recall, or the other way around, but usually not both at the same time. Research has been done for years, without any good solution to be found for existing hardware. Combining the MS160 with CBR may provide a solution to this problem. The MS160 microchip is particularly designed for fast retrieval of patterns from text, incorporating fuzzy matching. The MS160 technology opens up for a combination of statistical- and knowledge based methods to improve effectiveness in information retrieval.

2.3. Massive parallel processing

The history of massively parallel computers can be traced back to Illiac-IV [19], DAP [20], and MPP [21]. However the break through came with the development of the Connection Machine [22]. The currently available Connection Machine system is a parallel SIMD computer with up to 65.536 processing elements, each having 4.096 bits of memory and a 1 bit wide ALU. Every processor executes the same program, though processors can selectively ignore program steps. Processors are connected to each other via a hypercube based routing network.

The von Neumann machine architecture and the availability and price of large memory has, in the past, been the major obstacles to high-volume case-based reasoning. On a sequential machine, both the cost of the hardware and the time required for processing grows linearly with the size of the data base. One solution is to use a vector machine. However, unless we allow the number of processors utilized to grow as the size of the data base increases, we get no more than a constant speedup over sequential machines. In order to make memory based reasoning applicable, it seems that a paradigm shift in massively parallel searching technology is needed. The MS160 represents a different parallel processing architecture, and a particularly promising one for memory-intensive case-based information retrieval.

A Memory-Based Reasoning system, MBRtalk, has been implemented on the Connection Machine [7] and reports promising results. The cost of the hardware is $O(n \log n)$ in the size of the data base, while the processing time is $O(\log^2 n)$. However, the MBRtalk paper also points out several weaknesses of the approach, including the problems of incorporating explicit general domain knowledge to guide the search. Our particular MBR approach is related to the properties of the MS160 parallel-processing microchip, jointly developed by MRT micro AS and the University of Trondheim [23]. The MS160 chip is an affordable PC extension optimized for fast,

parallel search of long text strings or bit patterns.

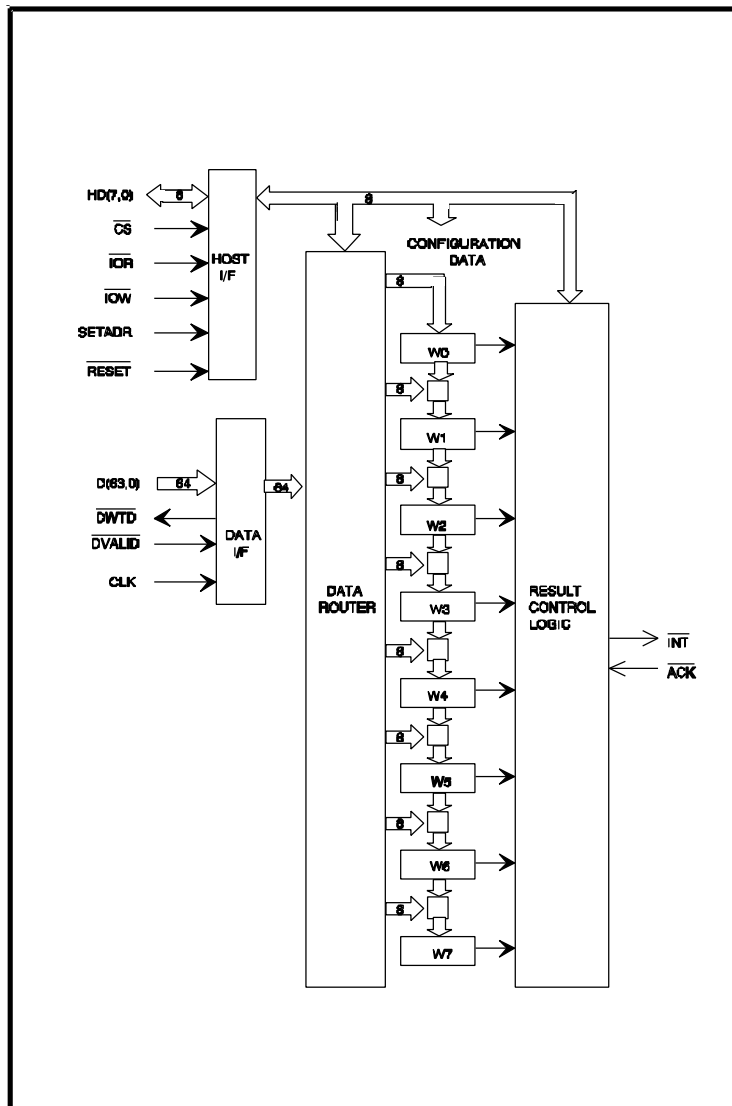


Figure 2. MS160 Block diagram

With the dramatic 40% memory capacity increase a year, the corresponding decrease of prices and the introduction of MS160, a chip for massively parallel searching in large data volumes, the technology to implement applicable memory based reasoning seems now to be within reach.

The main difference of the MS160 technology, in the context of IR applications, compared to the Connection Machine is that it is tailor made for ultra fast pattern matching (Compares 8 bytes in parallel per clock cycle), its capability to perform “fuzzy” search, and that it is available on standard PC at an acceptable price.

2.4. The MS160 Non Numeric co-processor

MS160 is a massively parallel, full custom VLSI chip with the capability of searching through 160 MB of data per second. This 400,000 transistor device uses a 64-bit wide input bus. When applied to the problem of performing a fuzzy search on data stored in memory, it provides orders of magnitude in increased speed compared to the equivalent throughput of a Pentium PC.

MS160 contains 512 processing elements (PE) set up to work as 8-bit comparators. At 20 MHz, it can perform 10 billion comparisons per second while scanning through 160 megabytes of data.

Eight so-called "blocks of eyes" in MS160 inspect the data streaming through the chip, each block (window) observing continuously 32 data elements of 1 byte. Each primitive eye has two match elements with programmable upper and lower match limits which not only makes it possible to find exact hits, but also more "fuzzy" candidates when applicable. When a hit is encountered, it signals a window match to the Windows Match Logic. The Windows Match Logic is programmed to signal a global hit when a predefined combination of windows has reported hits within a given distance.

Each window will be set-up for remembering a hit for a certain time. This will open up for the possibility of context sensitive searching. The windows may be grouped for more complex hit conditions. Data may be routed to the windows in many ways; 8 data streams to the 8 windows in parallel, 4 data streams connected in parallel or serial to two windows each, one data stream to one super window, etc. The chip has several programmable modes:

1. REPORT/COUNT mode.

When the chip is in REPORT mode, it stores the address of the hit each time a hit occurs. In COUNT mode, the chip increases an internal counter by one for each hit. For example, it is possible to find how many times the word "charity" occurs in the Bible in a fraction of a second.

2. STOP/CONTINUE mode.

When the chip is in STOP mode, the chip stops the data flow until acknowledgement has been given. In CONTINUE mode, the chip stores the hit address internally and continues the search.

3. FLANK/HIT mode.

When the chip is in FLANK mode it will generate only one hit for subsequent hits (useful in image processing etc.), in HIT mode, the chip will generate one hit for every occurrence.

Each of the 32 bytes wide windows works by comparing the data stream, byte by byte, with a template. At each location, a specific byte or range of bytes can be selected. For example, to search for the word "find", one would specify its letters precisely. To specify any word that contains four letters, the last of which are 'ind', one would use a wild card in the first location and the literal 'ind' in the next three. This request would convert to an MS160 template that accept the letter a...z for the first letter, "i" for the second, etc. MS160 may also look for ranges such as "[91-94]" or combinations of words and ranges such as looking for all medical cases containing inflammation and fever in the range [39-41] C in the time period June-July 19"[91-94]".

Up to now, almost all searching in large data volumes have been based on the technique of indexing the data. Indexes offer fast access to indexed data, but the storage space increase dramatically and advanced or "unexpected" queries are slow, or not possible at all. The MS160 offers a new fast way to deal with information retrieval in an object oriented way with none or limited use of indexes.

Originally intended to do text retrieval on massive document volumes for government agencies, law offices, and libraries, it has recently been discovered that MS160 can be applied to all sorts of very interesting scientific and engineering applications. This includes image processing, MBR, searching for fingerprints, gene pattern matching in chromosomes ("Find all GGT??ACGACG??TCG??CAT??GACACT patterns"), medical diagnosis, or molecules that have nearly identical spectra to the one just observed.

3. Memory-based reasoning with the MS160

3.1. A framework for extended memory-based reasoning.

The scope of the framework is the incorporation of general domain knowledge into parallel-processing MBR. Figure 3 illustrates the main components in an MBR system, i.e. the retrieval method, the knowledge base, and the case base.

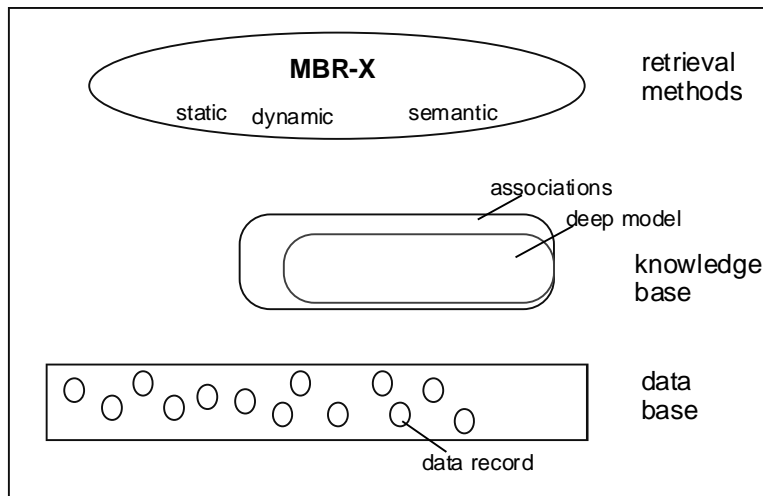


Figure 3. Basic framework components

The *retrieval method* may be based on a syntactic-oriented or a semantic similarity assessment, corresponding to whether similarity is evaluated on the basis of equality of the descriptor terms themselves (syntactic), or on the basis of the meaning of these terms (semantic). In the latter case, general domain knowledge is used to make the necessary transformations between virtually dissimilar descriptors. The syntactic-oriented methods are split into methods based on - on the one side - a static and global similarity metric (see Figure 3), i.e. a similarity metric common to all types of cases, and - on the other side - methods based on a dynamic similarity metric, i.e. a metric that captures local variations between types of cases. In the latter case, different attributes may have different degrees of relevance depending on the type of cases involved. The *knowledge base* contains general domain knowledge in explicitly represented form, either associations in the form of rules and other dependencies, or deeper knowledge models that for example capture taxonomic, causal, and functional relationships between the concepts corresponding to case descriptors. The *data base* is the store of cases.

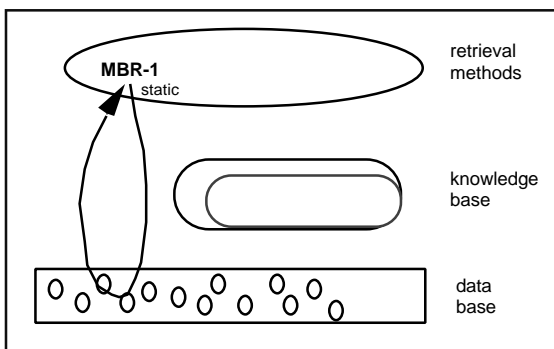
3.1. Case retrieval

Case retrieval is the most important process in memory based reasoning. All the following stages are dependent on the ability to retrieve the best possible cases from memory. As earlier mentioned, the retrieval process may be based on superficial and syntactic criteria as well as deeper, semantically and contextual criteria. We are investigating three possible approaches, moving from purely syntactic towards more knowledge-intensive. Since the core of the retrieval step is the similarity assessment between a problem description and the set of cases, the three approaches can be characterized in terms of their similarity assessment procedures.

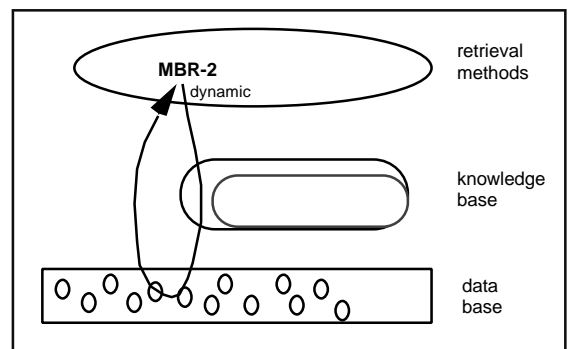
Most CBR systems use indexes in one way or another for case retrieval, but there are a lot of unresolved problems here. Research issues include what type of indexing terms to use, the actual index vocabulary, the way indexes are linked to cases, possible inter-case indexes, whether indexes are direct pointers to cases or parts of an index structure, the combination of indexes during retrieval, and the assessment criteria for case similarity. A problem with too heavy reliance on indexing, however, is that indexes are a kind of pre-compiled knowledge. A characteristic of case-based learning is that the generalization process is not made when learned knowledge is

stored, but when this knowledge is used, i.e. during problem solving. Indexing works in the opposite direction, since it anticipates and pre-sets the future use of case knowledge. The alternative approach to case retrieval is search, which on serial machines is time consuming, and often difficult to guide in the wanted direction. Parallel hardware changes this picture. The search process may be syntax-oriented, or it may be a knowledge-based, semantic-oriented process, where the cases are embedded in a model of general domain knowledge (e.g. a set of rules or a semantic network). But elaborate reasoning within an extensive and rather deep model of general knowledge, is a usually cost demanding process, and also a difficult one to realise in a parallel architecture. The approaches that have been taken for utilizing parallelism in case retrieval has therefore largely been syntax-oriented (e.g. [6]). Based on our previous work in knowledge-intensive CBR on serial machines [18], we extend our method to cases where the semantical context of a case-description is utilized to improve retrieval quality.

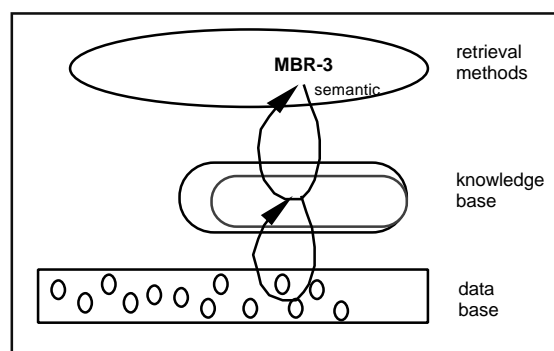
The three types of retrieval methods we are studying are a global (static) similarity metric, a local (dynamic) metric, and a knowledge-based matching process. The use (or non-use) of explicit domain knowledge by the three methods is illustrated in Figure 4. It is seen that the method involving local similarity metrics may make use of shallow, associational domain knowledge in its similarity assessment. It therefore spans the gap between a purely syntactical and a more knowledge-intensive retrieval.



a) Global similarity metric - purely syntactical



b) Local similarity metric - associational knowledge



c) Knowledge-driven similarity assessment - deep model knowledge

Figure 4. Methods of memory-based information retrieval

The rest of this chapter describes each of the three retrieval approaches in more depth. For each approach we outline a particular CBR method that takes advantage of the MS160 properties. The two extreme approaches, the global metric and the explicit model of general domain knowledge, respectively, are described in more detail than the 'intermediate' local metric approach.

3.1. Building a case base

Our example domain is medical diagnosis, i.e. the case-based retrieval and reuse of previous patient cases in diagnosis. The easiest way to establish a case base is to take the raw patient data as recorded by the doctor in direct interaction with the patient, add the confirmed diagnosis for the case which together make an ordered pair in the case base. Because the MS160 is capable of searching the raw cases without any structuring and indexing, the raw data representation is well suited for memory based reasoning. Instead of solving a new problem directly, the case base is employed in order to use solutions from earlier problems. Once retrieval has been achieved, the rest of the CBR cycle can be completed via interaction with specialist knowledge based systems or with human experts. In the following subchapters, the three types of retrieval methods are discussed with respect to parallelisation on the MS160.

An example of a patient record case is shown below. The patient's symptoms etc. are listed, as is also the diagnosis. Attribute-value pairs are enclosed in parentheses. This case is taken from the collection of patient cases analysed in [25].

```
patient#15
instance-of      value patient
has-history      value (age . 69) (sex . male) (height . 166) (weight . 98)
                  (personal-health-status . used-bronchodilators-and-antibiotics-
                    for-16-years)
                  (family-disposition) (way-of-living . former-shipyard-worker)
                  (way-of-living . smoking-15-cigarettes-each-day)
has-symptoms     value (dyspnea . yes) (cough . yes) (wheeze . frequent) (other-complaints)
has-signs        value (clubbing-and-cyanosis) (sputum . sever-production) (respiratory-
                    rate . 64) (blood-pressure- quote (71/36 . 26/11))
has-lab-findings value (fvc . 2.15) (irv . 1.89) (tlc . 7.32) (fev . 1.3) (fev/fvc . 60)
                    (mvv . 44) (pao2- quote (73 . 87)) (pco2 . 5) (ph . 7.46)
has-diagnosis    value moderate-chronic-bronchitis
```

A case describes the problem features, in the example listed by feature types (history, symptom, etc.), and solution (diagnosis). A feature consists of a feature name (e.g. *age*) and a feature value (e.g. 69), or simply a single feature term (*family-disposition*), assumed to be true. In addition to these type of values (identified by the *value* facet in the example case), cases may also contain other types of information, such as textual comments, possible value types, uncertainty of values, etc.

Currently, the type of cases we are studying represents actual information sources, e.g. patient records. On our agenda is also the investigation of "search session cases", i.e. cases which contents are *information retrieval episodes*. These cases are indexed by the retrieval situation (goal, user-type, initial query, etc.) and contains pointers to the actual information records that were successfully retrieved in the past.

3.2. Global, static similarity metric.

This is the simplest form of similarity assessment. All problem features are treated alike, and there is no explicit notion of feature relevancy or other general domain knowledge. The part of the case focused forms an ordered pair, case=(problem, solution), i.e. $c=(p,s)$. Parallellizing the retrieval process is in principle trivial, since each case is an independent source of information, and since the method makes no use of a shared structure. This is shown in Figure 4, illustrating the parallelization of this method on the relevant part of Figure 3. No knowledge base is involved. Our method utilizes the power of massively parallel search to construct a relevance measure during search (see also [6]). The problem is to retrieve a suitable case $c' = (p',s')$ in the case base and then transfer the solution s' of p' in order to get a solution s of p . This is broken down into four basic operations.

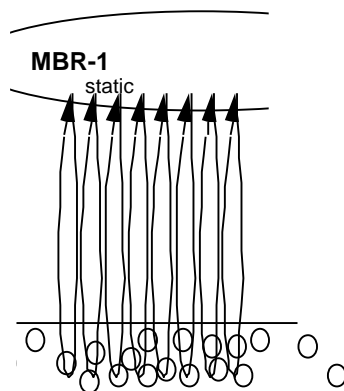


Figure 5. Parallel retrieval with global similarity metric

The first is counting the number of times various features or combination of features occur. In medical applications, it may be interesting to find how often high fever occurred in combination with various diseases. MS160 is capable of counting numbers of times a value lays within a given interval with a speed of up to 160Mb/sec. when the cases are located in memory. One can, for instance, ask how many times fever in the range of 39- 42 C occur in connection with inflammation of the throat.

The second operation is to use the features count to produce a similarity metric. Here we follow the idea developed for the Patdex-1 system [26]. Given p one selects $c' = (p', s')$ such that p' is most similar to p with respect to the case base. In a diagnostic problem solving context, the problem is the symptoms and the solution is the diagnostics $C=(Sym, Diag)$. We assume that the diagnostics have been verified for the recorded cases.

There are three basic outcomes:

1. Sym did not have sufficient many symptom values to determine Diag, but one has a good guess.
2. Sym did determine Diag, but contained redundant information, i.e. unnecessary symptom values
3. Sym did determine Diag, and no smaller set of symptoms would do so.

Since we do not know which possibility has occurred we have to take care of all of them, which is a requirement to the similarity measure. A suitable similarity measure for this approach may be expressed as:

$$Sim(Sym1, Sym2) = \frac{a \text{ count}(E) + b \text{ count}(C) + c \text{ count}(V) + d \text{ count}(W)}{\text{count}(E \cup C \cup V \cup W)}$$

where a, b, c and d are real numbers and

- E:= set of symptoms with the same values, within fuzziness range, for Sym1 and Sym2
- C:= set of symptoms with different values, outside fuzziness range, for Sym1 and Sym2
- V:= set of symptoms in Sym1 but not in Sym2
- W:= set of symptoms in Sym2 but not in Sym1.

After the similarity metric has been computed, the third step is to calculate the dissimilarity between each item in memory and the current case. The fourth step is retrieving the best matches

Since the searching capabilities in MS160 is extremely fast, it is possible to extract the key symptoms in real time from the record as it is written by the doctor in interaction with the patient

and display the number of hits, E, as soon as a new key symptom is entered. If the number E is small it is possible to add a synonym dictionary and automatically search also the synonyms because the speed of searching with MS160, even in very large case bases is almost instantaneous in comparison with the speed of writing. When the number of candidates are at an acceptable level, the doctor can stop writing and ask for the similarity measures. There are several possible outcomes:

1. No patient is sufficiently similar to the current case to make a diagnosis.
2. A small number of patients are retrieved.
3. A significant number of patient are retrieved and have similar diagnosis and
4. There are several diagnosis among the patient with highest similarity score.

If the first case holds, the system knows that it has never seen this set of symptoms before. In the second case, even with one patient similar, the system may be able to make a tentative diagnosis. If the third case is true, it is very likely that the patient should get the same diagnosis. Finally if the fourth case is true, then the system knows that it can not come up with a definitive answer without asking for more information such as blood tests etc.

3.3. Local, dynamic similarity metric.

This approach differs from the above in that it allows explicit relevance weights, or other associational knowledge, to be attached to each problem feature. In our approach this is represented as a table of relevance strengths associating particular symptoms with relevant patient diagnoses, for the retrieval of patient records. Parallelization is achieved by letting each processor use its own copy of this table (see figure 6).

To enable a more dynamic and adaptive similarity measure, a learning method will be incorporated. Based on how successful the retrieval is, symptom-to-diagnosis associations are strengthened or weakened over time. Richter and Wess [26] proposes a method based on neural networks for such learning. Our method, based on the Creek architecture [18], incrementally adjusts similarity threshold values, depending on how successful an index has been in retrieving a useful case.

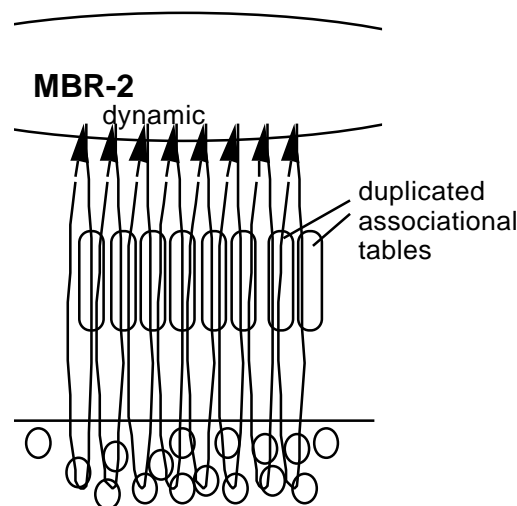


Figure 6. Parallel retrieval with local similarity metric

3.4. Knowledge-driven similarity assessment.

The knowledge-driven similarity assessment method is a way to utilize - to some extent - a body of general domain knowledge as support for case-based retrieval. In earlier work on the CREEK system [18] a knowledge-intensive method for sequential processing was described, referred to as

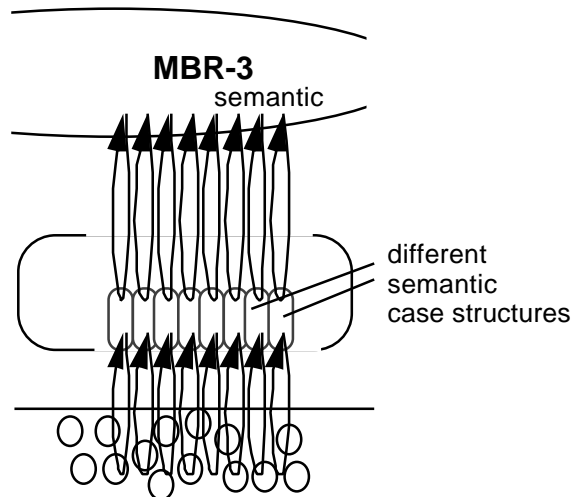


Figure 8. Parallel knowledge-driven retrieval

We are developing a method - partly based on [27], in which a part of the semantical model is kept within each case description. This corresponds to storing the part of the semantic network (Figure 7) which contains concepts and relations of relevance to the particular case, in the case. In a sense, this is similar to a justification of the case solution (i.e. a diagnosis) in terms of an explanation, as described above, but instead of storing a particular explanation, the essential part of the semantic network for which the explanation was generated is kept in the case. The content of a case then becomes a linked structure rather than a list of descriptors. In this structure, the problem descriptors and the solution is linked together with general domain concepts. This is illustrated in figure 8, showing that each case description is embedded in a semantic network of general domain concepts.

4. Concluding remarks

In this paper we have described a framework for memory-based reasoning in information retrieval tasks, in which purely syntactical case matching as well as more knowledge-intensive similarity assessment methods are described and modeled. The syntactical, more or less "brute force", method has been given the most attention in this paper. A more knowledge-based approaches have also been sketched and summarized - and will be stronger emphasized in our further research.

A strong motivation for the work reported here is to utilize the particular properties of the MS160 parallel-processing microchip to move beyond current MBR approaches in enabling simple and affordable tools for information focusing and retrieval. The conclusion of our research so far is that memory-based reasoning utilizing the MS160 Non-Numeric Co-processor offers considerable advantages compared to conventional computing because of the speed and flexibility in finding matches and near matches to specified patterns or strings.

References:

1. Information filtering, *Communication of the ACM*, Vol. 35, no 12., 1992. (Special issue on information filtering.)

2. Digital libraries, *Communication of the ACM*, Vol. 38, no. 4, 1995. (Special issue on digital libraries.)
3. Gerald Salton. Another look at automatic text retrieval systems. *Communications of the ACM*, Vol. 29, no. 7, 1986.
4. C. J. van Risjbergen. *Information retrieval*. Butterworths, 1979.
5. Ralph Barletta. Information retrieval vs. case-based reasoning. *IEEE Expert*, Vol., no., 1994.
6. Craig Stanfill and David Waltz. The memory based reasoning paradigm. In: *Case based reasoning. Proceedings from a workshop*, Clearwater Beach, Florida, May 1988. Morgan Kaufmann Publ.
7. Craig Stanfill and David Waltz. Toward Memory-Based Reasoning, *Communication of the ACM* December 1986 Volume 29 Number 12.
8. Halaas Arne, Kjos Bård. "Designing Information Retrieval Systems using the MS160 Co-processor" Div. of Computer Science and Telematics. The Norwegian Institute of Technology, N-7034. Trondheim Norway. 1992.
9. Roger Schank: *Dynamic memory; a theory of reminding and learning in computers and people*. Cambridge University Press. 1982.
10. Jaime Carbonell: Derivational analogy. In R.S. Michalski, J.G. Carbonell, T.M. Mitchell (eds.): *Machine Learning - An artificial Intelligence Approach*, Vol.2, 1986. Morgan Kaufmann.
11. Janet Kolodner, *Case-based reasoning*. Morgan Kaufmann, 1993.
12. Dedre Gentner: Structure mapping - a theoretical framework for analogy. *Cognitive Science*, Vol.7. s.155-170. 1983.
13. David Aha, Dennis Kibler, and M. K. Albert. Instance-Based Learning Algorithms. *Machine Learning*, vol.6 (1).
14. Hiroaki Kitano. Challenges of Massive Paallelism, *Proceedings of the Thirteens International Conference of Artificial Intelligence*, Chambery, France 1993.
15. Agnar Aamodt, Enric Plaza. Case-Based Reasoning: Foundational issues, methodological variations, and system approaches, *AI Communications*, Vol.7, No.1, March 1994.
16. Janet Kolodner: Reconstructive memory, a computer model. *Cognitive Science*, Vol.7, 1983.
17. Ray Bareiss. *Exemplar-based knowledge acquisition: A unified approach to concept representation, classification, and learning*. Boston, Academic Press, 1989.
18. Agnar Aamodt. Explanation-driven case-based reasoning: In S.Wess, K.Althoff, M.Richter (eds.), *Topics in case-based reasoning*. Springer 1994.
19. Bouknight, W.,J., et al., "The ILLIAC IV System" Proceedings of the IEEE, April 1972.
20. Bowler, K.C. and Pawley G. S., "Mmolecular Dynamics and Monte Carlo Simulation in Solid State and Elementary Particle Physics," Proceedings of the IEEE, January 1984.

21. Batcher, K., "Design of Massively Parallel Processor," IEEE Transactions on Computers, September 1980.
22. Hills, D. "The Connection Machine" The MIT Press 1985
23. MRT micro Doc.#G160-10, MS160 Data Sheet. January 1995.
24. Croft W.B. "Knowledge-Based and Statistical Approaches to Text Retrieval" IEEE EXPERT April 1993.
25. Karlman Wasserman, James Jansen, Darryl Sue, Brian Whipp. *Principles of Exercise Testing and Interpretation*. Lea et Febiger, 1987.
26. Michael Richter and Stefan Wess. Similarity, uncertainty and case-based reasoning in PATDEX. In R.S. Boyer (ed.): *Automated reasoning, essays in honor of Woody Bledsoe*. Kluwer, 1991.
27. William Andersen, James Hendler, Matthew Evett, Brian Kettler. Massively parallel matching of knowledge structures. In H.Kitano and J. Hendler (eds.): *Massively parallel artificial intelligence*, AAAI Press, 1994.
28. Ingeborg Sølvsberg, Inge Nordbø, Agnar Aamodt. Knowledge-based information retrieval. *Future Generation Computer Systems*. Vol. 7, 1991/92.

About The Authors.

Ketil Bø is president of Dynamic Imaging As, Trondheim, Norway.

He received his Ph.D. in Computer Science from the Technical University of Trondheim in 1982.

He has been heavily involved in the work of ISO, IFIP and ANSI in the area of Computer Graphics and CAD/CAM.

In 1977-78 he was visiting professor at SMU in Dallas Texas.

He was Guest Editor of the IEEE Computer, Special issue of Human/Computer Interaction in 1982.

He is part time associated professor at the University of Trondheim

His current interest is in Imaging, Information Retrieval and Computer Graphics.

Agnar Aamodt is Professor of Artificial Intelligence in the Department of Informatics, Norwegian University of Science and Technology, Trondheim, Norway.

He has a M.Sc. degree in Electrical and Biomedical Engineering, and a Ph.D, in Artificial Intelligence, both from the University of Trondheim.

He has been a visiting scholar at the University of Texas, where he mainly worked on case-based reasoning, and a research scientist at the Free University of Brussels, where he worked with knowledge modelling methods and machine learning. His current research focus is knowledge-intensive case-based reasoning, driven by the goal of improving the construction and continuous maintenance of knowledge-based systems for human decision support.