2.2.4 Search- and Retrieval-Methods

The aim of information retrieval is to find relevant objects in an existing data stock that meet special conditions. These conditions have to be formalized using special methods and operators to describe the search criteria in detail.

Problem solving by searching can be divided into three steps [Russell & Norvig, 1995]:

- **Phrasing the problem**:
  Defining the starting point, the goal and all permitted actions. Therefore an abstraction of the real world has to be made that is useful and sufficient for the current problem.
- **Searching**:
  Defining a sequence of actions using an algorithm to meet the goal. More than one approach should be taken into account if possible. To come to an optimal solution methods to measure the performance and effectiveness of the different approaches has to be defined.
- **Execution**:
  Processing the sequence of actions step by step.

An overview of methods for automated problem solving is given in Schneider & Werner, 2007. Figure 2.2-11 shows an overview of search- and retrieval-methods described in this Section.

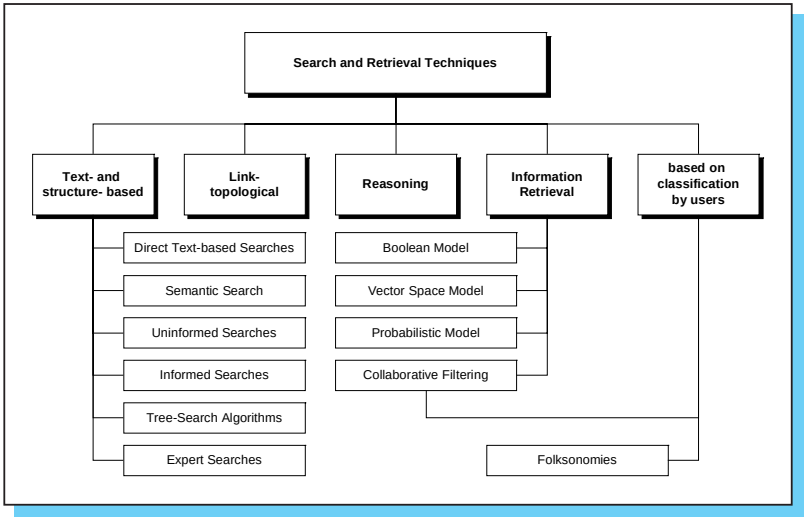

Figure 2.2-11 Overview of Search- and Retrieval-Methods

Search Techniques

- **Direct Text-based Searches**

  Most systems provide simple search methods within significant parts of their information stock for example in titles of documents. A special case of simple search is known as **word search**. It searches only for entire words within the information stock. Searching the word ‘data’ for instance did not find the word ‘database’.

  The **full-text search** is searching within the whole information stock – as it would be stored in a simple text file. Most systems use indexing methods to accelerate the performance of full-text searches. Some systems allow to use regular expressions within the search criteria.

- **Semantic Search**

  The basis for the semantic search is a semantic network. The semantic search uses the links between concepts and instances in the semantic network and therefore does not interpret the search terms individually and independently from one another, but searches for links between them, for example, in the form of a relation (or a shortcut relation, which combines

several “simple” relations). Concepts or instances connected to each other using a relation move nearer to each other. This way, they stand in a thematic context and in most cases constitute a very good search result. The semantic search may process results of a direct search.

- **Uninformed Searches** (also known as Blind Searches)
  These methods do not need additional information about the problem. The most frequent forms are depth-first search and breadth-first search.
  The **depth-first search** [Karagiannis & Telesko, 2001] starts at the root node and follows one branch to the end if it did not reach the goal on this route. If this leaf of the branch is not the goal, the system returns to the next higher branch, this is called backtracking. Thereafter, the system follows the branch to the end (see Figure B.3-1). The performance of this method is unpredictable and if there existing extremely long branches this could cause timeouts.
  The **breadth-first search** [Karagiannis & Telesko, 2001] starts at the root node and runs along one level of the tree. If the goal has not been found, the system continues with the next deeper level (see Figure B.3-2). The advantage of this approach is its completeness. This means that every node will be passed unless the goal has been reached. Due to this reason the breadth-first search is optimal for very deep trees but for trees with many branches the depth-first search performs better [Winston, 1993].
- **Informed Searches** (**Heuristic Searches**)
  Informed search uses knowledge about the problem / domain / application to enhance the efficiency compared to uninformed searches. This knowledge forms a **heuristic** implemented as a function
  h(n) = estimated cost of the cheapest path from a start node to the goal.
  Heuristics are informed guesses built for example by experience and ratings of other users. Therefore nodes are rated as shown in Figure B.3-3. A huge amount of different search algorithms using heuristic strategies is described in the literature. A selection thereof is presented here.
  Best-first search or **greedy search** [Karagiannis & Telesko, 2001] uses only the heuristic’s estimate to base its decisions. It always expands the node that it thinks is closest to its goal.

Hill-Climbing [Winston, 1993]: Like the depth-first search method but chooses the node with best heuristically value as illustrated in Figure B.3-3.
Beam-search [Winston, 1993]: Like with the breadth-first search the tree is navigated level by level. Using the heuristic function a defined amount of best nodes are used for forward navigation. (If w=1, beam-search is equivalent to Hill-Climbing. If w=infinite, beam-search is equivalent to best-first search.) Figure B.3-4 shows an example of beam-search with w=2.
A* Search [Karagiannis & Telesko, 2001]: Best-first search only looks forward. It ignores the costs of the path already gone. A* uses f(x) = g(x) + h(x), where g(x) is the cost of moves that have been made so far.

- **Optimal Searches**
  Many search problems have more than one possible solution (for example, more than one path through the network). Optimal search methods can be used to find the optimal path. A*, British Museum Procedure and Branch & Bound are optimal search methods [Karagiannis & Telesko, 2001].
  British Museum Procedure [Winston, 1993] calculates all possible solution paths and then chooses the best path. Depth-first and breadth-first search can be used to find the possible paths. British Museum Procedure can only be used for small graphs because of the huge number of possible paths that has to be calculated and compared.
  The **Branch & Bound** [Karagiannis & Telesko, 2001] strategy divides a problem into a number of sub problems. The calculated costs of a first solution are noted as an upper bound for the optimum. Following calculations are terminated if their costs exceed the upper bound to avoid the complete calculation of all partial trees. If a cheaper solution has been found, these costs are used as the new upper bound.
- **Tree-Search** Algorithms are optimised for searching within tree structures. Within knowledge management, most applications are based on networks. Hence, the tree-search algorithms are not discussed here. For more information see Bondy & Murty, 2008.

- **Expert Searches** [K-Infinity documentation, 2005]
  The expert query facilitates the compilation of concepts, instances, relations and attributes to build complex, static search queries whose results only change when the knowledge network itself changes (for example, by addition of new instances or creation of new relations). Figure 2.2-12 illustrates how expert searches can look like. Expert searches are a variation of semantic searches to provide searches adapted to special needs of the users, for example, a telephone book or a list of departments of a company.

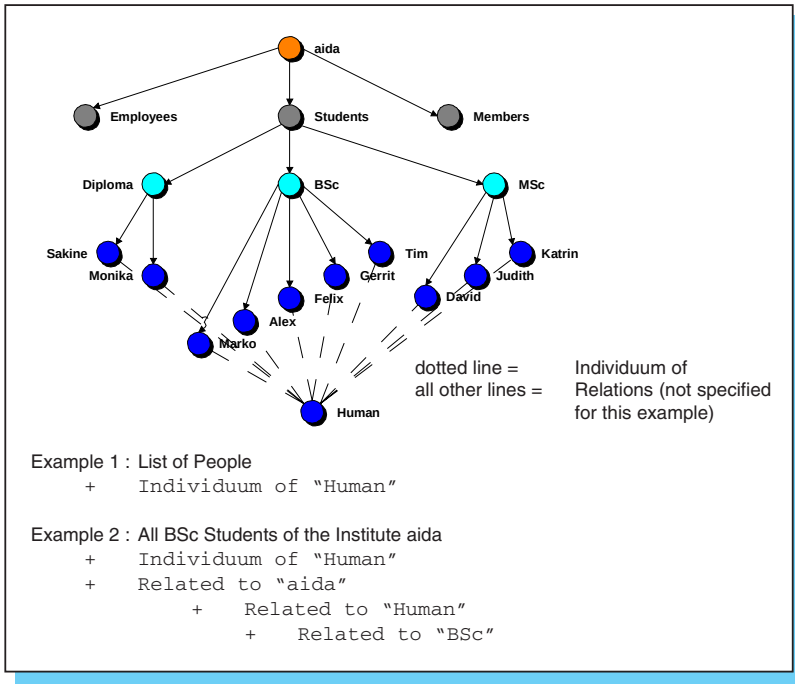

Figure 2.2-12 Example of Expert Search

Examples of the following search-algorithmn can be found in Appendix B.4:

- depth-first search and breadth-first search
- Hill-Climbing
- Beam-Search

Link-topological Algorithms

Link-topological algorithms are used in hypertext-oriented systems and in search engines for the World Wide Web. They calculate the popularity of hypertext pages using the links from other pages to the rated page. The rationale underpinning of most link-topological algorithms is that if the quality and relevance of a page is high it is linked by other pages. Usually, the semantics of a link does not matter; a link can be a recommendation or a warning (e.g. best or worst rated product in a consumer test).

Examples are Googles “PageRank Algorithm” [Brin & Page, 1998], “Hyperlink-Induced Topic Search Algorithm” (HITS) proposed by Jon Kleinberg [Kleinberg, 1997] and the “Hilltop-Algorithm” of Bharat and Mihaila that is based on HITS and uses only links provided by so-called “expert web sites”[Bharat & Mihaila, 2001]. Further algorithms can be found in Borodin et al., 2004 and Narsingh & Gupta, 2001.

A new trend is the use of so-called “domain popularity” as substitution of “link popularity” [Webb, 2005]. This means that only links from different domains are counted. Hence, it makes no difference how many links from one domain to a document exist, one link has the same weight as many links. This avoids the manipulation attempts by so-called search machine optimisers who developed web sites containing a huge amount of links to their customer’s web sites.

Reasoning

Case-Based Reasoning (CBR) is a strategy for solving actual problems by using information in the new context that have been stored during a solution of a similar problem in the past. CBR is often referred to as “reasoning by remembering” or “learning by examples”. The basic concept can be described as “similar problems have similar solutions”. [Aamodt & Plaza, 1994]

The classical domains for CBR are classification systems and help desk systems [Aamodt & Plaza, 1994]. In the meantime, it is used also in electronic commerce and knowledge management systems [Mansar et al., 2003].

A CBR system consists of

- the vocabulary,
- the case-base,
- the measurement of similarity and
- the adaptation of solutions.

The case-base is the storage container for previous cases. A case describes a specific problem situation and consists of a problem part and a solution part.

Case-Based Reasoning Cycle

The Case-Based Reasoning Cycle describes the four most important phases of a CBR system as shown in Figure 2.2-13 [Aamodt & Plaza, 1994]:

1. Retrieval phase: After presentation of the problem an applicable case will be selected from the case-base.
2. Re-Use phase: The solution of the selected case used or adapted for the actual problem.
3. Revise phase: The new solution will be tested concerning its practicability and correctness and revised if necessary.
4. Retain phase: The new solution will be stored in the case-base.

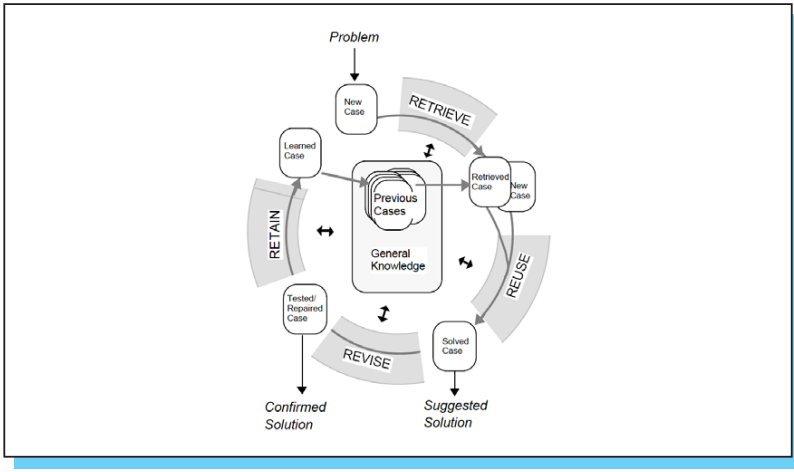

Figure 2.2-13 The Case-Base Reasoning Cycle [Aamodt & Plaza, 1994]

Additional activities can be added to the CBR cycle: Learning (for example by improving the similarity measurement) or maintenance (for example deleting cases that are not longer of relevance). Learning is of vital importance in a CBR system. The assimilation of new cases into the case-base implies an incremental learning process.

Typical applications for CBR are collections of FAQ (frequently asked questions) or "lessons learned systems" that are used for information distribution.

Information Retrieval

The definition of Information Retrieval (IR) is rather vague and emphasises the vagueness of search queries and the uncertainness of stored information [Ferber, 2003]. IR focusses on computer-based content-oriented searching for information within documents. IR also provides methods and strategies for the representation, storage and organisation of information. IR also can be used for searching for documents themselves, metadata which describe documents, or searching within all kinds of databases.

A common method in IR is to use so-called "Stop-Word" lists to exclude general words (for example "do", "with", "use", "all", "for") to optimise the performance of the different algorithms.

The most important IR Models are the Boolean Model, the Vector Space Model and the Probabilistic Model described here [Lewandowski, 2005].

Boolean Model

The Boolean Model works with the "exact match" method. The term of the request has to exist within the document and must match exactly with the request. Usually, the Boolean operators "AND", "OR" and "NOT" and brackets can be used to phrase the request.

The "exact matching" strategy causes problems when synonyms, different languages, singular or plural forms are used. Also a document is only found if it fulfils the request completely (if a document fulfils n-1 of n conditions it is not found). Ranking of results

is not possible, the significance of terms within the request or the text is ignored [Belkin & Croft, 1987] and it is not possible to indicate and sort similar results. Variations of the Boolean Model try to overcome these issues for example by using basic forms of words, a controlled vocabulary and a thesaurus for synonyms. Another disadvantage of the Boolean Model is the difficulty to phrase the requests.

Vector Space Model

Text (or other) documents are represented as vectors of identifiers (for example index terms). Each term is represented by one dimension. A vector is zero if the term does not exist within the document or non-zero if the term exists. The definition of term is application depending. Terms can be single words, keywords, or longer phrases. [Salton et al., 1975]

Using the Vector Space Model similarities between documents or between a search request and a document can be found. It does not search for exact matches. Several methods to calculate similarities can be used. One of the favourites is to calculate the cosinus of the angle between the two vectors. The smaller the cosinus the more similar the two compared documents or search request and document are. Based on this hypothesis search results can be ranked.

Disadvantages of the Vector Space Model are the missing operators to describe dependencies of search terms (e.g., synonyms with “OR”) or to exclude terms [Lewandowski, 2005].

To enhance the model, different methods can be used to quantify terms. Most commonly used is the frequency of a term within all documents of a collection which is a global and context-independent quantify factor. In information retrieval, the inverted document frequency is often used that describe the number of documents where a term is contained; this can be stored in separate (system global) lists to enhance the performance of the IR system. The frequency of a term within a document can also be used to quantify the document. The higher the frequency of a term the more important the term should be for the document. In practise, the measurement is restricted to an interval to understate the impact of terms that are all too frequently used in a document.

In addition, it makes sense to normalize the frequency of a term in relation to the document size or in relation to the most used term in the document.

Relevance Feedback is a method to enhance the quality of the retrieval results by modifying the used request vector. Therefore, the searching user has to give the relevance of the found documents as feedback to the system. Using this iterative search strategy, the results can be more similar to documents that are relevant to the user and less similar to documents that are not relevant by automated modification of the search vector. [Lewandowski, 2005]

Latent Semantic Indexing (LSI) is using semantic coherencies between terms to enhance the results of the Vector Space Model. LSI is a method for indexing of huge amounts of documents by reducing the dimension of the vector using “Singular Value Decomposition” [Ferber, 2003].

An extension of LSI is Probabilistic Latent Semantic Indexing (PLSI) [Hofmann, 1999] based on the Aspect-Model. The Aspect-Model is a statistical and probability model introduced by Jordan et al., 1998. In a collection of documents with X as the set of documents and Y as vocabulary (= set of terms) a pair (x,y) describes that the term y is contained in the document x. In addition, the strength respectively the frequency (or weight w) can be related to a pair: w(x,y). It is called the latent class. Pairs of the same class are called aspects. PLSI use the Expectation-Maximization-Algorithm (EM-Algorithm) to estimate and optimise model parameter of the Aspect-Model as described in Hofmann, 1999 and Jordan et al., 1998. PLSI uses the learned parameter of the Aspect-Model to create a vector-based representation of the documents in the Vector Space Model.

Probabilistic Model

In Probabilistic Models the process of document retrieval is defined as a probabilistic inference. The relevance of a document is calculated based on similarities between a document and a query where the value of similarity is depending on the frequency of query terms within the document. The better the similarity the more probable the relevance for the user. [Lewandowski, 2005]

The results can be ranked and a threshold is used to decide the minimum of relevance probability of a document to be included in the results.

Fuhr, 2004 describes the advantages of the Probabilistic Model but, in practise, the model has not been established and offers no improvements of retrieval results compared with other models [Chu, 2003].

	Boolean model	Vector space model	Probabilistic Model
Boolean Operators	+		
Quantifying		+	+
Ranking		+	+
Criteria of Conformance	Existence of Term	Vector Distance	Frequency of Term
Unique Selling Point		Relevance Feedback	

Figure 2.2-14 Comparing the Characteristics of the IR Models [Chu, 2003]
(+ means supported by)

Collaborative Filtering

Aim of Collaborative Filtering or Social Filtering is the evaluation and rating of information by a group of people interesting in a specific topic [Goldberg et al., 1992]. The other IR methods try to describe documents and their characteristics by a representation and use this representation for searches within the set of documents. The search context is depending of the content. Collaborative Filtering use another kind of context: How a user classifies a document. Also the relevance of a document can be classified of the user. If groups of users with same interests can be identified, the evaluation of information by members of that group should be of interest for other group members. Profiles of users are the basis for a good quality of results. This is one of the issues of that strategy: A profile has to be developed and therefore the user has to define his interests or rate some documents before the system can present results of interest for the user.

This kind of filtering is not limited to documents or other digital information. It can also be used for all kinds of objects [Ferber, 2003], for example books, films, music, events or physical objects as places and buildings of interest.

Online book stores are a good example for the use of Collaborative Filtering. If the user profile contains a list of books bought or rated by the user, the system can provide suggestions for other books which may be of interest for that user. In addition, if the user adds a book to the shopping cart, the system can tell him that other customers who bought this book also bought or are interested in a list of other books.

Folksonomies

Using Web 2.0 services and technologies, the consumer of information also is able to produce information in an easy way. Users became “prosumers” [Toffler, 1980], producers and consumers. They can collaborate not only by creating content, but to index these pieces of information as well. Folksonomies enable users to describe documents with meta-data, so-called “tags“, without regarding any rules [Peters & Stock, 2008]. Tags are subject headings and can be shown to the user as a “tag cloud“ [Sinclair & Cardew-Hall, 2008]. Figure 2.2-15 shows an example of a tag cloud on the left-hand side and how a tag for a document can be edited (within a document management system).

Folksonomies are an improvement of content in terms of adding free keywords to documents in the internet [Furnas et al., 2006].

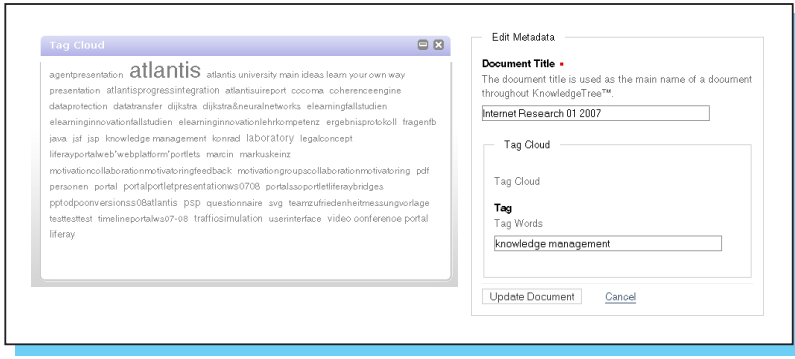

Figure 2.2-15 Example of a Tag Cloud and how to Edit a Tag for a Document

Al-Khalifa & Davis, 2007, stated that tags indexed by prosumers do not correlate to terms calculated by automated algorithms. This indicates the different point of view of humans vs. machines. The development and maintenance of existing controlled vocabularies might benefit from folksonomies [Aurnhammer et al., 2006].

Advantages of Folksonomies

The prosumers or actors index the document themselves. Hence the folksonomy use the language and knowledge in an authentical way [Quintarelli, 2005]. This comes to multiple interpretations and multicultural views of the same information [Peterson, 2006]. These shared inter-subjectivities allow the user to benefit, not just from own discoveries, but from those of others [Campbell, 2006].

Folksonomies complete the set of knowledge management tools and methods as illustrated in Figure 2.2-16. Three groups of actors can be distinghuished: Authors, professional indexers and users [Kipp, 2006]. The three kinds of actors index in different ways and focus probably on different attributes of the document [Peters & Stock, 2008]. This helps to provide useful search result sets for all possible user types.

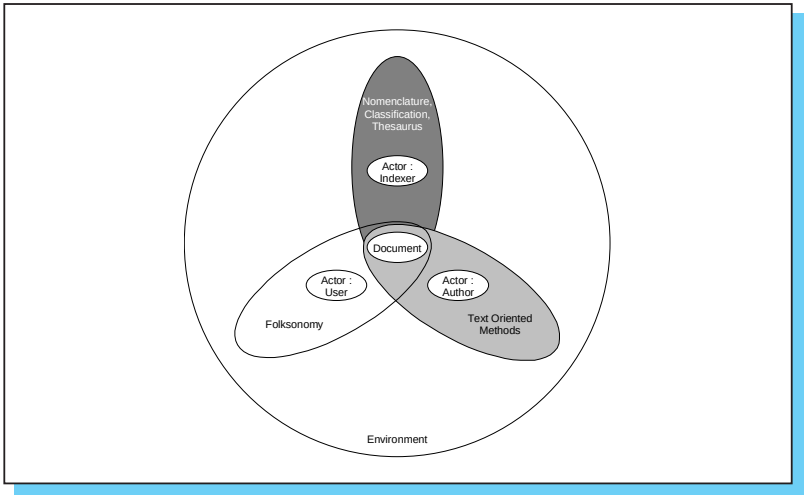

Figure 2.2-16 Methods of Knowledge Representation and their Actors [Stock & Stock, 2008]

Folksonomies can be integrated with ontology-based systems to construct and optimise ontologies [Van Damme et al., 2007]. Within professional environments like Intranets tagging can help users to find things again [Fichter, 2006] and to combine findings in their own meaning. A strategy for the design of a corporate knowledge management system combining a semantic web layer with Web 2.0 tools like folksonomies is introduced by Passant, 2007.