Search System Functions for Supporting Search Modes

Thomas Beckers
Information Engineering
University of Duisburg-Essen
Duisburg, Germany
thomas.beckers@uni-due.de

Norbert Fuhr
Information Engineering
University of Duisburg-Essen
Duisburg, Germany
norbert.fuhr@uni-due.de

ABSTRACT

Tasks in web search are often rather simple, e.g. navigating to an already known web page or looking up a fact. However, tasks in other domains are usually more complex and diverse. Thus, we discuss various search modes of tasks and how they might be supported by functions of a search system. We give examples of the required search functions of different search modes and describe the implications for the design of search systems.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: General

General Terms

Human Factors

Keywords

system functions, search modes, user interfaces

1. INTRODUCTION AND RELATED WORK

While tasks in web search are often rather simple [4] (e.g. navigating to an already known web page or looking up a fact), tasks in other domains (e.g. searches for scientific literature or patents) are usually more complex and diverse. A set of search system functions that is well-suited for these simple tasks is not appropriate for other more complex task types. In our opinion, each type of task requires a different set of search system functions. Thus, we argue that a "one size fits all" approach (that is, using a search systems with functions e.g. optimized for web search for different tasks in other domains) does not allow the user to search effectively and efficiently. We propose a model of search functions that allows mapping of search activities (search tasks) to necessary system functions comprising the entire search activity.

Hughes-Morgan and Wilson [7] have examined whether improvements of an interactive search system are due to the newly introduced meta-data or to new search functionality. They conclude that users can benefit from improved search features while still using the same meta-data.

Russel-Rose et al. developed a taxonomy for enterprise search and site search by analyzing real-world scenarios [11, 12, 10] based on three top-level categories of search activities originally proposed by Marchionini [8]:

Lookup *a*) Locate *b*) Verify *c*) Monitor

Learn *a*) Compare *b*) Comprehend *c*) Explore

Investigate *a*) Analyze *b*) Evaluate *c*) Synthesize

These categories are orthogonal to each other. Russel-Rose et al. [11] introduce the notion of *search modes*. A search mode is a concrete value of a search activity category. Search modes can be combined to longer sequences or networks. For enterprise search *Locate* is far less common then *Analyze* and *Evaluate*. In the domain of site search the emphasis is on *Locate* and *Explore*.

In the remainder of this paper, we first describe the functional level of search systems. We show how search functions can be mapped to different search modes by giving examples to illustrate how search systems can support each mode and its associated search functions. Subsequently, we describe the implications for designing and developing search systems. Finally, we give an outlook on future work and a conclusion.

2. SEARCH SYSTEM FUNCTIONS

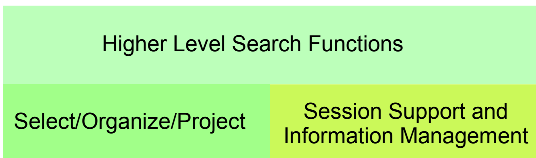

Figure 1: Functional level of IR systems

We divide the functionality of an IR system into three different groups depicted in Fig. 1: *i*) Select/Organize/Project (SOP) *ii*) Session Support and Information Management (SSIM) and *iii*) Higher Level Search Functions (HLSF). In our notion a *search function* is a functionality of the system with which the user can interact or that is fixed by the system designer. A more detailed explanation of the latter two groups and an overall architectural view is given by

The 2nd European Workshop on Human-Computer Interaction and Information Retrieval (EuroHCIR), Nijmegen, The Netherlands

Beckers and Fuhr [3]. We will concentrate on SOP and (to a lesser degree) on HLSF in the following. In doing so, we will focus on system functions and not discuss their concrete visualizations in the user interface.

Select functions

Select (S) comprises functions for selecting (searching) possibly relevant items.

Ranking method Retrieval functions/ranking methods may be more precision- or more recall-oriented, or they may consider different sources of additional information (like e.g. page-rank). Mutschke et al. [9] showed that search in scientific literature can be improved by considering information about the author, the publication venue or related terms from a thesaurus.

Ranking principle The final ranking might regard each document in isolation, or consider all items above the current one in the output ranking (like e.g. in diversity ranking).

Query language The query structure can be very simple (e.g. a list of terms) or more powerful and expressive, e.g. by supporting simple (boolean) and more complex (wildcards, word distances, etc.) query operators as well as fields and data types.

Formal filter conditions The result set can be filtered by some formal criteria (e.g. by data type, source, date) which is usually done without affecting the RSV.

Query formulation Queries can be formulated a priori as in most systems but also by referring to one or more given items (e.g. query by example, similarity search).

Organize functions

Organize (O) functions deal with the way how the set of result items is structured and organized logically.

Sorting The results can be sorted according to one or more criteria. When searching the best offer for a new smartphone the items may be sorted by price and the trustworthiness or customer ratings. While sorting usually is a one-dimensional organization, also two- or three-dimensional organizations may be helpful, provided that appropriate visualizations are available in the user interface.

Grouping The results can be grouped according to a simple criterion (e.g. grouping by release date, author, source) or according to several facets, as in faceted search [13].

Clustering While grouping is based on some formal criteria, clustering focuses on content aspects based on a some sort of similarity [5]. Although users might have problems interpreting the cluster structure, they might also gain new insights about the result set.

Linking In case there are (explicit or implicit) links between the answer items, the resulting tree or graph structure might be of interest (e.g. Web links, co-author relationships in scientific literature, or friendship connections in social networks).

Project functions

Project (P) comprises functions for the construction of the surrogates to be presented in the results.

Selection Surrogates consists of specific fields of the result items (like e.g. title, author and year in literature search).

Summarization Either unbiased or query-biased summaries (extracts) of the answer documents (or specific fields thereof) can be generated.

Aggregation This function generates a single entry representing several items differing in formal aspects (e.g. mirrors of a web page, various editions of a book) or content (e.g. different reviews of a book in an online store).

Faceting For displaying facets with their existing values and corresponding frequencies, the system must support projection on single facets along with counting values. From the point of view of a relational database, if `F` denotes a facet/attribute, then the system has to process the SQL query "`select F, count(*) from R where ... group by F`" for each facet. Query conditions and restrictions wrt. to the values of a facet then affect the `where` part of the query.

Enrichment By using external data sources the results can be enriched with additional data (e.g. on a product review site, linking to online stores for each product).

Extracting The items can be used to extract new data characterizing the whole result set, e.g. common terms in the documents or frequent authors.

Higher level search functions

According to Bates [1] a system should not only offer basic functionality. It should also provide support for search tactics, stratagems and strategies. In our model a HLSF is a function that uses lower level SOP and/or SSIM functions (called *moves* regarding Bates' terminology) for providing tactical and strategic support. For example, when searching for relevant literature about a certain topic a stratagem consisting of two tactics would be to *i*) search for documents that contain some terms describing the topic and then *ii*) using a function for exploring references and citations of documents to find related documents. An ideal system should also be able to support these kinds of search functions.

3. SUPPORTING SEARCH MODES

We think that the search mode taxonomy is flexible and general enough to be also well-suited for many other domains. We regard a search mode or a sequence of search modes (just called *search mode* in the following for the sake of simplification) as a higher level search function (or task) as defined by Bates. In the following we will give examples which functions are particularly required for supporting certain search modes. Functions from all three groups are required of course but we will focus on those that are the most important and distinctive ones. These requirements are listed in Table 1 and will be explained in more detail in the following.

		Search Functions		
		Select	**Organize**	**Project**
Lookup	**Locate**	Query language, Ranking method		
	Verify			Selection
	Monitor			Selection
Learn	**Compare**		Sorting	Selection
	Comprehend		Grouping	Faceting
	Explore	Formal filter conditions	Grouping, Clustering	Faceting
Investigate	**Analyze**		Sorting, Grouping, Clustering, Linking	
	Evaluate		Sorting	
	Synthesize		Join	

Table 1: Most important and distinctive (groups of) functions for each search mode

Lookup: Locate For supporting this search mode (often a known-item search) it is important to offer appropriate *select* functions that allow the specification of the known attributes. For example, when searching for a scientific publication, the user might know some words from its title as well as the publication venue—so the system must allow for searching in specific fields.

Lookup: Verify/Monitor If the user wants to verify that an item meets some specific and objective criteria or when s/he wants to monitor an item to maintain awareness the system should be able to *project* on the relevant parts or attributes of the result items. For example, when finding out whether a central processing unit (CPU) is compatible with a specific mainboard chipset the system should show the compatibility information of the CPUs in the result items.

Learn: Compare Comparing items in the results to identify similarities or differences requires the system to *organize* the items as a list and to offer a *projection* of all relevant aspects visualized in tabular form. Alternatively, the items may be organized in a multi-dimensional grid. For example, when comparing products, both price and performance of products are relevant criteria.

Learn: Comprehend For supporting comprehension of result items by finding patterns and traits the system should allow the user to *organise* and *project* the results by grouping them according to one ore more facets, in order to gain a understanding of the structure of the result set. For example, a user interested in buying an solid-state drive for his/her computer first has to comprehend the possible values of the relevant attributes (e.g. storage size, host interfaces, operating system requirements, etc.) by faceting.

Learn: Explore Faceted search supports exploration. Besides *selecting* a specific value of a facet as a formal filter condition, the system should offer functions for *organizing* the result items into different groups for each facet. For more content-oriented searches, clustering functionality may help the user in understanding the various aspects of a topic.

Investigate: Analyze Analyzing items to identify patterns and relationships is a very complex task. Thus, the system should offer several versatile and powerful *organization* functions.

There are several functions that may be helpful for the user here, such as *i*) (multi-dimensional) sorting, *ii*) grouping, *iii*) clustering and *iv*) linking of the result items. Sorting result items allows the user to inspect the items by the priority of one or more sorting criteria. The HyperScatter component of the visual information seeking system MedioVis [6] would be a proper visualization and interaction technique. Especially, multi-dimensional sorting might help in understanding the relationship between facets (e.g. when buying a digital camera, the user might want to learn which features have a strong influence on the camera price). Functions for grouping may help the user in gaining new insights or getting an overview of the result items (see preceding search modes). Clustering the result items may be helpful for finding previously unknown similarities by creating groups of items with an unknown meaning. Additionally, a clustered result set may support the user in getting an overview of the found items easier. Functions for linking the items can produce tree or graph structures of the result set. These functions can be used for creating e.g. networks based on some kind of relationship.

Investigate: Evaluate For judging the value of an item concerning a specific goal or purpose the system should be able to let the user *organize* the result items according to the important criterion, e.g. by sorting.

Investigate: Synthesize This search mode occurs when the user is creating new objects from the found result items. We envisage that a system may support this by offering a join function similar to joins in relational databases.

The system does not have to allow the user to perform all functions that are theoretically possible. Instead, the system should perform certain functions automatically and should use suitable preadjustments and defaults (see levels of system involvement by Bates [2]). Which functions the user should interact with depends on the search modes and the domain in which the system is actually used.

4. IMPLICATIONS FOR THE DESIGN OF SEARCH SYSTEMS

In the previous section we provided examples which functions are required for different search modes. An ideal search system should be flexible enough to support a broad variety of search modes. Which set of functions is exactly required certainly depends on the context the system is used within and the tasks a user typically performs. Adding as many functions as possible may leads to a feature-bloated system. Instead, only the appropriate functions should be offered to the user. Richer functionality requires increased user expertise. Thus, the interaction and visualization techniques have to be chosen carefully to provides an easy-to-use system. Further open research issues concerning rich functionality have been described by Beckers and Fuhr [3].

The discussion in this paper has shown that the ideal search system extends classical IR functionality with typical database functions, as well as more advanced IR functions. Thus, typical IR systems as well as relational database systems are both far away from the ideal system. An XQuery system with full-text search might come closest today, but it lacks all the more advanced IR functions. Whatever the resulting query language might look like, however, it should be clear that it mainly targets at the application developer, who specifies the functionality needed, which is then mapped onto a user-friendly interface.

5. CONCLUSION AND OUTLOOK

We demonstrated how different search modes require different search functions of the system. Thus, an ideal search system suitable for various search modes should not only support classic search functions for ad-hoc retrieval (e.g. ordinary web search) but also more advanced functions described in this paper. Our grouping of search functions allows the identification of functions possibly required for a certain search mode. Previous research in this area can be categorized and integrated.

Further empirical research is necessary to validate our proposed mapping from search modes to search functions. A first step may be to show exemplarily that for a particular search tasks the users can benefit from improved and suitable functionality by controlling the other variables.

6. REFERENCES

[1] M. J. Bates. Information search tactics. *Journal of the American Society for Information Science*, 30(4):205–214, 1979.

[2] M. J. Bates. Where should the person stop and the information search interface start? *Information Processing and Management*, 26(5):575–591, 1990.

[3] T. Beckers and N. Fuhr. Towards the systematic design of IR systems supporting complex search tasks. In *Proceedings of the Task Based and Aggregated Search Workshop @ ECIR 2012*, April 2012.

[4] A. Broder. A taxonomy of web search. *SIGIR Forum*, 36:3–10, September 2002.

[5] N. Fuhr, M. Lechtenfeld, B. Stein, and T. Gollub. The optimum clustering framework: Implementing the cluster hypothesis. *Information Retrieval*, 15:93–115, 2012. DOI: 10.1007/s10791-011-9173-9.

[6] M. Heilig, M. Demarmels, W. A. König, J. Gerken, S. Rexhausen, H.-C. Jetter, and H. Reiterer. Mediovis: visual information seeking in digital libraries. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '08, pages 490–491, New York, NY, USA, 2008. ACM.

[7] K. Hughes-Morgan and M. L. Wilson. Information vs interaction – examining different interaction models over consistent metadata. In *Proceedings of the IIiX conference*, 2012. To be published.

[8] G. Marchionini. Exploratory search: from finding to understanding. *Commun. ACM*, 49(4):41–46, Apr. 2006.

[9] P. Mutschke, P. Mayr, P. Schaer, and Y. Sure. Science models as value-added services for scholarly information systems. *Scientometrics*, 89(1):349–364, Oct. 2011.

[10] T. Russell-Rose. A taxonomy of site search. Talk at Enterprise Search Europe, UK, May 2012.

[11] T. Russell-Rose, J. Lamantia, and M. Burrell. A taxonomy of enterprise search. In *Proceedings of euroHCIR*, 2011.

[12] T. Russell-Rose, J. Lamantia, and M. Burrell. A taxonomy of enterprise search and discovery. In *Proceedings of HCIR 2011*, October 2011.

[13] D. Tunkelang. *Faceted Search*. Number 5 in Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers, 2009.