Automated functional testing of online search services [*],[**]

Zhi Quan Zhou[1],[†],[‡], ShuJia Zhang[1], Markus Hagenbuchner[1],
T. H. Tse[2], Fei-Ching Kuo[3] and T. Y. Chen[3]

[1] *School of Computer Science and Software Engineering, University of Wollongong,
Wollongong, NSW 2522, Australia*
[2] *Department of Computer Science, The University of Hong Kong, Pokfulam,
Hong Kong*
[3] *Centre for Software Analysis and Testing, Swinburne University of Technology,
Hawthorn, VIC 3122, Australia*

SUMMARY

Search services are the main interface through which people discover information on the Internet. A fundamental challenge in testing search services is the lack of oracles. The sheer volume of data on the Internet prohibits testers from verifying the results. Furthermore, it is difficult to objectively assess the ranking quality because different assessors can have very different opinions on the relevance of a Web page to a query. This paper presents a novel method for automatically testing search services without the need of a human oracle. The experimental findings reveal that some commonly used search engines, including Google, Yahoo!, and Live Search, are not as reliable as what most users would expect. For example, they may fail to find pages that exist in their own repositories, or rank pages in a way that is logically inconsistent. Suggestions are made for search service providers to improve their service quality.

KEY WORDS: software testing; metamorphic testing; verification and validation; Internet search

† Correspondence to: Dr Zhi Quan Zhou, School of Computer Science and Software Engineering, University of Wollongong, Wollongong, NSW 2522, Australia.
‡ E-mail: zhiquan@uow.edu.au

1. INTRODUCTION

The World Wide Web [*] is the largest repository of digital information ever produced by the human race. Owing to its size, the finding or retrieval of specific information from the Web is an increasingly difficult task. Finding information on the Web is made possible through search services such as Google, Yahoo!, Live Search, and numerous others.

Among these service providers, Google is currently the largest one providing the most comprehensive coverage of the Web. Even then, Google only has partial information about the contents of the Web. For example, it is known that Google indexed 25 billion Web pages in 2006 whereas the estimated size of the Web had already exceeded 200 billion pages in 2005 [1] and reached 447.98 billion by early 2007 [2]. As a consequence any attempt to search for information on the Web will only yield the results covered by the search service provider.

A large number of users rely on Web search services to retrieve reliable information. While it is known that search service providers cannot provide full Web coverage, the extent to which such large-scale search services are reliable remains unknown. This paper proposes a means by which Web search services can be tested. It will be shown that some of the largest Web search services provide inconsistent responses, and that these inconsistencies vary over time. [†] It will be shown that these inconsistencies can be of significance to users. It will also be shown that these inconsistencies are due to (possibly faulty) software features rather than due to the property of the Web domain.

A main contribution of this paper is the proposal of a metric by which Web search services can be tested automatically. This is because conventional evaluation methods for information retrieval (IR) systems are not suitable for Web search services: First, when it comes to the evaluation of the ranking quality of Web search results, the judgment has always been very subjective as different assessors can have very different opinions on the relevance of a Web page to a query. Second, conventional evaluation measures, namely *precision* and *recall* [3, 4], are difficult to apply on the live Web. Given a search criterion, let A be the set of all items retrieved, $R \subseteq A$ be the set of relevant items retrieved, and R' be the set of relevant items in the database but not retrieved. *Precision* is defined as $|R| \div |A|$, and *Recall* is defined as $|R| \div (|R| + |R'|)$. It is infeasible to measure recall for Web search engines because it requires the knowledge of not only the retrieved records but also the records in the database not retrieved. For example, Chu and Rosenthal [5] reported that 'Recall . . . is deliberately omitted from this study because it is impossible to assume how many relevant items there are for a particular query in the huge and ever changing Web system.' Similarly, Su [6] also pointed out that 'Recall was not used at all due to the difficulty of its implementation in the vast Web information environment.' To alleviate this problem, Clarke and Willett [7] proposed an approach using relative recall instead of absolute recall, which searches with different engines and then pools the relevant documents for computation. In short, one of the two major evaluation measures is often ignored or sidestepped when testing Web search engines.

The difficulty in testing search services is due to the *oracle problem*, which is a fundamental challenge in software testing. An *oracle* is a mechanism against which a tester can decide whether

---

[*] The paper will use the term *Web* when referring to the World Wide Web.

[†] In this paper, the term 'inconsistency' refers to the phenomenon that a search service gives inconsistent responses to different but logically related queries.

the outcomes of test case executions are correct [8]. Such an oracle is not available when testing a search service on the live Web. Despite the efforts made by various researchers in evaluating IR systems [9, 10, 11, 12, 13, 14], the oracle problem remains fundamentally difficult.

To alleviate the oracle problem in testing online search services, this paper proposes an innovative method that is quite different from conventional strategies. Rather than attempting to verify each *individual* response, the proposed method focuses on the logical consistency among *multiple* responses. Here, 'logical consistency' means that certain relations should hold in the search engine's responses to multiple and related queries. To behave in a logically consistent way is a necessary condition for a search service to be seen as 'good.' The proposed method automatically tests search services against consistency properties *despite the absence of a tangible oracle* of the search results. Relations among multiple executions are known as *metamorphic relations* [15]. The consistency properties discussed in this paper are instances of metamorphic relations. The proposed method can be used for both *quality assessment* and *quality improvement* of search services.

The remainder of this paper is organized as follows: Section 2 presents a method to automatically test search services in terms of logical consistency of returned counts and contents, together with the experimental results. The experiments are focused on the AND-, OR-, and EXCLUDE-relations. Anomalies are revealed for all the three search engines under investigation, and hints are provided for search service providers to debug or tune their systems. In Section 3, a method to automatically test search services in terms of ranking quality is presented, together with the experimental results. The relationship between the logical consistency properties and users' perceived quality of ranking is also discussed, and hints are provided for search engine designers to improve the ranking quality of their products. Section 4 presents the results of further experiments, which investigate several interesting issues, such as the impact of different languages on ranking quality, the impact of commercial interests on ranking quality, the change of ranking quality with time, and the relationship between ranking quality and page quality. Section 5 makes further discussions on the presented testing method and findings. Section 6 concludes the paper and discusses future work.

2. TESTING SEARCH SERVICES IN TERMS OF THE QUALITY OF RETURNED COUNTS AND CONTENTS

As an example of the inaccuracy of search results, consider an *advanced* search in the ACM Digital Library (http://portal.acm.org), for the following paper: 'In black and white: an integrated approach to class-level testing of object-oriented programs' (search by title, see Figure 1(a)). The system returned no results (Figure 1(b)). Surrounding the keyphrase with double quotation marks (as suggested by the system) did not change this outcome. However, when performing a *general* search on http://portal.acm.org using the same keyphrase (Figure 1(c)) produces a find. Obviously, the paper does exist in the database, but cannot be located in the first two attempts. This indicates a potential fault in the system.

Apart from general users, researchers from many fields have been doing research by consulting generic Web search engines for some time now. It has become appealing for some researchers to use the Web as a data source, and to use a Web search engine to find frequencies and probabilities of some phenomena of interest [16]. In the field of language analysis and generation, for instance, some researchers have been using the counts returned by Web search engines for their research [17, 18, 19,
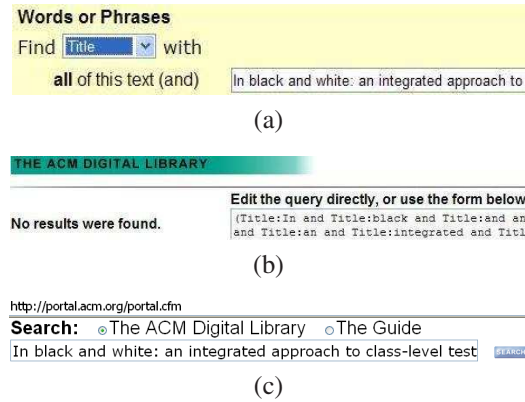
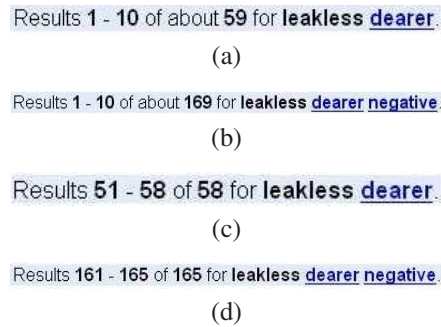Figure 1. Anomaly in searching the ACM Digital Library.

Figure 2. Anomaly in Google search.

20, 21]. The assumptions that Web search engines are trustworthy and that the counts are reliable, however, remain to be validated. There are numerous other areas that can be affected by inaccuracies in the functionality of search services. User acceptance is reduced if search services are perceived to be unreliable. Apart from obvious economic impacts, the acceptance of search services and therefore the Web as a medium for information distribution may be at stake.

As a second example, consider a search www.google.com for 'leakless dearer' (according to the specification in Google, spaces among words imply the AND-relation). The system returns 'about 59' results (Figure 2(a)). How can testers know whether Google correctly counted all the relevant Web pages indexed in its database? This paper proposes that, instead of checking the correctness of each individual response, many necessary relations among *multiple* responses (executions) of the search engine can be identified.

Let X be a search criterion (such as a condition 'Page contains string S.'), $Pages(X)$ be the set of pages satisfying X, and $Count(X)$ be $|Pages(X)|$, that is, the number of pages satisfying X. A useful general relation can be identified as R_{COUNT}: if $Pages(X_2) \subseteq Pages(X_1)$, then $Count(X_2) \leq$

$Count(X_1)$. A special case is R_{OR}: if $A_2 \equiv (A_1 \ OR \ B)$, then $Count(A_1) \leq Count(A_2)$, which means that the number of pages satisfying condition A_1 should be less than or equal to the number of pages satisfying condition A_1 or condition B. Similarly, another special case is R_{AND}: if $A_2 \equiv (A_1 \ AND \ B)$, then $Count(A_2) \leq Count(A_1)$. Yet another relation is $R_{EXCLUDE}$: if $A_2 \equiv (A_1 \ AND \ \overline{B})$, then $Count(A_2) \leq Count(A_1)$, where $(A_1 \ AND \ \overline{B})$ means that condition A_1 is satisfied but not condition B. Most search engines support these kinds of search functions. It should be pointed out that many other relations can be identified for testing, and many of them may not involve logic operators. This paper does not intend to give a comprehensive list of such relations. Furthermore, the relations checked by the proposed method are only necessary properties (and not sufficient properties) for a search engine to behave correctly. This is indeed a limitation of all testing methods. Note that in the result page of Google and Yahoo!, there is sometimes a statement 'In order to show you the most relevant results, we have omitted some entries very similar to the XXX already displayed. If you like, you can repeat the search with the omitted results included.' In the experiments, the option 'repeat the search with the omitted results included,' which effectively disables filters, is always chosen.

Another Google search for the keywords 'leakless dearer negative' resulted in 'about 169' pages found as is shown in Figure 2(b). This search was conducted *immediately after* the first search for 'leakless dearer.' Although it is not known whether either of the two results is accurate, an anomaly is identified. The results grossly violated the expected R_{AND}, which requires that the number of pages containing the three keywords 'leakless dearer negative' be smaller than or equal to the number of pages containing the first two keywords 'leakless dearer.'

2.1. Validity of the methodology

Does the violation of the logical consistency property indicate flaws or software faults in the search engine? First, various possibilities must be considered. (1) Was correct syntax used in the search? The help page of Google states: '**Automatic "and" queries** By default, Google only returns pages that include all of your search terms. There is no need to include "and" between terms. ... To restrict a search further, just include more terms.' Hence, the syntax was correct and the test was valid. (2) Was this anomaly caused by the dynamic nature of the Web and the search engine? As the database of the search engine changes dynamically, it might happen that there were originally 59 pages indexed in the Google database, but updated to 169 a second later. The test was, therefore, repeated immediately after detecting the anomaly. The same anomaly was identified in the second attempt. [‡] (3) Could the anomaly be due to approximate results given by search engines? This would be a plausible explanation if the differences were small, because almost all major search engines return approximate results. However, this argument would not hold for very large differences. In fact, for all the search engines tested and all the logical consistency properties employed, a great number of anomalies with even larger differences have been detected, such as the one in Microsoft's Live Search shown in Figure 3: A query for 'GLIF'

[‡] Having said that, it must be pointed out that many of the anomalies shown in this paper have disappeared since this paper was written. This would really be due to the dynamic nature of the Web and the search engines. However, while earlier anomalies disappeared, new anomalies are showing up. Every time the testing tool is run with new inputs, new repeatable anomalies are always found. This indicates that the root causes for the anomalies persist despite the change of anomaly-causing queries. It should also be noted that the possibility of a negative impact of database/cache updating on search result consistency has not been ruled out. This is further investigated in Section 4.3.
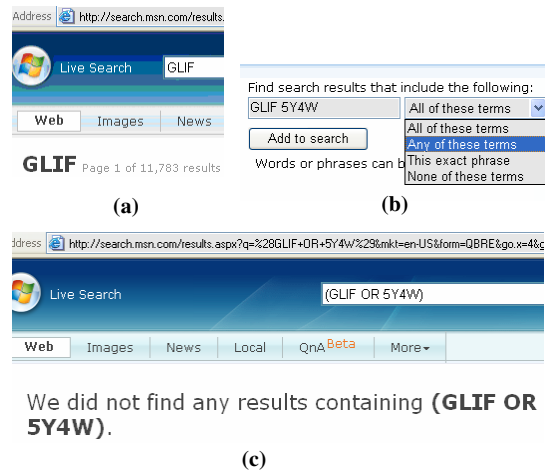
Figure 3. Anomaly in Live Search.

gave 11 783 results, as shown in Figure 3(a). An immediate second query for 'GLIF' OR '5Y4W' (Figure 3(b)) gave, surprisingly, zero results (Figure 3(c)). This obviously violated the expected R_{OR} property. All the anomalies reported in this paper were repeatable at the time of experiment, and all the queries used correct syntax. (4) For large search engines, the result counts may be estimated from statistics or samples about the query terms. This estimate may change between result pages. The last entries in the result pages, as shown in Figures 2(c) and 2(d), were therefore browsed. It is interesting to compare Figure 2(a) against Figure 2(c): The former reports 'of about 59' whereas the latter reports 'of 58.' Note that the word 'about' in the former disappeared in the latter. This means that Figure 2(c) can be seen as the accurate result given by Google. This final count '58' differs only slightly from the initial estimate. A similar observation is made for Figures 2(b) and 2(d). It is further found that out of the 165 URLs returned by Google for 'leakless dearer negative,' 119 did not appear in the set of 58 URLs returned for 'leakless dearer.'

Can the violations of the consistency properties be caused by intentional design decisions rather than design flaws or software faults? For example, is it possible that such a violation was caused by an internal leniency mechanism of the search engine? That is, when the search engine finds the number of pages containing term A is too small, it becomes more lenient when searching for pages containing both A and B to avoid returning too few or zero results. Such an assumption is not supported by the findings made in this research because significant inconsistencies were observed even when millions of pages matched a query. Another possible cause is that there might be a threshold defined by the ranking algorithm of the search engine: a Web page is counted only when it both contains the keywords and has a relevance score above the threshold. Therefore, it is possible that when searching for 'leakless dearer,' only 58 pages had relevance scores above the threshold; whereas when searching for 'leakless dearer negative,' the additional keyword 'negative' revealed more information about the user's intention and, as a result, more than 58 Web pages received relevance scores above the threshold. If this is indeed the case, then the rate of violations of the R_{AND} property should be quite high. However, as will be introduced shortly, experiments show that for all the search engines tested, their R_{AND} anomaly rates

were quite low. Hence, it is unlikely that the anomalies were caused by such a threshold mechanism. Furthermore, if there was indeed such a threshold mechanism, users should have been informed of the mechanism rather than being misled to believe that only 'XXX pages were found.'

The validity of the method can also be discussed from the perspectives of software *verification* and *validation*. Verification is defined as [22]:

> 'Verification' is checking the consistency of an implementation with a specification. Here, 'specification' and 'implementation' are roles, not particular artifacts. For example, an overall design could play the role of 'specification' and a more detailed design could play the role of 'implementation'; checking whether the detailed design is consistent with the overall design would then be verification of the detailed design. Later, the same detailed design could play the role of 'specification' with respect to source code, which would be verified against the design.

For all the search engines tested, their specifications are available online, and all these specifications state that the search engine will count/return *all* pages (indexed in the database) that meet the user's search criteria. The logical consistency properties used in the presented method are necessary properties identified from these specifications. Any violation of any logical consistency property, therefore, means a *software failure* in the verification, where 'software failure' is defined as [23] 'external, incorrect behavior with respect to the requirements or other description of the expected behavior.' It should be noted, however, that there are various other possible reasons for the failure apart from software faults. For instance, though unlikely, it is theoretically possible that the two searches involved in the violation of the logical consistency property were conducted at different servers with different databases and, hence, the two search results were not consistent. Nevertheless, this kind of lower-level design decisions cannot replace the top-level specifications during the verification of the entire system.

Next, consider the problem from the perspective of software validation, which is defined as [22]:

> Assessing the degree to which a software system actually fulfills its requirements, in the sense of meeting the user's real needs, is called *validation*.

Note the difference between verification and validation [22]:

> In every case, though, verification is a check of consistency between two descriptions, in contrast to validation which compares a description (whether a requirements specification, a design, or a running system) against actual needs.

The logical consistency properties are desirable properties that reflect users' actual needs, as it is natural for users to expect that a good search engine should behave in a logically consistent way. Therefore, if a user does realize that responses from a search service are often logically inconsistent, then the user's perceived quality of the search service can be affected in a negative way. Therefore, the logical consistency properties are a measure of users' perceived quality of search. [§]

[§] The first author wishes to thank Microsoft researchers for telling him that the following logical consistency property is also used within Microsoft to check the ranking robustness of Web search service: A robust search service should return similar results for similar queries. For instance, although a search for 'today's movies in Redmond' and a search for 'Redmond movies today' may return different results, the two result sets should share a large intersection if the search service is robust.

Another question is whether some query terms (such as 'GLIF' and '5Y4W' shown in Figure 3) really reflect users' real information needs. First, it must be noted that different users are interested in different kinds of information. For example, the search term '5Y4W' may not make sense to most users, but is of particular importance to some users who are looking for a product having the substring '5Y4W' in its model number or who are looking for a postal code containing '5Y4W' in certain countries. Indeed, it is hard to say what query terms reflect users' real information needs and what does not. Users should have the right to expect that the search engine will work reliably for all queries regardless of whether the query terms involve 'hot' words. Furthermore, research in software testing reveals that faults in programs are often exposed to unexpected inputs, and this is an advantage of random testing. Therefore, debuggers should check the system after serious violations of the logical consistency properties are detected.

To further investigate the relationship between users' information needs and logical consistency of search engines, the AOL search data (released in AOL Research Web site research.aol.com in 2006 but no longer online) were analysed. The search data contain Web search queries and user IDs. The data set contains a total of 39 389 567 queries collected from 630 554 users over a period of three months. It is found that a total of 266 142 (42.21%) of the users issued strongly logically connected (AND-related) queries, such as a query for A followed by another query for A AND B. In fact, 6 263 944 queries (that is, 15.90% of all queries) were involved in such a search pattern alone. Given that those 15.90% of queries were issued by 42.21% of users, the percentage of AND-related queries issued by these 42.21% of users was much higher, which is 44.56%. This finding indicates that logically consistent search results are highly desirable by many users.

Despite the relationship between logical consistency of search engines and users' information needs, it should be pointed out that the focus of the testing methods presented in this paper is not on users' information needs (which involve users' subjective judgment), but is on the *objective* assessment of search engine behaviour. It would be very interesting to know the relationship between search engine logical consistency and the conventional measures of precision and recall. However, because of the difficulties in measuring the latter on the live Web, such a study was not conducted.

2.2.   The experiments

Experiments have been conducted to test three Web search engines, namely Google, Yahoo!, and Live Search, against the logical consistency properties. The aim of the experiments was not to find the best search engine or the best properties for testing search engines. Instead, the aim was to see how some commonly used search engines compare in terms of consistency.

As mentioned in Section 2.1, the result counts returned by a search engine may vary between result pages. Normally, the count in the first result page is less accurate, and that in the last result page is the most accurate. Therefore, the original definitions of R_{AND}, R_{OR} and $R_{EXCLUDE}$ are slightly modified. R_{AND} is revised to $R_{AND}^{MultiPage}$: if $A_2 \equiv (A_1 \text{ AND } B)$, then $\min(Count(A_2)) \leq \max(Count(A_1))$, where $\min(Count(A_2))$ denotes the minimum of the count in the first result page and that in the last result page for query A_2, and $\max(Count(A_1))$ denotes the maximum of the count in the first result page and that in the last result page for query A_1. Similarly, R_{OR} and $R_{EXCLUDE}$ are revised to $R_{OR}^{MultiPage}$: if $A_2 \equiv (A_1 \text{ OR } B)$, then $\min(Count(A_1)) \leq \max(Count(A_2))$; and $R_{EXCLUDE}^{MultiPage}$: if $A_2 \equiv (A_1 \text{ AND } \overline{B})$, then $\min(Count(A_2)) \leq \max(Count(A_1))$.

Table I. Anomaly rates on result counts (unit: %).

	English dictionary			Random strings			Hot queries		
	AND	OR	EXCLUDE	AND	OR	EXCLUDE	AND	OR	EXCLUDE
Google	0.0	0.0	4.5	0.0	0.5	13.5	0.0	11.0	15.5
Live Search	1.5	15.0	37.5	0.5	5.0	33.0	0.0	5.0	14.0
Yahoo!	0.0	0.5	4.5	0.0	0.0	0.5	0.0	0.0	6.0

Three series of experiments were conducted to test search engines against the properties $R_{AND}^{MultiPage}$, $R_{OR}^{MultiPage}$, and $R_{EXCLUDE}^{MultiPage}$. In the first series, each search keyword was randomly selected from an English dictionary that contains 80 368 entries. In the second series, each search keyword was randomly generated strings containing combinations of any three characters from the set {A, B, ..., Z, a, b, ..., z, 0, 1, ..., 9}. In both series of experiments, words that are too common (such as 'is' and 'the') were excluded because they are often ignored by Web search engines. In the third series, each search keyword was randomly selected from a *hot query set* that contains 1000 latest hot queries obtained from Google Zeitgeist, Yahoo! Buzz, and MSN Insider (accessible via http://www.google.com/zeitgeist, http://buzz.yahoo.com, and http://www.imagine-msn.com/insider, respectively).

To obtain the most accurate results, quotation marks were used to enclose keywords in each search. This is because the quotation marks mean exact-word search in all the three engines. It should also be noted that, in order to obtain the most accurate results, it is important to disable all filters.

In each series of experiments, for each search engine under test, and for each of the three logical consistency properties, 200 tests were run, and each test involved two searches. The experimental results are summarized in Table I. First, consider the column of 'English Dictionary.' In this series of experiments, no AND-anomaly was detected for Google and Yahoo!, but Live Search had an AND-anomaly rate of 1.5%. Note that the anomaly shown in Figure 2 is not captured by these experiments because A_1 and B both contain only *one* English word, whereas in Figure 2, A_1 contains two English words, namely 'leakless' and 'dearer.' (Readers who are interested in AND-relations involving more than two keywords may refer to Zhou et al. [24] for experimental results.) It can be noted that different consistency properties produced very different anomaly rates, and the anomaly rate for $R_{EXCLUDE}^{MultiPage}$ was the highest for all search engines, namely 4.5% for Google and Yahoo!, and 37.5% for Live Search. All anomalies reported in this paper were repeatable at the time of experiment.

A look at the results on 'Random Strings' in Table I reveals a notable phenomenon: many of the anomaly rates have changed significantly as compared with the first series of experiments. Google's OR-anomaly rate increased from 0 to 0.5%, and EXCLUDE-anomaly rate increased from 4.5% to 13.5%. On the other hand, the anomaly rates of Live Search and Yahoo! have decreased. When considering the column 'Hot Queries' in Table I, Google produced its highest anomaly rates out of the three series. Live Search, on the contrary, reached its best performance. Yahoo!, however, yielded its highest EXCLUDE-anomaly rate. Overall, Yahoo! produced the most consistent search results.

Such observations may also help to improve users' perceived quality of search by providing hints on where flaws may or may not exist in the search system. Table I shows that, for each search engine tested, its anomaly rates for different logical consistency properties were significantly different. This suggests that the anomalies may be due to different causes/deficiencies in the system. Furthermore, consider each individual search engine. For Google, its anomaly rates increased significantly from 'English Dictionary' to 'Random Strings.' When compared with dictionary words, random strings can be assumed to occur less frequently as a search term, and they do not have much semantic meaning.

By looking at Google's anomaly rates in Table I, however, it is clear that such properties are not the cause for the anomalies since Google reached its highest anomaly rates on the 'Hot Queries.' This analysis also applies to the experimental results for Live Search and Yahoo!. In the absence of more information on the design of the search engines, it is not possible to find the causes for the anomalies. The above analysis, however, can provide a useful metric for search service providers to debug or tune their systems.

3.   TESTING SEARCH SERVICES IN TERMS OF THE QUALITY OF RANKING

The metric proposed earlier is search engine focused. A user focused metric is desirable since it can be assumed that the vast majority of general users are more concerned about what appears on the first few pages of the search results. In other words, users are not often concerned about the actual number of returned pages as long as the initial results (which are often limited to the top 10 results) displayed on screen match the user expectation. The first few results are therefore more critical to the overall perceived quality of ranking. Conventional methods for evaluating ranking quality can be both expensive and subjective because they entail a great amount of human judgment and manual labour. This section introduces an approach to assessing the ranking quality automatically using the concept of logical consistency properties among related responses.

Let <term> represent a search term submitted to a search engine, and let RS_0 represent the set of results returned. RS_0 can contain links to many kinds of files, such as HTML, PDF, TXT, and so on. Suppose that only the top 20 TXT files are of interest. Let RS_1 represent the subset of RS_0 containing the top 20 TXT files in their original order. Most search engines also allow users to search for a specific type of file directly, either through a command or through the Advanced Search option. Suppose a search engine is queried using the command '<term> filetype:txt', which denotes searching for files of type TXT containing <term>. Let RS_2 represent the set of the top 20 TXT files returned. (See Figure 4 for an illustration of RS_0, RS_1, and RS_2. In case of duplicate URLs, only the first one is taken and the duplicate ones are ignored.)

For an ideal search engine, RS_1 and RS_2 should be identical. That is, RS_1 and RS_2 should consist of the same set of files in the same order. This is because the same keyword <term> is used in both queries. If a search engine does not produce $RS_1 = RS_2$, then the smaller the difference between RS_1 and RS_2, the higher the ranking quality in terms of logical consistency, as it is natural for users to expect that good ranking algorithms should give logically consistent results.

The above discussion also applies to other file types. In total, four file types are selected to experiment with, namely TXT, PDF, HTML, and HTM files. These types of files are most common on the Internet, and all the three search engines support type-specific searches for these file types. Furthermore, they are also representative of files with different degrees of inter-document structures: TXT files are least structured; PDF files support structure through (hyper-)links but these are not often exploited; whereas HTML and HTM files are the most structured among the four.

Again, there are various other logical consistency properties that can be used to test the ranking quality of search services. Many of these properties can be identified by simply looking at the Advanced Search options of the search services. The aim of this research is not to find a comprehensive list of such properties or to compare their effectiveness. Instead, this research aims to propose and empirically
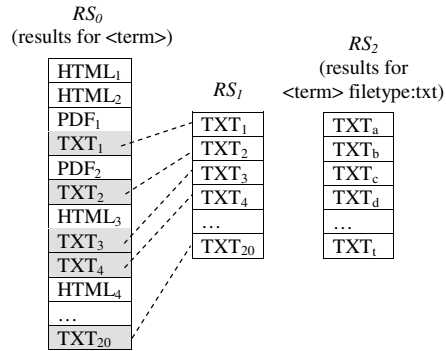
Figure 4. The ordered sets RS_0, RS_1, and RS_2.

study a new testing method. Under this methodology, many different logical consistency properties can be identified by practitioners for different kinds of search services.

3.1. Validity of the methodology

Is it reasonable to assume that a search engine uses a ranking scheme for the query '$<$term$>$' and a different ranking scheme for the query '$<$term$>$ filetype:txt'? From the user validation perspective, the 'filetype:txt' command is only a filter according to the specifications of the search engines and, hence, should not affect the ranking of pages. Regardless of how the search engines implement the command, this is a desirable property. In other words, if users find that this property does not hold, their perceived quality of ranking can be affected. Further discussions on the relationship between this property and the users' perceived quality of ranking will be given in Section 3.4.

A search engine can rank Web pages according to many factors, which can be basically classified as on-page and off-page factors. For example, according to keyphrase-specific findings reported by Fortune Interactive [25], the main factors used by Google, MSN (Live Search), and Yahoo! to rank Web pages are: (1) In-Bound Link Quality. This 'is a measurement of key elements on the page containing an in-bound link which, in combination, influence the link reputation for the target of the link,' which is 'the only factor that had the same level of relative influence across the search engines and happened to be the most influential in all cases.' (2) In-Bound Link Relevance. This is 'a measurement of the topic/keyphrase relevancy of the content on the page containing the in-bound link.' It can be further divided into 'In-Bound Link Title Keyword Density' and 'In-Bound Link Anchor Keyword Density.' (3) In-Bound Link Quantity. That is, the number of links pointing to the current page. This factor 'is of least relative importance among the off-page factors across the board.' (4) Title Keyword Density. This is an on-page factor.

Let F be a file of type $<$T$>$ on the Web. Suppose that F is ranked using scheme S_1 for query '$<$term$>$,' and ranked using scheme S_2 for query '$<$term$>$ filetype:$<$T$>$.' If S_1 and S_2 are different ranking schemes, then it must be because S_1 and S_2 adopt different off-page factors for F. To illustrate this, consider Figure 5(left), which shows a simplified topological structure of the files indexed in a search engine's database. For ease of discussion, it is assumed that all of the files are relevant to the user's query '$<$term$>$.' Among these 10 files, four are PDF files, four are HTM files, and two are
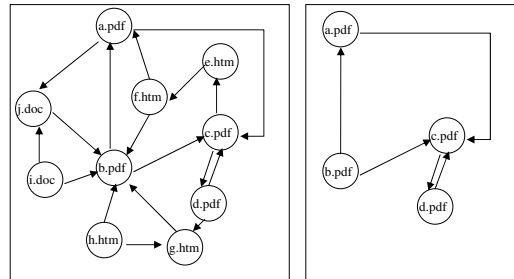
Figure 5. A simplified topological structure.

DOC files. The arrows in the figure represent hyperlinks between files. All other factors being equal, b.pdf will be ranked the highest among all the four PDF files because b.pdf has the highest in-degree (5 incoming arrows). Consider a second search with the query '<term> filetype:pdf.' Theoretically speaking, this time the search engine may rank the PDF files using either the *global view* (Figure 5(left)) or the *local view* (that is, PDF-only view as shown in Figure 5(right)) since the search is now confined to only PDF files. In the former case, b.pdf should still be ranked the highest; in the latter case, however, b.pdf should be ranked last because no PDF file points to b.pdf. Although the view actually adopted by the search engines is not known to external users, the following analysis ensures that, for all the three search engines under test, the global view is most likely to be adopted. First, to implement the local view is resource-consuming and inefficient. Second, the local view is not applicable to TXT files, because plain text (TXT) files do not have an inter-document structure and are normally not subject to hyperlinks analysis. Next, consider what happens if <T> is a non-TXT file type and involves hyperlinks structures, such as PDF, HTM, and HTML file types. If the local view (Figure 5(right)) is adopted to generate RS_2 in Figure 4 (in this case the 'TXT' files in the figure should be changed to 'PDF,' 'HTM,' or 'HTML' files), then the difference between RS_1 and RS_2 for PDF, HTM, or HTML files must be larger than that for TXT files because the former uses two different ranking schemes whereas the latter uses only one ranking scheme. However, as will be shown in Section 3.3, for all the three search engines tested, the difference between RS_1 and RS_2 (measured by common line rate (CLR), to be defined shortly) for PDF, HTM, and HTML files is much smaller than that for TXT files. As a result, it can be concluded that global views are used for the file types in all queries. Hence, the proposed logical consistency property is valid.

3.2.   Design of the experiments

A series of experiments have been conducted to test the three Web search engines against the consistency property by using the search APIs provided.

The experiments consist of two phases. In phase 1, the search engines are queried and their responses are recorded. In phase 2, the data collected in phase 1 are analysed. The following procedure describes

the basic flow of a test in phase 1. For ease of presentation, the following only describes a test for TXT files. Other file types are treated in the same way.

Step 1: Randomly select a word w from the English dictionary and query the search engine with w (without the 'filetype' command). The response of the search engine is RS_0.

Step 2: Collect the top 20 URLs that refer to TXT files from RS_0. That is, build RS_1 as illustrated in Figure 4. If the number of TXT files is smaller than 20, then collect all TXT files in RS_0. If this number is smaller than 10, then discard RS_0 and RS_1, and go to step 1.[¶]

Step 3: Query the search engine for 'w filetype:TXT' and collect the top 20 results to build RS_2. If the number of files is smaller than 20, then collect all files. If this number is smaller than 10, then discard RS_0, RS_1, RS_2, and go to step 1.

Step 4: If RS_1 and RS_2 have different sizes, then truncate the larger set by removing its tail elements, so that RS_1 and RS_2 can have the same size. (For instance, if RS_1 has 17 elements and RS_2 has 20 elements, then the last three elements in RS_2 will be removed.)

For each search engine and for each of the four file types (TXT, PDF, HTML, and HTM), 120 tests have been run. The dictionary that contains 80 368 English words was used to generate the search keywords. Again, common words such as 'of' were excluded.

The data collected in phase 1 are then analysed against two measures: the *CLR* and the *offset*, as defined below.

Common Line Rate (CLR): Let RS_1' and RS_2' be the subsets of RS_1 and RS_2, respectively, which consist only of the common URLs of RS_1 and RS_2 in their original orders. Therefore, RS_1' and RS_2' have the same elements. The orders of these elements in the two sets, however, may not be the same. The (CLR) is defined as $|RS_1'| \div |RS_1|$, which is the ratio of the number of common URLs of RS_1 and RS_2 to the number of URLs in RS_1. Note that $|RS_1| = |RS_2|$ and $|RS_1'| = |RS_2'|$.

Offset: In addition to the measuring of CLR, it is also important to know whether the elements in RS_1' and RS_2' have been ranked consistently, that is, whether they have the same order and, if not, how much they differ. The smaller the difference, the higher will be the ranking quality. Let $RS_1' = (e_1, e_2, \ldots, e_k)$, where the ranking of e_i in RS_1' is i, $i = 1, 2, \ldots, k$, and $k \leq 20$. Here, the cases where $k > 1$ are considered, because the intention is to measure how the elements of RS_1' and RS_2' are ranked differently. Let $position_{RS_2'}(e_i)$ denote the ranking of element e_i in RS_2'. Let $offset(e_i) = |i - position_{RS_2'}(e_i)|$. For instance, if $RS_1' = (a, b, c)$ and $RS_2' = (c, a, b)$, then $offset(a) = 1$, $offset(b) = 1$, and $offset(c) = 2$.

For an ideal search engine and for each e_i, $offset(e_i)$ should be 0, which means that all elements (URLs) are ranked consistently. For real-world search engines, the smaller the offset values, the higher the ranking quality in terms of logical consistency. Based on the notions introduced above, let the *average ranking offset* (ARO) be $ARO = \left(\sum_{i=1}^{k} offset(e_i)\right) \div k$. Let the *maximum ranking offset* (MRO) be $MRO = Max\{offset(e_i) \mid i \in [1, k]\}$.

Weighting: For the evaluation of experimental results, it is desirable to weight the elements when calculating the offset values. This is to accommodate for the properties of some common ranking algorithms, and for the common user perception that the topmost returned results are more significant

---

[¶]Most commercial search engines impose some limitations on the use of their search APIs. For instance, Google imposes a quota of 1000 queries per day. For each query, only the top 1000 results can be viewed. Therefore, it is sometimes not possible to collect 20 TXT files out of the top 1000 pages. But the procedure ensures that at least 10 TXT files are collected.
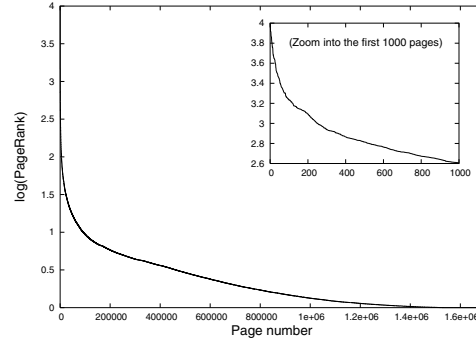

Figure 6. A logarithmic plot of PageRank values when ordered by values in descending order.

than the latter results. It should hence be appropriate to differentiate pages in different ranking positions when analysing the experimental results. In order to achieve this, a weight value is added to each element (URL) indicating its level of importance in response to a particular query. A non-linear weighting scheme is used based on a well-defined Web page ranking algorithm PageRank. PageRank [26] is arguably the most successful of Web page ranking algorithms. The distribution of rank values becomes evident when plotting the pages by PageRank value in descending order [27]. This is shown in Figure 6. The figure shows a *logarithmic* plot of PageRank values for a data set containing 1.67 million pages (taken from a well-known snapshot of a portion of the Web called WT10G). It is found that a reciprocal squared logarithmic function follows the curvature of the descending ordered PageRank values (the original values before applying logarithm) quite closely. In other words, the distribution of PageRank values falls sharply at a rate of approximately $f(x) = \frac{1}{\log^2(x)}$. This is the reason that it is a logarithmic rather than the original plot shown in Figure 6. It is thus observed that a change of order of highly ranked pages has a significantly higher impact on the overall perceived quality of ranking than does that of lowly ranked pages.

This observation provides the motivation to weight the results based on the distribution of PageRank values. A weighting function is utilized based on the reciprocal squared logarithmic function $f(x) = \frac{1}{\log^2(x)}$. Moreover, to avoid any dependence on the number of pages considered, the function is normalized by using the integral of $f(x)$, as follows:

$$w_i = \frac{\frac{1}{\log^2(i+2)}}{\left(li(k+2) - \frac{k+2}{\log(k+2)}\right) - \left(li(3) - \frac{3}{\log(3)}\right)} \tag{1}$$

where w_i is the weight that the i-th listed page contributes to the assessment procedure, i is the position (ranking) of the page in the result set, log refers to the natural logarithm, and k is the number of results collected for evaluation. For instance, to weight the second page out of the top 20 pages, $i = 2$ and $k = 20$. As explained previously, for the experiments, $1 < k = |RS_1'| = |RS_2'| \leq 20$. $li(x)$ denotes the Logarithmic Integral. Nielsen's approach is used to compute $li(x)$.

Equation (1) assumes an offset of 2 for i. This offset allows to more closely follow the curvature of the PageRank distribution, and helps to avoid problems when $i = 1$. The normalization step allows for a direct comparison of the various experimental results.
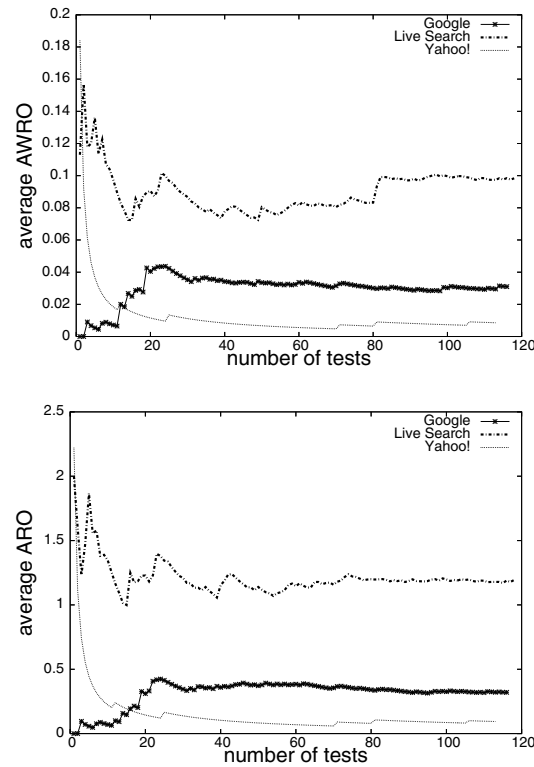
Figure 7. Average AWRO (above) and ARO (below) for files of type TXT.

Weights are taken into consideration and the *average weighted ranking offset* (AWRO) is defined by: AWRO$= \left(\sum_{i=1}^{k}\left(offset\left(e_i\right)\times w_i\right)\right)\div k$, where e_i, w_i, and k are as defined before. The *maximum weighted ranking offset* (MWRO) is defined by: MWRO $= Max\{offset\left(e_i\right)\times w_i \mid i \in [1,\,k]\}$.

3.3. Experimental results

A total of 120 tests have been run for each search engine and for each of the four file types. In each test, the values of CLR, ARO, MRO, AWRO, and MWRO were recorded. Each query involved only one word randomly selected from the English dictionary.

Figure 7(below) shows how the average AROs (for TXT files) vary when the number of tests increases from 1 to close to 120. (It is 'close to 120' because out of the 120 tests, those that resulted in $k \leq 1$ were skipped. This is explained previously in the paper. In the end, there were 116 valid tests for Google, 114 for Yahoo!, and 118 for Live Search.) It can be seen that when the number of tests increases, all the curves become more and more stable. Fluctuations at 60 are already very small. This illustrates that a sample size of 120 has yielded representative performance indicators for all three search engines. Similar patterns have been observed in all the other experiments.

Table II. Experimental results on common line rate (CLR).

	Average(%)				Minimum(%)				Maximum(%)				Standard deviation(%)			
	TXT	PDF	HTM	HTML	TXT	PDF	HTM	HTML	TXT	PDF	HTM	HTML	TXT	PDF	HTM	HTML
Google	41	64	69	78	5	15	18	20	100	100	100	100	25	18	19	17
Live Search	53	56	83	87	0	0	0	0	100	95	100	100	18	24	25	22
Yahoo!	75	74	97	99	0	0	45	60	100	100	100	100	26	27	6	4

Tables II and III summarize the results of the 120 tests based on the common line rate and the offset, respectively. Consider Table II first. A higher CLR value means a higher ranking quality in terms of logical consistency. For Google, its ranking quality improved significantly when the file type changed from TXT to PDF, HTM, and HTML. This can be seen not only from the average figures (increased from 41% to 78%) and minimum figures (increased from 5% to 20%), but also from the standard deviations (decreased from 25% to 17%, indicating more stable performance). For Yahoo!, its average and minimum CLRs and standard deviations for TXT and PDF files were at a similar level, but improved dramatically for HTM and HTML files. Live Search also had a similar pattern except for the standard deviations. Among the four file types, HTM and HTML files are the most structured, followed by PDF files, and TXT files are structureless. Therefore, the above experimental results suggest that all the three search engines are sensitive to file structures (including hyperlink structures) and vulnerable to the topology of Web components and off-page factors (as the ranking of TXT files relies more on off-page factors). For all the three search engines, the ranking quality for less structured files needs to be improved. Among the three search engines, Yahoo! outperformed the other two in terms of CLR.

Next, consider Table III, where ARO and MRO are unweighted offset measurements, and AWRO and MWRO are weighted offset measurements. As shown in Figure 7(above) and Figure 7(below), the curvatures of the average weighted and unweighted offset values were quite similar. The lower the offset value, the higher the ranking quality in terms of logical consistency. For Live Search, it is evident that all its average offset values and standard deviations improved when the file type changed from less structured to more structured types. This pattern is quite consistent with its pattern in CLR. Yahoo! exhibited a similar pattern. Note, however, that Yahoo! average ARO for HTML was 0.0145, greater than its PDF value 0.0057. But when one continues to look at Yahoo! AWRO, it is found that its HTML value was 0.0004, smaller than its PDF value 0.0005. This means that many of the inconsistent orders in Yahoo! search results for HTML files only happened to lowly ranked pages instead of highly ranked pages and, as a result, they did not have a significant impact on the weighted measurement AWRO.

It is surprising to find in Table III that Google yielded lower average offset values for TXT files than for the other more structured file types. In other words, Google exhibited two opposite patterns: Its CLR favored the more structured file types but its offset values favored the least structured file type. In the absence of more information on the design of the search engine, it is not possible to find the reason for this unusual phenomenon. It should be noted, however, that CLR and offset are two measurements having different granularity, and they indicate different aspects of ranking quality. This can indicate the need for Google to improve its CLR for less structured file types, and offset values for more structured file types. Among the three search engines, Yahoo! outperformed the other two in its offset values.

Table III. Experimental results on offset.

		Average				Minimum				Maximum				Standard Deviation			
		TXT	PDF	HTM	HTML	TXT	PDF	HTM	HTML	TXT	PDF	HTM	HTML	TXT	PDF	HTM	HTML
Google	ARO	0.3204	0.7470	0.6933	0.7931	0	0	0	0	2.5714	4.1426	4.2667	3.8750	0.4619	0.8459	0.6859	0.6892
	MRO	1.1810	3.1000	3.4250	4.1542	0	0	0	0	10	15	13	14	1.8444	3.3409	3.2139	3.3594
	AWRO	0.0310	0.0394	0.0328	0.0336	0	0	0	0	0.3123	0.2666	0.1656	0.2198	0.0546	0.0454	0.0346	0.0319
	MWRO	0.0913	0.1571	0.1517	0.1753	0	0	0	0	0.5241	1.0428	0.6359	0.8562	0.1278	0.1686	0.1440	0.1563
Live Search	ARO	1.1931	0.4205	0.1351	0.1385	0	0	0	0	5.0909	3.7333	2	3.4545	1.1723	0.5781	0.3332	0.4320
	MRO	3.3136	2.0487	0.7682	0.7722	0	0	0	0	11	15	12	11	2.8486	2.8663	1.8188	1.9482
	AWRO	0.0983	0.0309	0.0122	0.0086	0	0	0	0	0.9159	0.9159	0.9159	0.4640	0.1230	0.0783	0.0646	0.0416
	MWRO	0.2297	0.1111	0.0472	0.0385	0	0	0	0	1.0305	1.0305	1.1063	1.2422	0.1932	0.1478	0.1422	0.1374
Yahoo!	ARO	0.0931	0.0057	0.0044	0.0145	0	0	0	0	2.25	0.7059	0.6667	1.3684	0.4203	0.0547	0.0501	0.1062
	MRO	0.1930	0.0255	0.0375	0.1125	0	0	0	0	4	3	5	8	0.8295	0.2433	0.4125	0.7085
	AWRO	0.0086	0.0005	0.0001	0.0004	0	0	0	0	0.2322	0.0664	0.0235	0.0373	0.0393	0.0513	0.0016	0.0028
	MWRO	0.0231	0.0019	0.0011	0.0030	0	0	0	0	0.6116	0.1741	0.1771	0.2039	0.1036	0.0171	0.0130	0.0185

Table IV. Using Google to search for HTML pages written in different languages.

	Avg. CLR	Avg. ARO	Avg. AWRO
English	77.8%	0.7931	0.0336
Chinese	74.7%	0.7951	0.0387
French	76.0%	0.8622	0.0410

3.4. Relationship between the logical consistency properties and users' perceived quality of ranking

Consider Table II again. In the best case, all the search engines yielded a CLR of 100%. In the worst case, however, Google, Live Search, and Yahoo! had a CLR of 5%, 0%, and 0%, respectively. For instance, for the query 'tumefied,' Google produced its minimum (worst) CLR: The size of RS_1 and RS_2 (as shown in Figure 4) was 20, but they had only one element in common (namely the URL http://www.zen6741.zen.co.uk/quinapalus/misps/ds.txt). Therefore, the CLR of Google in this case was only $\frac{1}{20}$, or 5%. On the other hand, for the query 'acalephe,' Google achieved its maximum (best) CLR: Both RS_1 and RS_2 consisted of 20 elements, all of which were common elements.

This raises an important question: What is the relationship between the logical consistency properties, such as CLR, and users' perceived quality of ranking? Suppose that a user is looking for a specific type of file containing the keyword <term>. Suppose the search engine returned RS_2 as the top pages. Without knowing anything about RS_1, the user may or may not be satisfied with RS_2. If, however, the user knows that there is also another choice, namely RS_1, but RS_1 is very different from RS_2, then the user will either prefer RS_1 to RS_2, or prefer RS_2 to RS_1. In other words, when RS_1 and RS_2 are different, then from the user's perspective, one of them will have a higher ranking quality than the other. It is, therefore suggested that search engine designers should inspect the algorithm which generated the lower-quality result set, and improve it towards the higher-quality result set. The above discussion also applies to other logical consistency properties, such as the offset. Table III shows that Google, Live Search, and Yahoo! can generate (unweighted) offset values of considerable magnitude.

4. EXAMPLES OF FURTHER APPLICATIONS

The proposed method is simple in concept, easy to implement, and completely automatic. In addition to serving the purpose of conventional testing and debugging, it can be used for *various* other purposes. A few examples of such applications are given in the following, where several interesting issues are investigated.

4.1. A case study on language bias

This subsection attempts to check whether there is a bias in the ranking quality when searching for Web pages written in different languages. A case study is conducted with Google. In the experiment, HTML pages written in the English, Chinese, and French languages were searched. For each language, 200 tests were run. Google search API was used, which allows to set language preference by invoking the function *setLanguageRestricts*(). The results are summarized in Table IV, which shows that Google

had the best performance on all the three measurements for English Web pages. The differences among languages, however, were quite small. It can be concluded, therefore, that no language bias was found in Google.

4.2.   The impact of commercial interests on ranking

This subsection attempts to find whether commercial interests could have an impact on the ranking quality. This is because many search keywords (such as 'travel,' 'hotel,' and 'laptop') have commercial/advertising values. Will the search engines change the order of pages when presenting them to users because of commercial reasons? As a pilot study, 200 tests were conducted on Google. Each test used commercial query terms instead of dictionary words. As Google returns both *normal* search results and *sponsored links*, the following items were collected in each test: (1) the number of sponsored links (complete sponsored links were collected via the URL http://www.google.com/sponsoredlinks), (2) the number of commercial sites‖ that appeared in the top 20 search results, and (3) the CLR and offset values of the search results when considering HTML pages only.

The correlations between the number of sponsored links/commercial sites and the CLR/offset values are analysed. Because of space limitations, the details of the analyses will not be presented in this paper. The result indicates that there was no correlation. This means that the ranking quality/consistency of Google was not affected by commercial interests.

4.3.   Change of ranking quality with time

According to the experimental results reported in the previous sections, all of Google, Live Search, and Yahoo! yielded inconsistencies in their search results. The further study focused on whether and how the ranking quality changes. If changes are detected, can a model be built to describe them? To what degree is the quality of Web search engines affected by the dynamics of the Web?

It is postulated that the quality of the search results may change with time, because all Web search systems need to update their databases and caches based on certain rules. When updating is in progress, the consistency of the search results may be affected in a negative way. Furthermore, there are many other time-related factors that may affect ranking consistency, such as network congestion, sudden emerging searches, and so on. At different times, a client's requests might also be directed to different servers according to certain server selection rules.

A time-based experiment is then designed as follows: Over a duration of four weeks, for each of the three Web search engines, 100 tests were run every hour and its hourly mean CLR and ARO values for HTML pages were recorded. In order to make a meaningful comparison, the same set of queries was reused every hour. To generate this query set, the English dictionary used in the previous experiments was still used. The experiment was designed to show whether and how the ranking consistency of Web search engines changes hourly, daily, and weekly. A large number of observations and analytical results were made. In the following, an extract of these is presented.

‖ A Web site is considered to be a commercial site when it is from the '.com' or '.com.xx' (such as .com.au) domain. This is, however, only a rough classification in order to automate the testing process.
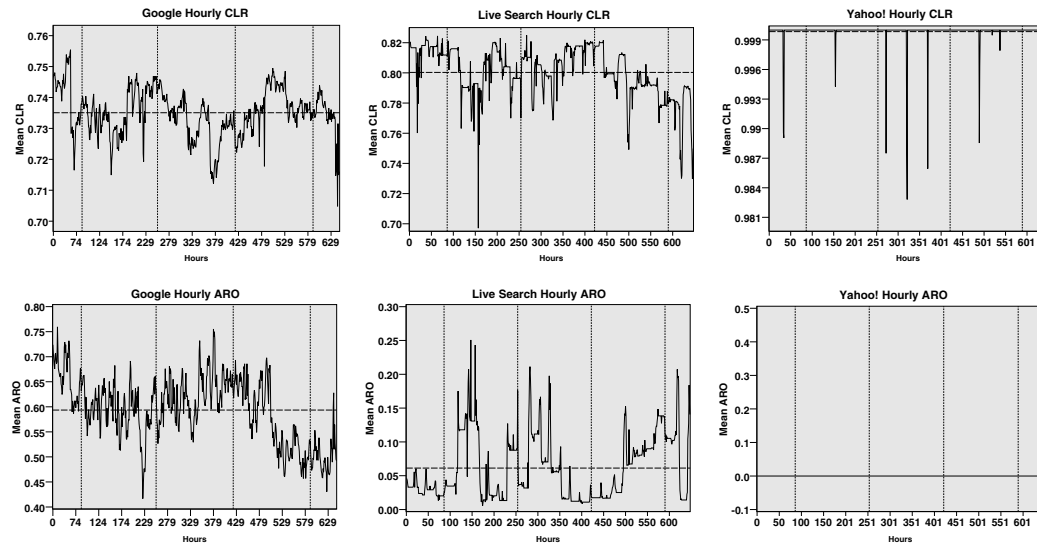
Figure 8. Hourly observations of mean CLR and ARO values for Google (left), Live Search (middle), and Yahoo! (right).

Overall results: Figure 8 shows hourly changes of the mean CLR and mean ARO values of Google, Live Search, and Yahoo!. The *x*-axis indicates the hours when the results were collected. The *y*-axis indicates the hourly mean CLR or ARO values (of the 100 tests per hour). The experiment started at 10:00 a.m., 12 June 2008, Sydney time, and finished at the 647th hour at 9:00 a.m., 9 July 2008. Because of network connection issues, the results of a few hours are missing. For Google, one day's result (14 June 2008) is missing. Where comparisons are made, these missing data are avoided. Furthermore, to avoid influences of causes that could result in an incorrect measurement, when calculating each hourly result, the two maximum and the two minimum values are eliminated (hence, $100 - 4 = 96$ tests per hour contributed to the calculation). The horizontal reference line represents the overall mean value of all the hourly mean values. The vertical reference lines represent the first hour on Monday every week.

It can be observed that, even for the same set of queries, both CLR and ARO values vary over time. For example, Google produced hourly mean CLRs fluctuated within the range from 70% to 76% during the 647 hours and exhibited an overall mean CLR of 73.5%. For ARO, the overall mean value of Google is about 0.59 and the change was within 0.35. In comparison, Live Search mostly produced a comparatively higher mean CLR than Google with an average of 80% where most observations were within the range of 74%–83%. There was one significant drop on the curve, which corresponds to 18 June 2008 (a drop to 69%). This could be led by an unexpected surge in visitors to the search engine system, which is typically due to some newsworthy event that had just taken place. Such events are called *flash crowds*. The curve of Live Search hourly mean ARO presents roughly an opposite trend as the one of hourly mean CLR. The higher the CLR value or the lower the ARO value, the better will be the ranking consistency. Thus, in this case, either CLR or ARO values can approximately indicate the general ranking consistency of Live Search.
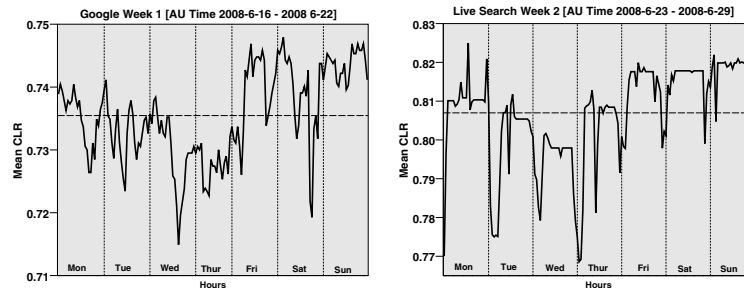
Figure 9. A typical weekly observation for Google (left) and Live Search (right).
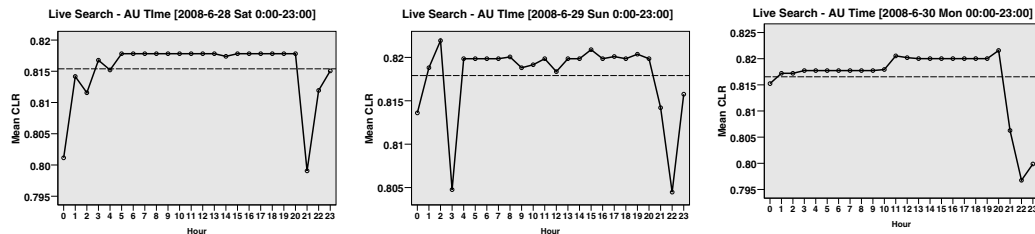

Figure 10. A typical daily observation for Live Search.

Yahoo! yielded the best results with the highest overall mean CLR close to 100%. There were less than 10 (minor) deviations during the observation period of 647 hours. Yahoo! also produced a perfect offset result, which was consistently 0 for the HTML pages. Note that the three search engines were tested at the same time.

The comparative performance of the three search engines shown in Figure 8 agrees with that indicated in Tables II and III despite the fact that the two sets of experiments were conducted using different query sets, on different dates, and that the results were calculated using slightly different methods (two maximum and two minimum values were excluded when calculating each mean value for Figure 8).

Weekly results: For weekly results, only the results of Google and Live Search are displayed since Yahoo! produced near perfect results during the observation period and, hence, does not require further analytical discussion. Figure 9(left) shows a typical week's results of Google. Similar to the previous plots, the *y*-axis represents the hourly mean CLR and vertical dashed lines separate days in the week. The curve of this typical week reveals few similarities between daily patterns, and it is difficult to find any regularity in the patterns. The erratic and frequent changes of the mean CLRs indicate that the Google search system is a dynamic system.

Figure 9(right) shows that Live Search yielded comparatively more stable CLR results than Google as the observations produced a somewhat smoother curve. The hourly mean CLRs of the week show decreasing trend from Monday to Wednesday. More general is the observation that the lowest values are often observed on Wednesday midnight into the early morning hours on Thursday. The results

on Friday, Saturday, and Sunday were roughly consistent and changed within a relatively small range during the daytime hours. Overall, the observations (see also the daily results shown in Figure 10) reveal a pattern that the CLR of Live Search is generally the best and also stable during the daytime hours whereas this can reduce quite significantly at a time near midnight. This is presumably a result of a scheduled event at Live Search, such as a synchronization of caches.

Overall, it appears that CLR results observed for Live Search follow an apparent daily and weekly pattern that can be used to describe and predict the changes of ranking consistency over time. In contrast, Google produced patterns that are much more dynamic and change very frequently. Among the three search engines, Yahoo! yielded almost a continued perfect ranking consistency during the observation period. This demonstrates a remarkable advantage of Yahoo! in terms of user-centric reliability of ranking. The observed difference among the results of the three search engines can be explained on their different focuses. Google pays great attention to enlarging the Web coverage and, for this purpose, continuously runs crawlers to keep its database up to date. In addition, owing to the popularity of Google, it has to dynamically adjust its ranking algorithm to counter problems associated with spam. Some of these measures involve human intervention. In contrast, it can be stated that Yahoo! focuses on the relevancy and accuracy of results in a largely automated fashion involving exact methods rather than using approximate methods. Live Search is placed somewhat in between Google and Yahoo! in terms of consistency and, hence, exhibits methods that balance the two prior directions. However, the experimental results do not provide absolute evidence that fully supports the reasons behind the changes of ranking consistency with time for each search engine. From another point of view, even though the precise cause of the observed phenomenon may not be identified, the results do provide hints to search engine developers, who have actual knowledge of the underlying techniques. This provides a means for the developers to verify whether the search engine behaviour complies with its design goals.

4.4. Users' perceived quality of Web pages

Recent work by Kc et al. [28] proposed a ranking scheme on the basis of users' perceived quality of documents on the Web. The atomization of the quality assessment procedure [28] was made possible through an algorithmic interpretation of documented cognitive processes that lead to the assessment of perceived quality of a Web document. An interesting observation by Kc et al. is that there is no correlation between the popularity of a Web document and the quality of the same document. In other words, Web documents that are commonly hyperlinked by other pages are as likely to be of high quality as documents that are rarely or not hyperlinked by other pages.

It is interesting to know whether there is a relation between the logical consistency of a search engine and the quality of pages that it returns to users. A set of experiments have, therefore, been carried out using the procedure as described in Section 3.2 for HTML pages. For each of the three search engines, namely Google, Live Search, and Yahoo!, 1000 tests were run. Query keywords were generated from the English dictionary. In each test, in addition to calculating the CLR and offset values, all the HTML pages contained in RS_1 and RS_2 were downloaded, and the *quality score* of each of these HTML pages was calculated using the method described in Kc et al. [28]. A higher quality score indicates higher quality of the Web page. The calculation was based on the following quality indicators [28]: spelling accuracy, grammatical correctness, explicit indication of authorship, existence of references, non-spam probability, correctness of content, timeliness, the probability of the Web page's content being biased, and whether the document is too short to contain sufficient information.

Table V. Page quality and ranking consistency.

	Mean QS_1	Mean QS_2	Mean CLR	Mean ARO	Mean AWRO
Google	0.22068814	0.22180014	75.0%	0.5703	0.0275
Live Search	0.22860018	0.22883642	80.0%	0.1577	0.0100
Yahoo!	0.23365403	0.23367587	99.8%	0.0000	0.0000
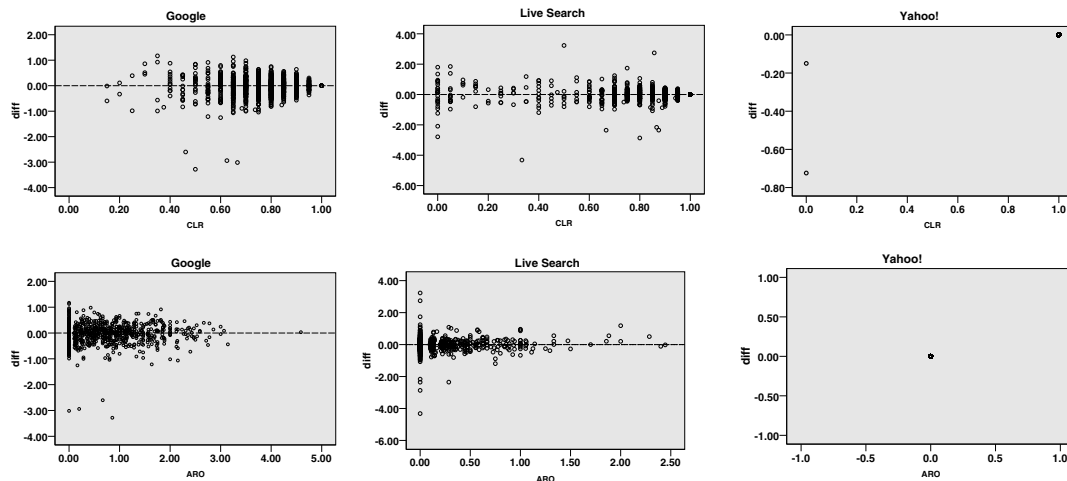
Figure 11. Distribution of 1000 *diff* values plotted against ranking consistency measurements for Google (left), Live Search (middle), and Yahoo! (right).

For test i ($i = 1, 2, \ldots, 1000$), the total quality score and mean quality score of all the HTML pages contained in RS_1 were calculated, denoted by $total(QS_1^i)$ and $mean(QS_1^i)$, respectively. The total and mean quality scores for RS_2 were also calculated, denoted by $total(QS_2^i)$ and $mean(QS_2^i)$, respectively.

The results of the 1000 tests are summarized in Table V, where Mean $QS_1 = \left(\sum_{i=1}^{1000} mean(QS_1^i) \right) \div$ 1000, and Mean $QS_2 = \left(\sum_{i=1}^{1000} mean(QS_2^i) \right) \div 1000$. The comparative performance of the three search engines in terms of the mean CLR, ARO, and AWRO for HTML pages agrees with the results of the previous experiments, namely Yahoo! best, Live Search second, and Google third. It is interesting to observe that the comparative performance of the three search engines in terms of *page* quality also followed the same order: Yahoo! yielded the highest (best) mean quality scores (both mean QS_1 and mean QS_2), followed by Live Search, and Google returned pages with quality scores comparatively lower than the other two. This observation gives search engine developers a hint for quality improvement by suggesting that there can be a relationship between the logical consistency of a search engine and the quality of pages it returns. Further study on this relationship is an interesting topic for future research.

So far the discussion has been focused on the quality of Web pages. The ultimate purpose of search engines is to satisfy users' information needs. As most Web searchers prefer high quality pages, there should intuitively be a strong correlation between the quality of Web pages and the quality of meeting users' information needs. This will be an interesting topic for future research.

It is also observed that, for each of the three search engines, mean QS_2 is slightly higher than mean QS_1. A further question is whether different treatments of the page quality indicators in ranking algorithms could be a cause for the observed ranking inconsistencies. To answer this question, the following analysis has been conducted. For each test, the difference between the total quality scores of RS_1 and RS_2 is calculated. That is, for test i ($i = 1, 2, \ldots, 1000$), its difference value is defined to be $diff = total(QS_1^i) - total(QS_2^i)$. Note that $total(QS_1^i)$ and $total(QS_1^i)$ can be compared directly because RS_1 and RS_2 have the same size. When $diff > 0$, RS_1 has a better overall page quality; when $diff < 0$, RS_2 has a better overall page quality; when $diff = 0$, RS_1 and RS_2 have the same overall page quality. These $diff$ values are first plotted against CLR, and then against ARO, as shown in Figure 11. Figure 11(upper left) shows that, for Google, when CLR=1 (which means that RS_1 and RS_2 contain the same set of pages), $diff$ =0. This is obvious by definition. It is further observed that when CLR<1, about a half of the points are distributed above the horizontal dashed line, and the other half of the points are below the line. The distribution is quite even except that there are a few far apart points below -2. This means that in about 50% of the situations where a ranking inconsistency in terms of CLR occurred, the algorithm that generated RS_1 outperformed the algorithm that generated RS_2 in terms of page quality, and vice versa. This indicates that there was not much difference in the way that the page quality indicators were treated by these two algorithms. A similar observation is also made for Figure 11(lower left). Hence, it can be concluded that the treatment of the page quality indicators in the ranking algorithms was not a cause for the ranking inconsistencies measured by CLR and ARO. Similar observations are also made for Live Search. Figure 11(upper right) shows Yahoo! results, where only three points can be seen. This is because 998 points overlap at CLR=1 and $diff$ =0. For the other two points, although both of their $diff$ values are smaller than 0, one cannot conclude that RS_2 gives better quality pages than RS_1 because the sample size is trivial. Both these two points have a CLR value of 0 and, hence, are excluded from calculation for ARO. This explains why in Figure 11(lower right) all the points overlap at ARO=0 and $diff$ =0.

Further analysis has also been conducted by applying the weighting scheme introduced in Section 3.2, and similar observations have been made. Therefore, it can be concluded that page quality was not a cause for the ranking inconsistencies.

5. DISCUSSIONS

The search engines considered in this paper are very large-scale distributed systems that operate within a very dynamic environment. It is well known that it can be difficult to maintain large-scale distributed systems in a consistent state at all times [29, 30]. Designers may allow a certain level of temporary inconsistency in the system in order to achieve a better system performance during periods of peak system load. Such inconsistencies are transient in nature as the system will strive to return to a consistent state when the system load is eased. This paper has provided evidence that many of the observed inconsistencies cannot be explained by such system variables, hence indicating possibly faulty software features.

A main problem with inconsistencies is how to measure it. In distributed system design this is always application specific, and is often based on some form of numerical deviation or order deviation [29,30]. However, these measures do not express the impact on user perception about the systems' behaviour.

This paper introduces a user centric inconsistency measure. The proposed measure is computed based on what the users get to 'see' when interacting with a search engine.

6. CONCLUSION AND FUTURE WORK

Online search services are the main interface between users and the Internet. They are of major economic and political significance. A fundamental challenge in the quality assurance of these important services is the oracle problem. Not only does the sheer volume of data on the Internet prohibit testers from verifying the search results, but it is also difficult to objectively compare the ranking quality of different search algorithms on the live Web, owing to the fact that different assessors often have very different opinions on the relevance of a Web page to a query. If the quality cannot be *measured*, it cannot be *controlled* or *improved*.

The first contribution of this paper is the proposal of a novel method, which employs logical consistency properties to alleviate the oracle and automation problems in testing online search services. The presented method automatically tests search services in terms of (a) the quality of returned counts and contents, and (b) the ranking consistency. Hence, this method can perform user-centric as well as provider-centric tests of search services. The logical consistency properties are necessary, but not sufficient, conditions for good search services, and this is a limitation of all testing methods.

The second contribution of this paper is that significant experimental findings have been obtained. It has been shown that failures in search engines can be detected effectively and efficiently. The experiments reveal that three commonly used Web search engines, namely Google, Yahoo!, and Live Search, are not as reliable as many users would expect. It is found that Yahoo! generally outperformed the other two in both its count quality and ranking quality. It is also observed that the accuracy of any of these search engines varies over time. These variations are predictable in the case of Live Search and Yahoo! but are rather unpredictable in the case of Google. It is furthermore observed that there is a dependency between file types and the accuracy of a search engine. Relationships with users' perceived quality of ranking and with users' perceived quality of Web pages have also been discussed.

As there are many factors affecting the performance of search services, it is not the aim of this research to give a comprehensive comparison of their qualities. The method presented in this paper is simple in concept and easy to implement. It can be used by both search service developers for quality improvement and algorithm comparison, and by assessors (including users) for quality assessment.

A next step of this research is to consider client side measures that can be taken to compensate some of the (now known) inaccuracies of a search service. This may be possible through a machine learning approach. In particular, recent research extended the ability of machine learning to allow the encoding of structured, Web-based information in either an unsupervised fashion [31] or a supervised fashion [32].

REFERENCES

1. Patterson AL. Multiple index based information retrieval system. United States Patent Application 20060106792, 2006. Available at http://www.freepatentsonline.com/y2006/0106792.html (last access: 22 May 2010).
2. Kc, M.W.-T. *Building a Prototype for Quality Information Retrieval from the World Wide Web*, PhD Thesis, University of Wollongong: Wollongong, Australia, 2009.
3. Landoni M, Bell S. Information retrieval techniques for evaluating search engines: A critical overview. *Aslib Proceedings: New Information Perspectives* 2000; **52**(3):124–129.
4. Zhao H, Meng W, Yu C. Automatic extraction of dynamic record sections from search engine result pages. *Proceedings of the 32nd International Conference on Very Large Data Bases* (*VLDB 2006*), VLDB Endowment, 2006; 989–1000.
5. Chu H, Rosenthal M. Search engines for the World Wide Web: A comparative study and evaluation methodology. *Proceedings of the ASIS 1996 Annual Conference*, 1996. Available at http://www.asis.org/annual-96/ElectronicProceedings/chu.html (last access: 22 May 2010).
6. Su LT. A comprehensive and systematic model of user evaluation of Web search engines: I. Theory and background. *Journal of the American Society for Information Science and Technology* 2003; **54**(13):1175–1192.
7. Clarke SJ, Willett P. Estimating the recall performance of Web search engines. *Aslib Proceedings: New Information Perspectives* 1997; **49**(7):184–189.
8. Weyuker EJ. On testing non-testable programs. *The Computer Journal* 1982; **25**(4):465–470.
9. Buckley C, Dimmick D, Soboroff I, Voorhees E. Bias and the limits of pooling. *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR 2006*), ACM Press: New York, NY, 2006; 619–620.
10. Carterette B, Allan J, Sitaraman R. Minimal test collections for retrieval evaluation. *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR 2006*), ACM Press: New York, NY, 2006; 268–275.
11. Järvelin K, Kekäläinen J. IR evaluation methods for retrieving highly relevant documents. *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR 2000*), ACM Press: New York, NY, 2000; 41–48.
12. Järvelin K, Kekäläinen J. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems* 2002; **20**(4):422–446.
13. Soboroff I. Dynamic test collections: Measuring search effectiveness on the live Web. *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR 2006*), ACM Press: New York, NY, 2006; 276–283.
14. Soboroff I, Nicholas C, Cahan P. Ranking retrieval systems without relevance judgments. *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR 2001*), ACM Press: New York, NY, 2001; 66–73.
15. Chen TY, Tse TH, Zhou ZQ. Semi-proving: An integrated method for program proving, testing, and debugging. *IEEE Transactions on Software Engineering* 2010; **37**(1):109–125.
16. Kilgarriff A. Googleology is bad science. *Computational Linguistics* 2007; **33**(1):147–151.
17. Grefenstette G. The WWW as a resource for example-based MT tasks. *Proceedings of the 21st International Conference on Translating and the Computer*, Aslib Publications: London, UK, 1999.
18. Keller F, Lapata M. Using the Web to obtain frequencies for unseen bigrams. *Computational Linguistics* 2003; **29**(3):459–484.
19. Lapata M, Keller F. Web-based models for natural language processing. *ACM Transactions on Speech and Language Processing* 2005; **2**(1):Article No. 3.
20. Nakov P, Hearst M. Search engine statistics beyond the n-gram: Application to noun compound bracketing. *Proceedings of the 9th Conference on Computational Natural Language Learning* (*CONLL 2005*), Association for Computational Linguistics: Morristown, NJ, 2005; 17–24.
21. Turney PD. Mining the Web for synonyms: PMI-IR versus LSA on TOEFL. *Proceedings of the 12th European Conference on Machine Learning* (*EMCL 2001*), Springer: London, UK, 2001; 491–502.
22. Pezzè M, Young M. *Software Testing and Analysis: Process, Principles, and Techniques*. Wiley: New York, NY, 2008.
23. Ammann P, Offutt J. *Introduction to Software Testing*. Cambridge University Press: New York, NY, 2008.
24. Zhou ZQ, Tse TH, Kuo F-C, Chen TY. Automated functional testing of Web search engines in the absence of an oracle. Technical Report TR-2007-06, Department of Computer Science, The University of Hong Kong: Pokfulam, Hong Kong, 2007.
25. http://www.fortuneinteractive.com/laptop.php (last access: 18 April 2010).
26. Page L, Brin S, Motwani R, Winograd T. The PageRank citation ranking: Bringing order to the Web. Stanford InfoLab Publication 1999-66, Stanford University: Palo Alto, CA, 1999. Available at http://dbpubs.stanford.edu/pub/1999-66 (last access: 22 May 2010).

27. Tsoi AC, Morini G, Scarselli F, Hagenbuchner M, Maggini M. Adaptive ranking of Web pages. *Proceedings of the 12th international conference on World Wide Web* (*WWW 2003*), ACM Press: New York, NY, 2003; 356–365.
28. Kc M, Hagenbuchner M, Tsoi AC. Quality information retrieval for the World Wide Web. *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology* (*WI-IAT 2008*), IEEE Computer Society Press: Los Alamitos, CA, 2008; 655–661.
29. Yu H, Vahdat A. The costs and limits of availability for replicated services. *Proceedings of the 18th ACM Symposium on Operating Systems Principles* (*SOSP 2001*), ACM Press: New York, NY, 2001; 29–42.
30. Yu H, Vahdat A. Minimal replication cost for availability. *Proceedings of the 21st Annual Symposium on Principles of Distributed Computing* (*PODC 2002*), ACM Press: New York, NY, 2002; 98–107.
31. Tsoi AC, Hagenbuchner M, Chau R, Lee V. Unsupervised and supervised learning of graph domains. *Innovations in Neural Information Paradigms and Applications* (*Studies in Computational Intelligence* 247), Springer: Berlin, Germany, 2009; 43–65.
32. Yong SL, Hagenbuchner M, Tsoi AC. Ranking Web pages using machine learning approaches. *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology* (*WI-IAT 2008*), IEEE Computer Society Press: Los Alamitos, CA, 2008; 677–680.