UDC 004.738.52

Evaluation and Comparison of Search Engines Using the LSP Method

Jozo Dujmović [1]and Haishi Bai [1]

[1] Department of Computer Science
San Francisco State University
jozo@sfsu.edu, haishi.bai@gmail.com

Abstract. We present a comprehensive model for quantitative evaluation and comparison of search engines. The model is based on the LSP method for system evaluation. The basic contribution of our approach is the aggregation of all relevant attributes that reflect functionality, usability, and performance of search engines. In this respect our model is fully consistent with the ISO 9126 standard for software product evaluation. Performance analysis of competitive search engines is based on our search engine benchmarking tool (SEben) that is also described in the paper.

1. Introduction

Search engines were introduced in 1993 and the first attempts to develop techniques for their evaluation were published in 1996 [CHU96]. During the period of ten years the search technology made a dramatic progress [BRI98, BAE99, SPA97, CHA03, HAW06] and currently search engines are the most influential web tools. A survey of some major search engines is presented in Table 1. Some of the engines use proprietary search technology (PST), some of them are meta search engines that distill and aggregate results of multiple PST search engines, and some of them offer specialized search services using a selected PST provider.

General search services are accessible through all web browsers (e.g. Netscape) and are also offered by Internet service providers (e.g. AOL). According to a study conducted by Nielsen NetRatings in July 2006 [SUL06] out of the 5.6 billion searches placed in that month, the leading GYM trio (Google, Yahoo, Microsoft) handled more than 80% of total traffic, with Google taking a leading 49.2% (see Table 1 and [BAE99] for a very different distribution in 1998).

Regardless a visible industrial interest in search engine companies (e.g. see [SEW06, SEG06, GAR06]), and intensive research in the area of information retrieval, the existing search engine evaluation papers are restricted to evaluating only selected aspects of search technology [CHU96, JAN03, TAN03]. Search engines are not quantitatively evaluated as complex industrial products that are designed to satisfy user requirements.

In this paper we focus on building a comprehensive model for evaluation and comparison of general PST search engines. Our model reflects the ability of search engines to satisfy user requirements. The experimental part of the paper includes the evaluation of four leading PST search engines: Yahoo! Search, Ask, Google, and MSN. Our criterion reflects the needs of both nonprofessional and professional searches. The results of performance measurements, collected by our benchmarking tool, are aggregated with all attributes that reflect the functionality and usability of search engines to generate a compound indicator of the overall quality.

Table 1. A survey of English domain search engines

Search Engine	Begin [year]	Traffic July'06	Type May'06
Aliweb	1993	-	PST
WebCrawler	1994	-	Meta
Infoseek/Go	1994	-	Provider
Lycos	1994	-	PST
Altavista	1995	-	Provider
Excite	1995	-	Meta
Mamma	1996	-	Meta
Dogpile	1996	-	Meta
Ask	1996	2.6%	PST
Google	1998	49.2%	PST
AlltheWeb	1999	-	Provider
Teoma	2000	-	Provider
Vivisimo	2000	-	PST
Kartoo	2001	-	Meta
AOL Search	2003	6.3%	Provider
Yahoo! Search	2004	23.8%	PST
MSN Search	2004	9.6%	PST
A9	2004	-	PST
Snap	2004	-	PST
Quaero	2006	-	PST

Our evaluation method is presented in the following five sections. We first introduce the search engine user and workload models, and then the search engine attributes that are the basis of the evaluation model. The attributes include performance and quality of information retrieval that are measured using a specialized tool, which is presented in a separate section. Then we outline the LSP evaluation method. Finally, we present a comprehensive LSP criterion for search engine evaluation and the evaluation results.

The area of search engines is very dynamic, characterized by permanent developments and improvements. The evaluation results presented in this paper reflect the status of evaluated search engines in June 2006.

2. Search engine user and workload models

All parameters of search engine evaluation model, as well as the search engine benchmark workload, reflect a specific model of the search engine user. We characterize users using two primary attributes: (1) volume, and (2) significance of search. The volume is measured in searches per day of a specific topic. The significance is characterized as “low” if the search is purely recreational (as the majority of searches in Table 3). The significance is “high” if the results of search significantly affect professional work, health, security, family activities, etc. Two characteristic types of users are shown in Table 2.

Table 2. Characteristic search engine users

Symbol	Type of search engine user	Volume of search	Significance of search
GP	General population	High	Low
SU	Specialized user	Low	High

Short-term high-intensity search traffic is generated by special events that cause high level of public interest. Such events are illustrated by the search frequency distribution shown in Fig. 1 [GOG06b]. After important events (e.g. natural disasters, war events, global-scale diseases, terrorist attacks, events related to celebrities, etc.) the frequency of searches related to such events rapidly increases, reaches the maximum, followed by an almost exponential decrease, similar to the Rayleigh or log normal distributions. In many cases the interest in special events fades out rather quickly and does not affect the top 10 yearly queries that characterize the GP user.

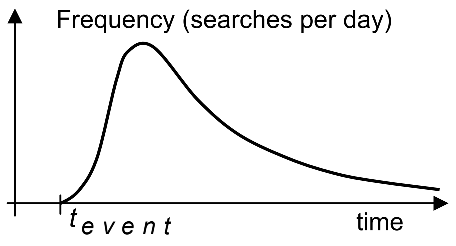


Fig. 1. A typical event search frequency

The GP user is defined using the statistics of most frequent requests obtained from search engine traffic monitors. The GP traffic is dominated by large public interest in special (sometimes accidental) events and popular public personalities. Table 3 shows an example of the top 10 Google queries [GOG06a] that illustrate the worldwide activity of the GP user. The overlap of each pair of consecutive sets is only 10% showing the shift of public interest.

Table 3. Top 10 Google queries in years 2002, 2003, and 2004

Rank	2002	2003	2004
1	BBC	Prince Charles	BBC News
2	Big Brother	Eastenders	Big Brother
3	Easyjet	Winnie the Pooh	CBBC
4	Britney Spears	Jonny Wilkinson	Autotrader
5	Ryanair	Easyjet	Dictionary
6	Gareth Gates	David Beckham	Tesco
7	Weather	Michael Jackson	Eastenders
8	Kylie Mingue	2 fast 2 furious	Weather
9	World Cup	Paris Hilton	British Airways
10	Holly Valance	Simpsons	National Lottery

The SU search load is generated by many categories of special interest user groups, where the most important are professional searches. The SU group reflects the fact that Internet is today a critical component of the majority of professional activities, education, business, research, health protection, politics, social work, entertainment, etc. A significant fraction of search activities is related to search of literature.

Performance and quality of information retrieval are important components of our search engine evaluation model. In the case of GP workload the performance and retrieval quality attributes can be measured using a specialized benchmarking tool. In this case the drive workload must reflect the interest of general public, and we used the statistics of most frequent requests obtained from traffic monitors of major search engines. Table 4 shows examples of GP workload that we used in performance measurements.

Table 4. Samples of GP workload

Workload	Queries	Retrieved URLs
Top 10 Google Queries 2003	10	8000
Top 20 Yahoo Queries 2006/5	20	12816
Top 200 MSN Queries 2005/8/19	200	112183
Top 10 Google Queries 2002/4	10	6922
Top 10 Ask Queries 2006/11/3	10	6803

The SU workload has strong semantic component and the quality of information retrieval can be fully analyzed only by experts in a specific area. For example, the recall of a query about “andness” should be evaluated by decision analysts, and the recall of a query about “Rituximab” should be evaluated by medical experts. A complete analysis of performance and quality of search engines for SU workload cannot be done automatically using a benchmark tool. Therefore, an automatic analysis of performance and quality of search is reasonable for GP workload, and this is done by our SEben tool. In the case of SU workload, a benchmarking tool has limited applicability.

3. SEben – a search engine benchmarking tool

SEben is our tool for search engine benchmarking and for measurement of quality of information retrieval (IR). Following are main SEben design goals:

- Measurement of IR quality (precision, recall, and coverage of search)
- Measurement of response time
- Comparison of competitive search engines
- Continuous measurement of distributions of performance indicators
- Extensibility to support more search engines

The global organization of SEben is shown in Fig. 2. The performance measurement consists of *C* cycles, where each cycle selects one of *b* information retrieval benchmark queries ($Q_1,\ldots,Q_b$), and applies this query to all *n* analyzed search engines. Measured performance values (quality of search, response time, etc.) are stored in a performance database, and the accumulated data are processed at the end of measurement to compute global performance indicators for each search engine.

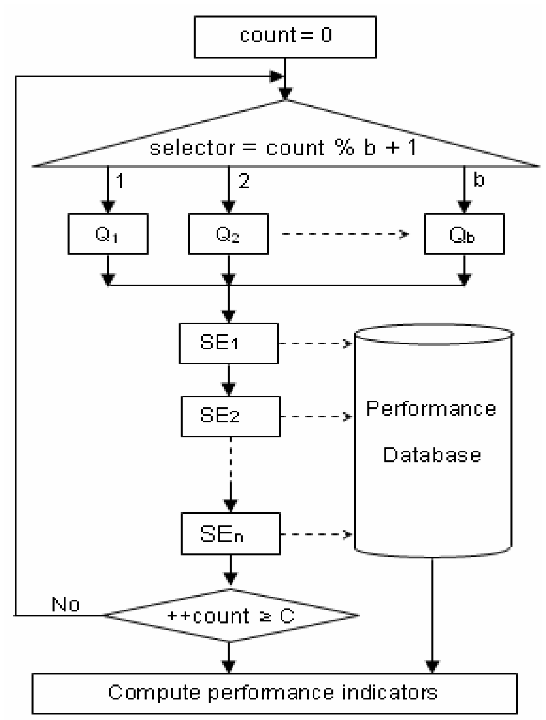


Fig. 2. The organization of the SEben tool

The goal of selecting the benchmark queries is to have a representative workload that primarily includes those queries that have high frequency. Our benchmark queries (initially *b*>300) reflect the activities of the general public and include most popular search topics reported by Yahoo! Search, MSN, Google, and Ask in the past 6 years. For example, one of our query sets was taken from Google Zeitgeist Archive [GOG06b]. SEben first submits selected queries to analyzed search engines and measures response times. Then, the returned pages are analyzed to collect URL's for later analysis. Coverage, recall, and precision are calculated at a later time based on the collected URL's in database.

The core functionality of SEben is exposed as a Web Service that receives test requests from Web UI, Windows scheduled tasks, or other client applications (Fig.3). The test requests are then forwarded to various search engine classes, which inherit from one base class.

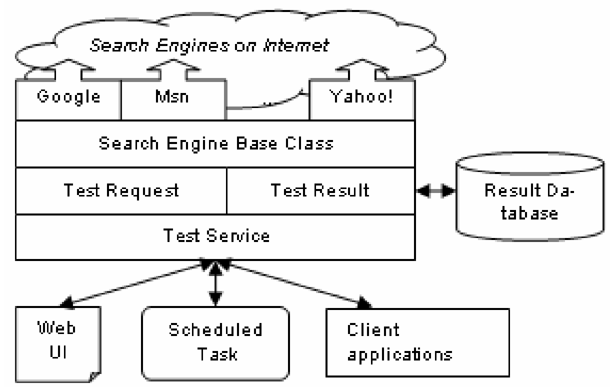


Fig. 3. SEben input and output

The base class encapsulates common functionalities such as sending HTTP request, retrieving response stream, etc. All test results, including both collected metrics and retrieved URL's, are logged to the performance database.

The performance database contains several tables. The most important are the summary table (containing response time for the first page, the cumulative response time for all retrieved pages, the number of bytes per each page, and the total number of retrieved URL's) and the detailed table (containing all retrieved unique URL's). We also collect time stamps and other data necessary for analyzing distributions of measured values.

This design provides a reusable framework for search engine evaluation, and new search engines can be supported by creating new search engine classes. Engine-specific code only needs to implement two methods: GetSearchAddress, which accept keywords and format a query string following specific format defined by the target engine; and GetSearchResult, which parses acquired HTTP response stream to collect returned URL's.

In the following sections, we summarize the theoretical background of information retrieval quality measurements implemented in SEben.

3.1. Measurement of server response time

Let us define the server response time is the time that a search engine needs to execute a query and to generate the result set. Although most of search engines display server response times on result pages, we decided to find an independent measurement to verify these claims. In order to eliminate effects of network delays, we measured the server response time in the following way:

1) An empty search request (the search without any keyword) is submitted to the search engine. The total response time between when the request is sent and when HTTP response stream is acquired is recorded as t_1.
2) Immediately after we get response from step 1, we send out actual search request and the corresponding response time t_2 is recorded.
3) The server response time is $t_2 - t_1$.

The rationale behind this measurement is simple: both t_2 and t_1 contain three components: the time for the request packet to be routed to search engine, the time for search engine to process the request, and the time to establish response stream connection. By subtracting t_1 from t_2, we eliminate factors affected by network transportation, assuming consecutive request packets are routed in similar routes. At the same time, we try to eliminate the effect of caches by sending unique queries, which are composed by combining keywords from different workloads.

Theoretically, if D denotes the set of all relevant documents and A is the set of answers returned by the search engine, then the main performance indicators are precision $|D \cap A|/|A|$ and recall $|D \cap A|/|D|$ (here $|A|$ denotes the number of elements in the set A). In rare cases of SU workload the set D can be known to expert evaluator, and $|A|$ can be a small value, so that precision and recall can be accurately computed and used for evaluation and comparison of competitive search engines. Unfortunately, in a general case of web search engines we cannot directly measure D and A, and exactly compute precision and recall. Below, we present approximate methods for measuring precision, recall and coverage.

3.2. Measurement of precision

Precision of search is defined as the ratio of the number of retrieved relevant documents $|D \cap A|$ and the number of all retrieved documents $|A|$. Most of existing studies are based on manual scoring systems to decide relevancy of documents. As our goal is to establish a framework of automatic repetitive low-cost tests, an alternative measurement is required.

A statistical analysis in [JAN03] shows that 73% of users only viewed 2 documents out of all retrieved documents. In addition, 53% of users were able to find wanted document using only one query. The results show that the quality of ranking is crucial to fulfill user search requirements, because the majority of users only browse very few documents in the result set. Furthermore, the existing ranking algorithms can ensure that statistically most relevant documents are ranked highest.

Given this result, we limited our analysis to first *k* documents returned by search engines. Suppose that our analysis includes *n* search engines and the number of reported URL's for each search engine is limited to *k*. Let $U_1,...,U_m$ be all the different URL's collected by n search engines and $k \le m \le nk$. Then, let $u_1^{(i)},...,u_k^{(i)}$ denote sorted URL's returned by search engine *i,* where index *r* in $u_r^{(i)}$ denotes the rank (*r*=1 is the most important and *r*=*k* is the least important according to search engine *i*). Let $R_1^{(i)},...,R_m^{(i)}$ be the ranking assigned to $U_1,...,U_m$ by search engine *i* according to the following formula:

$$R_j^{(i)} = \begin{cases} r_j^{(i)}, & U_j \in \{u_1^{(i)}, u_2^{(i)}, ..., u_k^{(i)}\} \\ k+1, & U_j \notin \{u_1^{(i)}, u_2^{(i)}, ..., u_k^{(i)}\} \end{cases} \quad , \quad j = 1,...,m$$

Here $r_j^{(i)} \in \{1,...,k\}$ is the ranking assigned to U_j by search engine *i*.

The final ranking of documents is based on the average value of $R_j^{(i)}$:

$$R_j = \frac{1}{n}\sum_{i=1}^{n} R_j^{(i)} \qquad j = 1,...,m$$

The first *k* documents with highest final ranking are selected as the final collection of all relevant documents to be studied, denoted as *U*. Finally, the precision P_i of search engine *i* can be defined as:

$$P_i = \frac{|U \cap \{u_1^{(i)},....,u_k^{(i)}\}|}{k}, \quad 0 \le P_i \le 1, \quad i = 1,...,n$$

3.3. Measurement of recall

Recall is the fraction of all relevant documents that is returned by a search engine. In a general case, it is not possible to collect all relevant documents from the web. However, recalls of different engines can be compared using relative ratios. For a specific search, let *D* be the (unknown) set of *all* relevant documents. Then $D \cap A_1,...,D \cap A_n$ are the sets of *relevant* documents returned by search engines 1,…,*n*. The recall of search engine *i* can be defined as follows:

$$C_i = \frac{|D \cap A_i|}{|D|}$$

For any pair of search engines *i* and *j:*

$$\frac{C_i}{C_j} = \frac{|D \cap A_i|}{|D|} \cdot \frac{|D|}{|D \cap A_j|} = \frac{|D \cap A_i|}{|D \cap A_j|}$$

For search engine *i*, we approximate $D \cap A_i$ by the result sets we collected during precision measurements:

$$D \cap A_i \approx U \cap \{u_1^{(i)},....,u_k^{(i)}\}$$

U is the collection of documents that all participating search engines voted to be relevant. A document returned by search engine *i* is considered relevant if and only if the document belongs to *U* as well.

3.4. Measurement of coverage

The *coverage* of a search engine can be determined as the total number of pages returned by the search engine. It is impractical to physically measure the absolute coverage of search engines because that would require us to develop a crawler that is superior to existing crawlers and is able to find all documents on the Internet. On the other hand, measuring relative coverage sizes of search engines is possible, as suggested in [BHA97]. We adopted the approximation formula proposed in [BHA97], which gives a relative ratio of coverage. Let A be the set of all pages indexed by the search engine E_1, and let B be the set of all the pages indexed by search engine E_2. Then, for any random URL *u* from Internet, the probability of $u \in A$ and the probability of $u \in B$ are proportional to sizes of A and size of B:

$$\frac{|A|}{|B|} \approx \frac{P_r(u \in A)}{P_r(u \in B)}$$

Using the conditional probability rule we have

$$P_r(u \in A) = \frac{P_r(u \in A \cap B)}{P_r(u \in B \mid u \in A)}$$

$$P_r(u \in B) = \frac{P_r(u \in A \cap B)}{P_r(u \in A \mid u \in B)}$$

$$\frac{|A|}{|B|} = \frac{P_r(u \in A \mid u \in B)}{P_r(u \in B \mid u \in A)}$$

This formula assumes that sampled documents are uniformly distributed, and the accuracy of the formula is affected by recalls of analyzed engines. A more complete discussion of this model can be found in [BHA97] where the authors used a lexicon of 40000 words and Yahoo to generate sample documents. In our measurement, we used so far over 10000 documents retrieved using PST search engines that are not among the competitive search engines. Although the sample does not necessarily represent the actual web word distribution, it is a fair-scaled, diverse selection over space and time. In addition, using independent PST search engines eliminates the possible bias caused by using one of the analyzed search engines as data source.

In all performance indicators (coverage, recall and precision) we normalize the measured values with the value that corresponds to the best system. In this way the best system has relative quality 1 (or 100%) and all other systems have lower values. These normalized (relative) values are then used as inputs for the evaluation criteria.

4. Search engine quality attributes

General models for software quality evaluation have been analyzed by many authors [BOE78, FEN97] and standardized by ISO [ISO91] and IEEE [IEE93]. The basic idea to evaluate software from the standpoint of its operation and upgrading is shown in Fig. 4.

1. **Operation of software product**
   - 1.1 Functionality (Suitability, Accuracy, Security, Interoperability, Compliance)
   - 1.2 Usability (Understandability, Learnability, Operability)
   - 1.3 Performance (Processing time, Throughput, Resource consumption)
   - 1.4 Reliability (Maturity, Fault tolerance, Recoverability)
2. **Upgrading of software product**
   - 2.1 Maintainability (Analyzability, Changeability, Stability, Testability)
   - 2.2 Portability (Adaptability, Installability, Conformance, Replaceability)
   - 2.3 Reusability (Structuredness, Conciseness, Self-descriptiveness, Device independence)

Fig. 4. General attributes of software quality

In the case of search engines the general software quality model must be modified to reflect specific features of search engines. First, from the end user standpoint it is only interesting to evaluate the operation of the software product. Consequently, the performance evaluation area must be expanded with the evaluation of information retrieval quality. At the same time, the reliability group is found not to be critical and can be omitted. This approach yields the following tree of attributes that affect the ability of search engines to satisfy user requirements:

1. Functionality
    - 1.1. General search methods
        - 1.1.1. Basic search
            - 1.1.1.1. Single keyword
            - 1.1.1.2. Multiple keyword
            - 1.1.1.3. Phrase
        - 1.1.2. Advanced search
            - 1.1.2.1. Boolean expression
            - 1.1.2.2. Number range search
            - 1.1.2.3. Negative terms
        - 1.1.3. Non-English search
        - 1.1.4. Multimedia search
            - 1.1.4.1. Images
            - 1.1.4.2. Video clips
                - 1.1.4.2.1. Categorization of material
                - 1.1.4.2.2. Popularity rating
                - 1.1.4.2.3. Search criterion
            - 1.1.4.3. Audio clips
        - 1.1.5. Extended search
            - 1.1.5.1. Case-sensitive search
            - 1.1.5.2. Common words exclusion
            - 1.1.5.3. Word variations (plural etc.)
            - 1.1.5.4. Use of synonyms
    - 1.2. Data filters
        - 1.2.1. Adult content
        - 1.2.2. Time filters
            - 1.2.2.1. Last update time
            - 1.2.2.2. Time interval
        - 1.2.3. Domain/page/link filtering
            - 1.2.3.1. Domain or site
            - 1.2.3.2. Location in page
            - 1.2.3.3. Linked pages
            - 1.2.3.4. Keyword frequency in page
            - 1.2.3.5. Pages from same site
        - 1.2.4. Miscellaneous filters
            - 1.2.4.1. Access rights
            - 1.2.4.2. Countries
            - 1.2.4.3. File types
            - 1.2.4.4. RSS file format support
            - 1.2.4.5. Similar pages
    - 1.3. Topic-specific search
        - 1.3.1. Technologies
        - 1.3.2. Academic
        - 1.3.3. Local life
        - 1.3.4. Maps
        - 1.3.5. Blog
2. Usability

- 2.1. Operability
  - 2.1.1. Visibility of functionality
  - 2.1.2. Ease of customization
  - 2.1.3. User interface quality
  - 2.1.4. Direct display of best match
- 2.2. Result presentation
  - 2.2.1. Customizable page size
  - 2.2.2. Customizable ranking of results
  - 2.2.3. Availability of cached results
- 2.3. Learnability
  - 2.3.1. On-line help (short references)
  - 2.3.2. User manual (book quality)
  - 2.3.3. Independent literature
  - 2.3.4. Tutorial (learning by example)
  - 2.3.5. Frequently asked questions

3. Performance of information retrieval
   - 3.1. Measured response time
     - 3.1.1. Time to return first page
     - 3.1.2. Time to return first 100 records
   - 3.2. Resource consumption
4. Quality of information retrieval
   - 4.1. Coverage
   - 4.2. Recall
   - 4.3. Precision

Each of the search engine evaluation attributes contributes to the engine ability to satisfy user requirements. The attributes are individually evaluated and the results of evaluation are aggregated to get a compound performance indicator for each search engine as a whole. This process takes into account desired logic relationships of inputs, and their relative importance according to the Logic Scoring of Preference (LSP) method presented below.

5. LSP method for system evaluation

System evaluation is a process of determining the extent to which a given system satisfies a set of requirements specified by a human decision maker (evaluator). This definition implies that the evaluated system has users, and users expect that the system can satisfy their requirements.

The LSP method [DUJ96, DUJ97, DUJ06] essentially consists of three steps:

- Creating a system attribute tree
- Defining elementary criteria
- Development of preference aggregation structure

The system attribute tree (SAT) is a decomposition structure that includes all system attributes that affect the ability of the evaluated system to satisfy user requirements. The SAT presented in Section 2 includes two types of attributes: compound attributes (that can be further decomposed) and terminal attributes (called *performance variables*) that cannot be further decomposed and can be directly evaluated.

Performance variables are evaluated using elementary criterion functions that map the values of performance variable to the corresponding preference score. The preference score is the degree of satisfaction of requirements and its range is from 0 to 1 (or 100%). For example, if the requirement is that a search engine response time t greater than $t_{\max}$ seconds is unacceptable, and the time that is less than $t_{\min}$ seconds satisfies completely user expectation, then the corresponding preference score E is

$$E = 100\min\{1, \max[0, (t_{\max} - t)/(t_{\max} - t_{\min})]\}$$
$$0 \le E \le 100\%$$

This mapping can be conveniently expressed using the preference scale shown in Fig. 5.

Similar elementary criteria can be defined for all n performance variables, and from these criteria we get an array of elementary preferences: $E_1,...,E_n$. Using these elementary preferences we can compute the global preference $E = L(E_1,...,E_n)$ that reflects the total satisfaction of all requirements.

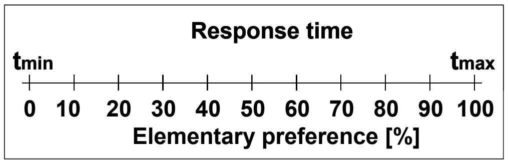


Fig. 5. An example of elementary criterion

To compute the global preference we must use preference aggregation structure that consists of superposition of appropriate aggregation operators. We assume that no system can be better than its best component, or worse than its weakest component. Consequently, the preference aggregators should be organized as means.

Some components of an evaluated system are always more important than other components. So, if we use means to aggregate elementary preferences, then the selected means must have adjustable weights. We use preference aggregators based on weighted power means; an aggregator that has k input preferences $e_1,...,e_k$ generates the output preference

$$e_0 = (w_1 e_1^r + ... + w_k e_k^r)^{1/r}$$
$$0 < w_i < 1, \quad w_1 + ... + w_k = 1$$
$$0 \le e_i \le 1, \quad i = 0,1,...,k$$

The weights $w_1,...,w_k$ reflect the relative importance of inputs $e_1,...,e_k$, and the exponent r reflects the logic properties of the aggregator. Such aggregator is graphically presented in Fig. 6 and a survey of all basic aggregators is presented in Table 5.

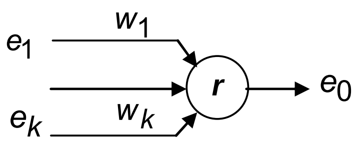


Fig. 6. A preference aggregator (power mean)

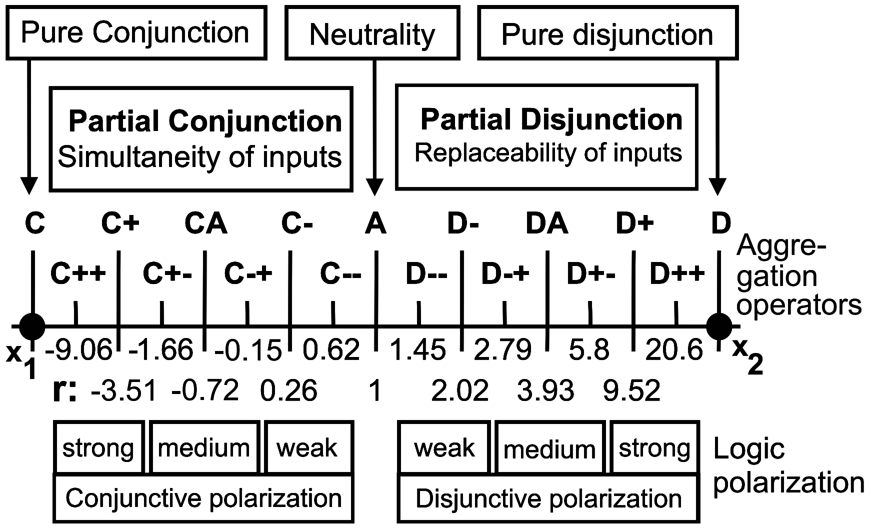


Fig. 7. Preference aggregators from ***and*** to ***or***

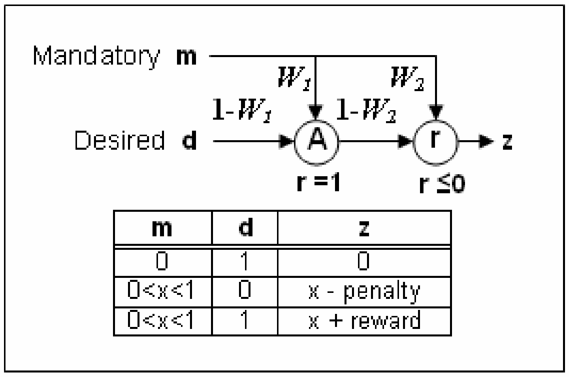
The most frequently used operator is partial conjunction, because in the majority of cases evaluators want simultaneous satisfaction of user requirements. If $r \le 0$ the partial conjunction is a model of mandatory requirements (any zero input generates zero output). Seventeen conjunctive and disjunctive aggregators with symbolic names C, C++, C+, C+-, CA, C-+, C-, C--, A, D--, D-, D-+, DA, D+-, D+, D++, D and corresponding values of the exponent r are presented in Fig. 7. If we aggregate preferences x_1 and x_2 then the aggregated value will be located between x_1 and x_2 approximately in location that is denoted by the position of aggregator (e.g. for A the result will be in the middle).

Among aggregators presented in Table 5, the second most frequently used operator is CPA. Its structure is shown in Fig. 8. This is an asymmetric operator: m is a mandatory input (if m=0 then the output z=0) and d is a desired input (if d=0 then the output z is only decremented by an adjustable penalty). The parameters r, W_1, and W_2 can be computed from the desired values of the mean penalty and mean reward.

Table 5. A survey of basic preference aggregators

Name	Description
NOT	**Negation** $(\bar{x} = 1 - x)$
AND $r = -\infty$	**Full conjunction**, the maximum level of simultaneity.
ANDOR (PC) $-\infty < r < 1$	**Partial conjunction** (PC), a spectrum of simultaneity levels. All input preferences must be to some extent simultaneously satisfied.
AM (CDN) $r = 1$	**Arithmetic mean** (AM), or Conjunctive/ Disjunctive Neutrality (CDN). AM models a perfect balance of simultaneity and replaceability. All inputs are desired, but no one is mandatory or sufficient.
ORAND (PD) $1 < r < +\infty$	**Partial disjunction** (PD), a spectrum of replaceability levels. Each input can be used to partially compensate the lack of remaining inputs.
OR $r = +\infty$	**Full disjunction**, the maximum level of replaceability.
CPA	**Conjunctive Partial Absorption**. A combination of mandatory and desired inputs. The mandatory input (m) must be (at least partially) satisfied. Assuming m>0, the desired input (d) can partially compensate an insufficient level of m.
DPA	**Disjunctive Partial Absorption**. A combination of sufficient and desired inputs. The sufficient input (s) can fully compensate the lack of desired input (d). The desired input can partially compensate the lack of sufficient input.

More details about the mathematical aspects of the LSP method can be found in [DUJ05]. By systematically applying the aggregation of preferences according to the structure of SAT and desired logic properties of specific aggregators we eventually compute the global preference, i.e. a compound indicator of the ability of the evaluated system to satisfy all user requirements. This indicator is then used for evaluation and comparison of search engines, as shown in the next section.

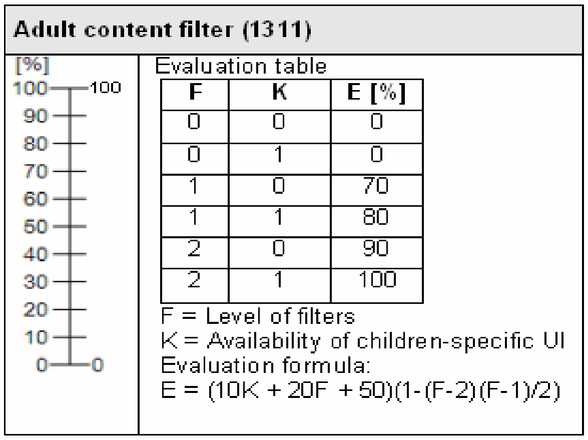


m	d	z
0	1	0
0<x<1	0	x - penalty
0<x<1	1	x + reward

Fig. 8. The CPA aggregator and its properties

6. LSP criterion for evaluation of search engines

In this section we use the LSP criterion for search engine evaluation to compare four major PST search engines: Yahoo! Search, Ask, Google, and MSN. Our LSP criterion consists of 52 elementary criteria, which include 83 individual quality attributes collected using various measurements and analyses. Figure 9 shows three examples of typical elementary criteria.

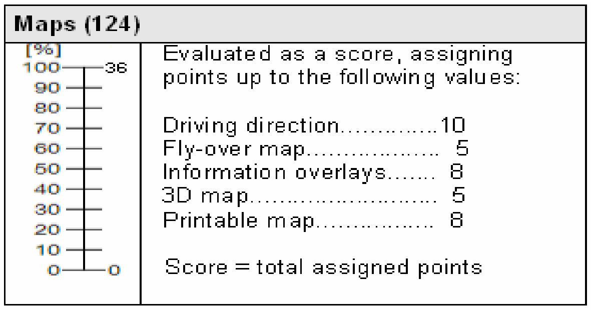


Evaluation table

F	K	E [%]
0	0	0
0	1	0
1	0	70
1	1	80
2	0	90
2	1	100

F = Level of filters
K = Availability of children-specific UI
Evaluation formula:
E = (10K + 20F + 50)(1-(F-2)(F-1)/2)

Fig. 9-1. Adult content filter



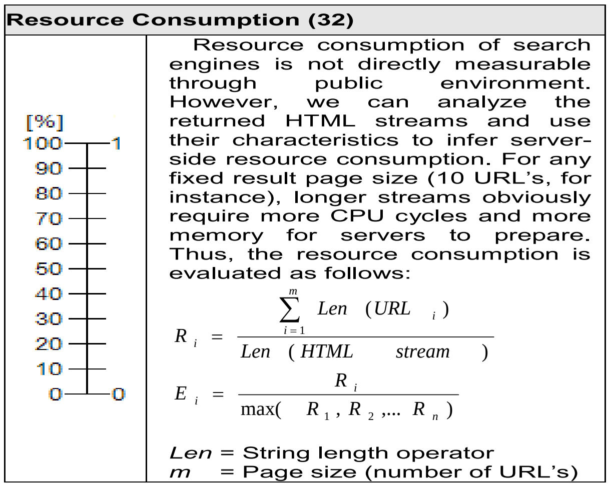
Maps (124)

Evaluated as a score, assigning points up to the following values:

Driving direction…………..	10
Fly-over map………………	5
Information overlays…….	8
3D map………………………	5
Printable map……………..	8

Score = total assigned points

Fig. 9-2. Maps



Resource Consumption (32)

Resource consumption of search engines is not directly measurable through public environment. However, we can analyze the returned HTML streams and use their characteristics to infer server-side resource consumption. For any fixed result page size (10 URL's, for instance), longer streams obviously require more CPU cycles and more memory for servers to prepare. Thus, the resource consumption is evaluated as follows:

$$R_i = \frac{\sum_{i=1}^{m} Len\ (URL_i)}{Len\ (HTML\ stream)}$$

$$E_i = \frac{R_i}{\max(R_1, R_2, \ldots R_n)}$$

Len = String length operator
m = Page size (number of URL's)

Fig. 9-3. Resource Consumption

All elementary criteria that evaluate performance and quality of information retrieval can be organized ac combination of GP and SU workload. In the case of coverage, recall, and precision, the results of SU workload are generated by expert evaluators. These results can be aggregated with results for GP workload collected by SEben. The GP and SU results can be aggregated using complementary weights (*w* and 1-*w*, 0<*w*<1) that can shift the emphasis of evaluation continuously from GP to SU type of user. The default approach characterized by *w*=1/2 reflects the standpoint that high volume deserves equal attention as the high significance of search.

Elementary criteria are used to compute normalized preference values (from 0 to 100%) for all 52 inputs. The next step in the evaluation process is to organize the logic aggregation of preference structure (Fig. 10). This structure uses aggregators shown in Figures 7 and 8 to compute the global preference score for each of competitive search engines. The global preference score corresponds to the evaluated system as a whole. The aggregation of preferences is a process of averaging that reflects desired formal logic and semantic relationships between aggregated values.

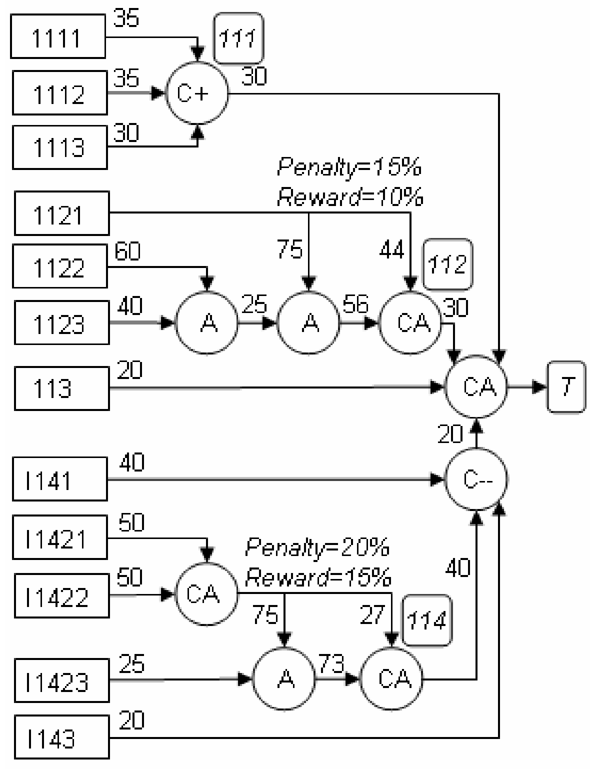


Fig. 10-A. General search method (1/2)

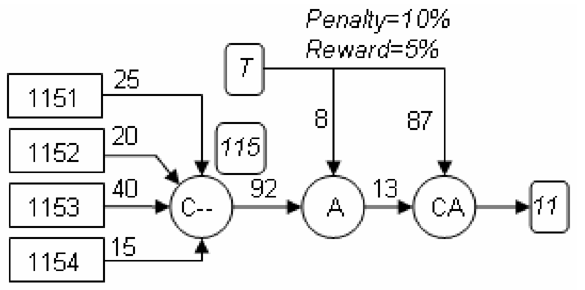


Fig. 10-B. General search method (2/2)

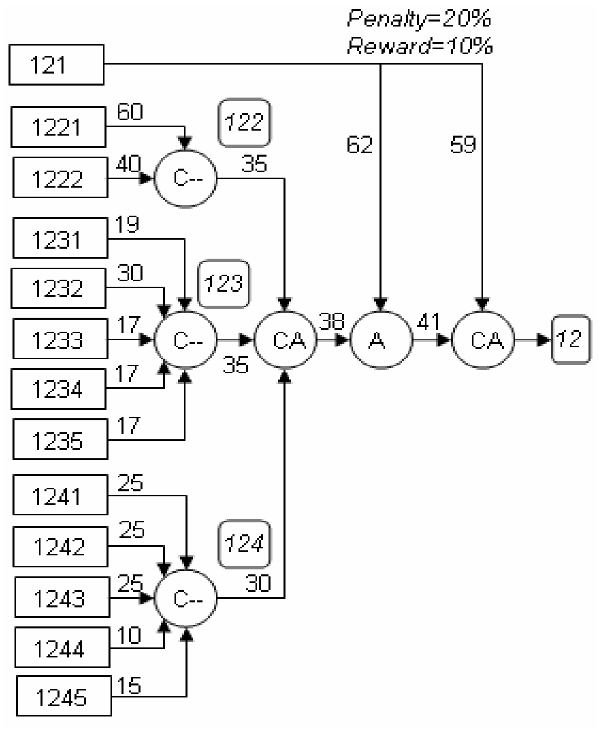


Fig. 10-C. Data filters

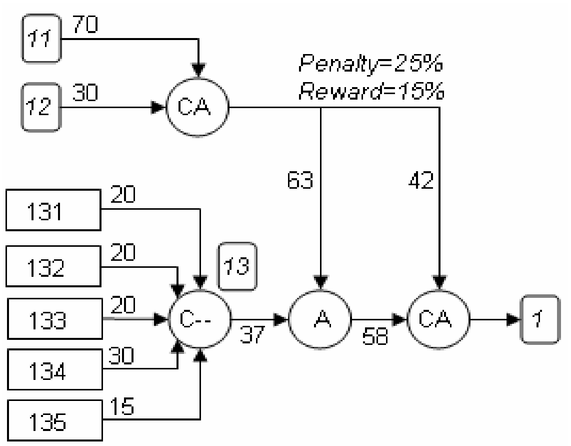


Fig. 10-D. Functionality

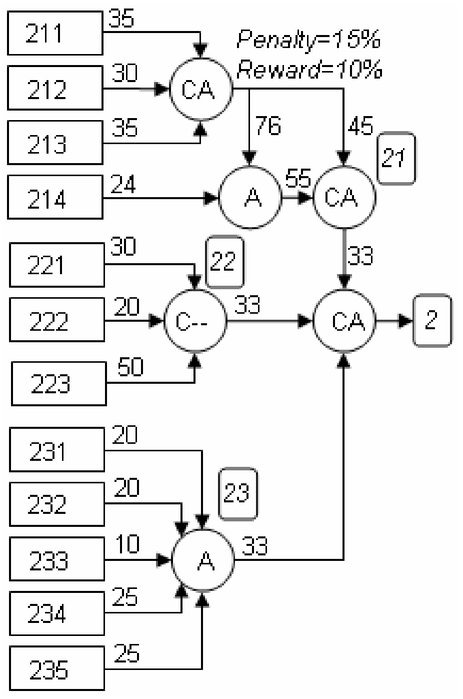


Fig. 10-E. Usability

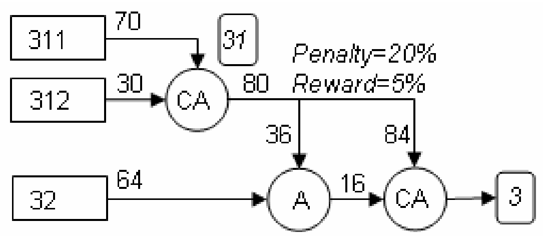


Fig. 10-F. Performance of information retrieval

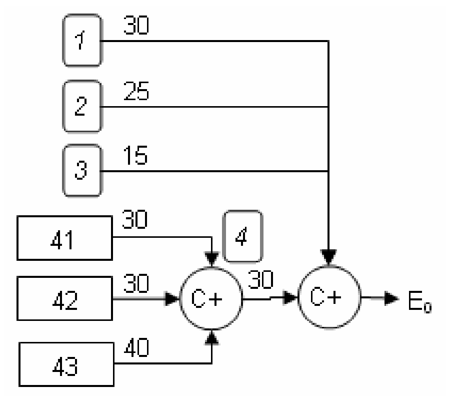


Fig. 10-G. Global preference

The final results of evaluation are presented in Table 6 and Fig. 11. These results reflect the status of evaluated systems in June 2006. Google has a leading position by satisfying 86.67% of requirement. MSN and Yahoo are in a close match with only 2% of difference in global preference score. Ask ranks last with a rather low global score (54.72%).

Table 6. System preferences [%]

	Google	MSN	Yahoo	Ask
Global preference	**86.67**	**77.85**	**75.74**	**54.72**
Functionality	***83.10***	63.38	62.98	39.44
Usability	82.84	***86.44***	77.35	73.38
Performance	87.94	81.84	***91.45***	76.98
Quality of inf. retrieval	***92.57***	82.80	80.27	56.04

A very high information retrieval quality (92.57%) and rich functionality help *Google* to establish the dominating position. In addition, *Google* has the best balance of quality – all major groups of *Google* features satisfy more than 80% of evaluation requirements. On the other hand, *Google* could improve some usability components. E.g., one of the negative aspects in *Google*'s UI is the "I'm Feeling Lucky" button. Regardless its long history, this label makes no sense and the button creates confusion to new users. There is also enough space for further development of functionality.

MSN Search has the best usability and ranks second because it dominates the third-placed *Yahoo! Search* in all areas except performance. In particular, *MSN* attains a higher coverage of page indexes, and gets a high usability score due to its easy-to-use customized ranking feature. On the other hand, *MSN* has usability areas that can be improved. For instance, in the case of function visibility, *MSN* gets a low score due to its strange design of putting some search categories on top of the page, while putting some other categories in a drop-down list besides the “Search” button. *MSN* offers a good balance of usability, performance, and information retrieval quality. However, its functionality is insufficient and could be significantly improved.

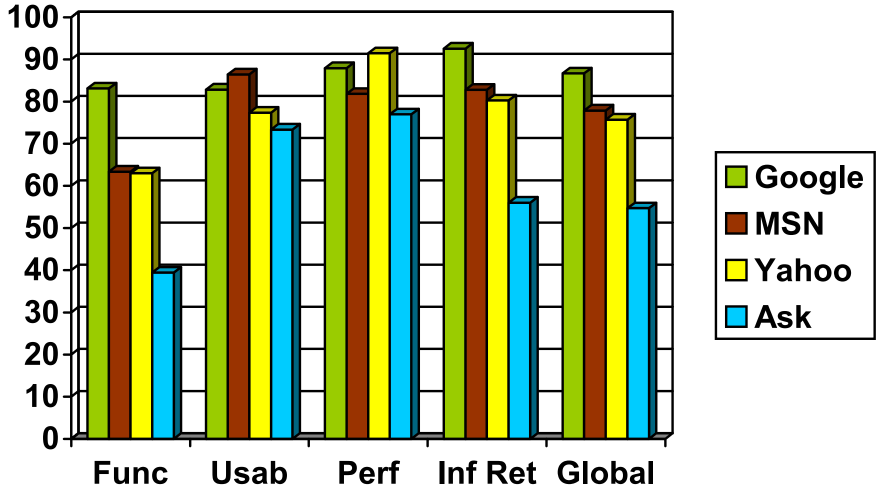


Figure 11. Final results of evaluation of Google, MSN, Yahoo, and Ask

Yahoo! Search dominates the performance area. It ranks third, close to *MSN Search.* It has good results in the average server response time. In addition, *Yahoo* result pages show a good balance of information richness and efficiency. Compared to *Google*, *Yahoo! Search* provides less topic-specific search options and less advanced search options such as word variations and word synonyms. *Yahoo! Search* functionality could be significantly improved and main groups of features are insufficiently balanced.

Ask got the lowest scores in all subsystems, resulting in the last position. Its weakest subsystems are functionality and information retrieval quality. In the area of information retrieval quality, *Ask*’s low index coverage significantly hurts its global preference score. In addition, *Ask* provides fewer search options and filters comparing with other three search engines. In the performance area, *Ask*’s resulting HTML source documents are badly organized, with lots of spaces and empty lines. The average length of *Ask*’s result pages is 4 times longer than the length of *MSN*’s result pages. Ask is competitive in the area of usability and performance, but needs improvements in the areas of functionality and information retrieval quality. Its poor balance of features indicates an imbalanced development strategy.

7. Conclusions

Evaluation and selection of software systems is a complex problem characterized by a large number of heterogeneous inputs. Many of input attributes are important but difficult for quantification and measurement. Both IEEE and ISO standards for software quality metrics provide initial guidelines for defining general-purpose software quality attributes. The presented search engine evaluation criterion shows that the LSP method is suitable for building complex criteria on top of basic concepts provided by software quality standards.

Evaluation of search engines is one of complex software evaluation problems. It includes a spectrum of functionality, usability and performance inputs including more than 80 individual quality attributes. In the case of performance, it was necessary to develop a specialized tool for measurement of response times, resource consumption, and the quality of information retrieval.

Basic advantages of using the LSP method for software evaluation and comparison can be summarized as follows:

- Ability to aggregate any number of heterogeneous inputs, including functionality, usability, performance, and cost indicators.
- Ability to express necessary logic relationships between inputs, such as simultaneity, replaceability, mandatory and desired inputs, etc.
- Ability to express both formal logic and semantic components of the decision model.
- Unrestricted applicability in all regions of input attributes space (the method generates meaningful results for any combination of input values).
- Ability to automatically reject systems that do not satisfy any of mandatory requirements.
- Ability to explain and justify evaluation results in a simple and understandable way.

Our evaluation of four leading search engines provides insight into the state of the art in the Internet search technology. *Google* offers the best search technology, and satisfies approximately 10% more requirements than the closest competitors (*MSN* and *Yahoo*). The difference between *MSN* and *Yahoo* is small. Modest functionality and quality of information retrieval are main reasons why *Ask* satisfies roughly half of our requirements. Our model also shows that no search engine seems to be perfect and there is space for improving even the best of them. The obtained evaluation results can be used to suggest what improvements of search technology would be the most beneficial for each of competitors.

The area of search engines is extremely dynamic, characterized by fast changes in offered services, the scope of search, functionality, usability, performance, and business relationships between competitors. In such an environment we cannot expect that evaluation criteria or evaluation results can last very long. Consequently, the proposed criterion will need permanent refinement and updating, according to functionality of new search engines and other developments. However, we believe that the proposed approach, the structure of the evaluation criterion, and the way of aggregating functionality attributes with usability, quality, and performance of IR will stand the test of time. Therefore, the future work should be focused on expanding and improving the performance measurement tool, refinement and evolutionary adjustment of the LSP criterion, development of specialized criteria for specific areas of search, and the coverage of all currently operational search engines.

8. References

1. [BAE99] Baeza-Yates, R. and B. Ribeiro-Neto, Modern Information Retrieval. Addison-Wesley, 1999.
2. [BHA97] Bharat, K. and A. Broder, A Technique for Measuring the Relative Size and Overlap of Public Web Search Engines, DIGITAL Systems Research Center, 1997
3. [BOE78] Boehm, B.W at al., *Characteristics of Software Quality.* North Holland, 1978.
4. [BRI98] Brin, Sergey and Page, Lawrence. The Anatomy of a Large-Scale Hypertextual Web Search Engine. http://www-db.stanford.edu/~backrub/google.html.
5. [CHA03] Chakrabarti, S., Mining the Web: Analysis of Hypertext and Semi Structured Data. Elsevier, 2003.
6. [CHU96] Chu H. and M. Rosenthal, Search Engines for the World Web Web: A Comparative study and Evaluation Methodology, ASIS 1996 Annual Conference Proceedings October 19-24, 1996
7. [DUJ96] Dujmović, J.J., A Method for Evaluation and Selection of Complex Hardware and Software Systems. The 22nd International Conference for the Resource Management and Performance Evaluation of Enterprise Computing Systems. CMG 96 Proceedings, Vol. 1, pp. 368-378, 1996.
8. [DUJ97] Dujmović, J.J., Quantitative Evaluation of Software. Proceedings of the IASTED International Conference on Software Engineering, edited by M.H. Hamza, pp. 3-7. IASTED/Acta Press, 1997.
9. [DUJ05] Dujmović, J.J., Continuous Preference Logic for System Evaluation. Proceedings of Eurofuse 2005, edited by B. De Baets, J. Fodor, and D. Radojević, pp. 56-80. ISBN 86-7172-022-5, Institute “MihajloPupin”, Belgrade, 2005.
10. [DUJ06] Dujmović, J.J. and Hajime Nagashima, LSP Method and its Use for Evaluation of Java IDE’s. International Journal of Approximate Reasoning, Vol. 41, No. 1, pp. 3-22, 2006.
11. [FEN97] Fenton, N.E. and S.L. Pfleeger, Software Metrics. ITP, 1997.
12. [GAR06] Garcia, E., Information Retrieval Tutorials.
13. http://www.miislita.com/information-retrieval-tutorial/indexing.html, 2006
14. [GOG06a] http://www.google.co.uk/press/zeitgeist/ archive.html
15. [GOG06b] ttp://www.google.com/press/zeitgeist2003/graph_sars.gif
16. [HAW06] Hawking, D., Web Search Engines. Computer Vol. 39, No.6 (Part 1) and Vol. 39. No. 8 (Part 2), 2006.
17. [IEE93] IEEE Standard for a Software Quality Metrics Methodology. IEEE Std 1061-1992, ISBN 1-55937-277-X. Published 1993.

18. [ISO91] ISO/IEC, International Standard ISO/IEC 9126 (E). Information technology – Software product evaluation – Quality characteristics and guidelines for their use. International Organization for Standardization, First edition,1991.
19. [JAN03] Jansen, B.J. and A. Spink, An Analysis of Web Documents Retrieved and Viewed, The 4th International Conference on Internet Computing. Las Vegas, Nevada, p. 65-69. 23-26 June 2003.
20. [SEG06] http://searchengineguide.com/ , 2006
21. [SEW06] http://searchenginewatch.com/ , 2006
22. [SPA97] Sparck Jones, K. et al., Readings in Information Retrieval. Morgan Kaufmann Series in Multimedia Information and Systems, Elsevier, 1997.
23. [SUL06] Sullivan, D., Nielsen NetRatings Search Engine Ratings, http://searchenginewatch.com/
24. reports/article.php/2156451, 2006
25. [TAN03] Tang, M. C. and Y. Sun, Evaluation of Web-Based Search Engine Using User-Effort Measures, School of Information, Communication and Library Studies, Rutgers University, 2003.
26. [VAU04] Vaughan, L., New Measurements for Search Engine Evaluation Proposed and Tested, Information Processing and Management 40 (2004) 677-691

Jozo Dujmović was born in Dubrovnik and received his BSEE, MS, and Sc.D. from the University of Belgrade. He is Professor of Computer Science and former Chair of Computer Science Department at San Francisco State University. His teaching and research activities are in the areas of soft computing, software metrics, and computer performance evaluation. In 1973 he introduced the concepts of andness and orness and developed decision models based on functions that allow continuous transition from conjunction to disjunction. He is the author of more than 120 refereed publications, recipient of three best paper awards, and a Senior Member of IEEE. He served as General Chair of IEEE MASCOTS 2000 and as General Chair of ACM WOSP 2004.

Before his current position at San Francisco State University, Dr. Dujmović was Professor of Computer Science at the University of Belgrade, University of Florida (Gainesville), University of Texas (Dallas), and Worcester Polytechnic Institute. In addition, he was teaching in the graduate Computer Science programs at the National Universities of San Luis and Jujuy (both in Argentina). At the University of Belgrade, he served as Chairman of Computer Science Department, and as founding Director of the Belgrade University Computing Center. His industrial experience includes work in the Institute "M. Pupin" in Belgrade, and consulting in the areas of decision methods, performance evaluation, and software design.

Haishi Bai was born in China and received his BS in Computer Science from Jilin Institute of Technology. He is currently an MS candidate in Software Engineering program at San Francisco State University. He has worked as a professional software engineer and software architect in mainland China, Hong Kong, and Silicon Valley. He is the principal developer and designer of projects in various areas such as multi-dimensional databases, e-Business, ERP, OLAP, system integrations, enterprise security and compliance, mobile platforms and embedded systems. He is the coauthor of a book about e-Business website construction, and he is the coauthor of two published papers in the area of software engineering. Mr. Bai is the lead .NET architect and developer at ShotSpotter, Inc., the world leader in gunshot location systems and technology, where he is in charge of all software integration and collaboration technologies.