folksonomy can be acquired. (Trant et al. [Trant 06], as well as Kellogg [Kellogg 06], have addressed the use of social tagging for describing museum collections.)

## 3. Project Design and Evaluation Plan

*SoLibS* is designed for (i) automatic query expansion which facilitates the retrieval of relevant library records, (ii) performing exact and partial word matching, and (iii) using representative keywords (i.e., LibraryThing tags) to describe a (library) book, which reduces, if not completely eliminates, the relatively high percentage of failed library catalog searches. The merit of *SoLibS* will be verified by showing that it can significantly improve the *quality* and *quantity* of the search results retrieved by existing library search engines with an optimal query processing time.

### 3.1 Project Design

In the following subsections, we detail the query processing steps, the overall system design of *SoLibS,* and various well-established, technically sound approaches adopted by *SoLibS* for enhancing its efficiency and effectiveness on query evaluation and retrieving relevant results.

**3.1.1** ***Word Similarity.*** During the process of evaluating a library patron's query *Q*, *SoLibS* computes the degree of resemblance of *Q* and the representation of a record *R* in the corresponding library catalog, which is determined by *partial similarity* and *exact matching* among the keywords in *Q* and *R*. In order to determine the similarity among different keywords, we adapt the *word-correlation factors* in the word-correlation matrix *M* defined in [Koberstein 06]. The word-correlation factors in *M,* which were computed by using a set of approximately 880,000 Wikipedia documents (downloaded from http://www.wikipedia.org/), reflect the similarity value of any two words based on their (i) *frequency of co-occurrence* and (ii) *relative distances* in each Wikipedia document that establishes how closely related the two words[3] (in *Q* and *R*) are.

Wikipedia documents were chosen to construct *M,* since they (i) were written by more than 89,000 authors with different writing styles, (ii) cover a wide range of topics, and (iii) are unbiased in word usage and content. Furthermore, the words in *M* are commonly-used words that appear in various online English dictionaries, such as 12dicts-4.0 (http://prdownloads.sourceforge.net/wordlist/12dicts-4.0.zip), Ispell (http://ficus-www.cs.ucla.edu/geoff/ispell.html), and BigDict (http://download393.mediafire.com/tftfwbgmj2jg/2x64euxe4pt/bigdict.zip). Compared with synonyms and other related words (such as hypernyms or hyponyms) compiled by WordNet (http://wordnet.princeton.edu/), in which none of the word pairs is assigned any *similarity weight*, our word-correlation factors provide a more accurate measure of word similarity that were computed by the appearance of words in a huge set of documents.

Due to the size of the word-correlation matrix *M,* which sums up to 6.0 GB, we will examine and use alternative, reduced versions of *M* with the intention of finding the most suitable *minimized word-correlation matrices* to reduce the query processing time without compromising the accuracy of retrieving relevant results.

**3.1.2** ***Using LibraryThing Tags to Represent Library Catalog Records.*** During the pre-processing step and query evaluation process, we use an existing folksonomy, i.e., *collaborative tagging* that describes the content of each book, at *LibraryThing*[4] (http://www.librarything.com), which was founded in 2006 for aiding users in cataloging and referencing books. As of January 31, 2009, LibraryThing archives 35,794,718 unique records (on books), and approximately 610,010 users have added approximately 46.5 million tags to different book records at LibraryThing, according to the statistical data released by the Zeitgeist Overview (http://www.librarything.com/zeitgeist).

We intent to represent each record in a library catalog using the *tags* for the corresponding book created by ordinary users of LibraryThing, which are not part of a controlled vocabulary. These LibraryThing tags are words ordinary users are familiar with, as opposed to the keywords in LCSH. We will exclude (i) tags that are

[3] Words within the Wikipedia documents were *stemmed* (i.e., reduced to their grammatical root forms) and *stopwords* (i.e., words that carry little meaning such as articles, prepositions, conjunctions, etc.) were removed, which (as a side effect) minimize the number of words that should be considered.

[4] The use of LibraryThing tags in academic libraries is discussed in [Ismail 08], and the benefits of using collaborative tagging to enhance the retrieval quality of an IR system are given in [Clements 08, Crecelius 08].

personalized and used as a reminder, such as "read," "want to read," and "borrowed," [5] as well as (ii) tags that are stopwords, which provide little meaning in distinguishing the contents of library catalog records.

Due to the huge number of tags available at LibraryThing, we minimize the number of tags to be compared during the query evaluation process by choosing only the top-$n$ ($n \geq 1$) tags describing a particular book $B$, where the top $n$ tags are determined by the *frequency counts* of tags associated with $B$ as specified in LibraryThing. The proper number of $n$ is chosen according to (i) its impact on the *accuracy* in retrieving relevant library records and (ii) the *processing time* in establishing the *degree of resemblance* between a query and the corresponding tags associated with a library catalog record.

**3.1.3** ***Installing a MySQL Database for Query Processing***. In an attempt to further reduce the query processing time in computing the degrees of resemblance between a patron's query and the representations of library catalog records, we have considered using sophisticated data/file structures in MySQL for different purposes. These structures store (i) general information of library catalog records, (ii) LibraryThing's tags describing the content of library catalog records, and (iii) the reduced word-correlation matrices, besides using the InnoDB storage engine of a MySQL 6.0 database system, which is designed for maximizing the performance in processing large volumes of data and has a CPU efficiency that is not matched by other disk-based relational database engines. (See http://dev.mysql.com/doc/refman/6.0/en/innodb-overview.html.)

***3.1.4 Selecting Subset of Relevant Records for Reducing the Query Processing Time.*** Prior to computing the similarity, i.e., the degree of resemblance, between a library patron's query $Q$ and the $n$ LibraryThing tags that represent the content of each record in a library catalog, we first discard records that are highly likely *irrelevant* or *fairly dissimilar* to $Q$ by selecting and retaining only the ones with tags (i) matching *at least one* query keyword in $Q$ exactly, or (ii) *highly* correlated (based on their word-correlation factors) to at least one of the keywords in $Q$, which yields a *subset* of library catalog records that are likely relevant to $Q$. This filtering technique, which is a crucial step of *SoLibS*, potentially reduces the very large number of comparisons on library records, and is generally known as *blocking* [Kelley 84], which can significantly minimize the computational cost and processing time [Yan 07] while avoids excluding any highly relevant library records with respect to $Q$.

**3.1.5** ***Relevance Ranking of the Retrieved Library Books***. The *degree of resemblance* between a query $Q$ and each of the *selected* library catalog records $R$ (as chosen in Section 3.1.4) can be determined by *adding* the correlation factors (in the word-correlation matrix) between each of the words in $Q$ and each of the tags associated with $R$. We limit to 1.0, the value of an *exact* match, as the highest possible cumulative word-correlation value between a tag in $R$ and the keywords in $Q$. Subsequently, if tags of $R$ are *highly similar* to *most* (if not all) of the keywords in $Q$, then $R$ should be ranked *higher* than another record in which only one of its tags is *highly similar* to only a few of the keywords (or matches exactly with only one keyword) in $Q$.

***3.1.6 Use of Boolean operators***. *SoLibS* is designed so that users can specify well-defined Boolean operators, i.e., AND, OR, or NOT, when formulating a query. In implementing this advanced search capability, we intent to (i) develop a friendly user interface to facilitate the process of constructing Boolean queries, and (ii) adopt the Fuzzy Set IR Model [Ogawa 91] to perform partial-similarity matches and apply the Fuzzy Boolean operator evaluation strategy on the conjunctive components of a Boolean query. In constructing Boolean queries, users are only required to combine keywords (that are entered first) and Boolean operators (to be chosen from a drop-down menu) one at a time. In other words, library patrons will be asked to specify in between a pair of search keywords a binary Boolean operator (AND or OR) to combine the two search terms, or the (similar) keyword that should not appear in the retrieved results using the Boolean operator NOT. The end result is a simple Boolean subexpression, i.e., ($A$ AND $B$), ($A$ OR $B$), or (NOT $A$), where $A$ and $B$ are search keywords, and the resultant subexpression can be combined with another search keyword or subexpression using a binary Boolean operator or associated with the Boolean operator NOT. This process is invoked recursively until a well-formed Boolean expression (i.e., query) is defined. Applying the evaluation strategies of the Fuzzy Set IR Model on Boolean queries, *SoLibS* can retrieve library records that cannot be retrieved by using simple keyword library queries.

[5] We will rely on the recommendations made by Tim Spalding, creator of LibraryThing [MRethlefsen 07] as a starting point in identifying personalized and reminder tags. In other words, we will focus on removing tags that (i) refer to where a particular book might be physically located, (ii) whom it belongs to, and (iii) tags that are not real words, e.g., "historish".

***3.1.7 Prefix-String Indexing.*** In order to maintain the query processing time compatible with the existing library search engines, we create *prefix-string indexes* on MySQL tables (as discussed in Section 3.1.3) to speed up the process of locating the subset of relevant library catalog records and hence reduce the computational time required to calculate the degree of resemblance between a user's query and (the tag representation of) a library catalog record. In creating prefix-string indexes, which avoid indexing the entire columns in different tables used by *SoLibS*, we will consider prefix-string indices of different sizes, which are 3, 5, and 8. An ideal prefix-string size should reduce the query processing time, while the space allocated in memory for storing the indexes is minimized and the overall degree of accuracy in retrieving relevant results is not compromised.

**3.1.8** ***The Overall Query Evaluation Process of SoLibS***. Figure 1 shows the entire query evaluation process of *SoLibS*. When a library patron submits a query *Q*, keywords in *Q* are first reduced to their grammatical roots (i.e., stemming using the Porter stemmer algorithm [Porter 80]) and stopwords are removed according to a pre-defined list of stopwords, i.e., step (i). Using the set of non-stop, stemmed keywords *K* and the word-correlation factors in a reduced word-correlation matrix, the set of keywords *SK* that are highly correlated to *K*, including the keywords in *K,* is retrieved, i.e., step (ii). Hereafter, each keyword in *SK* is matched with the top-*n* LibraryThing tags that define the content of a library catalog record, and the matching yields the subset *S* of library catalog records that are highly likely relevant to *Q*, i.e., step (iii). This subset selection step is possible due to the use of LibraryThing tags to represent library records (discussed in Section 3.1.2), which are stored in a MySQL database (discussed in Section 3.1.3), and the word-similarity matching strategy (discussed in Section 3.1.1), in addition to blocking (detailed in Section 3.1.4). Note that if library patrons choose to perform an advance search, then they can construct a Boolean query *Q* to be evaluated against the library catalog (as discussed in Section 3.1.6), which can further refine the set of relevant library records retrieved by using the corresponding set of keywords in *Q* for simple keyword search only.

Last, using top-*n* LibraryThing's tags of each record in *S* and the word-correlation factors, the retrieved records in *S* are ranked according to their degrees of resemblance with *Q*, i.e., step (iv). The ranking strategy adopted in this step is the relevance ranking approach detailed in Section 3.1.5.

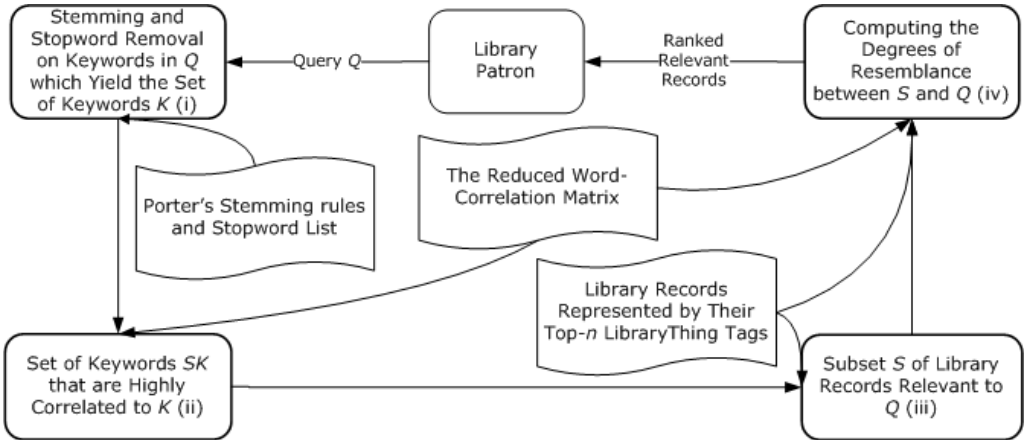


Figure 1 – The overall query evaluation process of *SoLibS*

### 3.2 Evaluation Plan

We intent to assess and analyze the performance of *SoLibS* by conducting an extensive empirical study that includes a controlled experiment using a representative dataset and various performance evaluation measures.

**3.2.1** ***The Dataset***. In evaluating the performance of *SoLibS* in processing patrons' queries using library catalogs, we will consider the queries in the most up-to-date query log provided by BYU HBLL (or any other library query logs that are available). Each entry in the HBLL query log includes (i) a query, (ii) date and time when the query was formulated, and (iii) the corresponding number of records retrieved. These queries cover a wide variety of subject areas, which include Biology, Computer Science, Education, Geography, Mathematics, Medicine, Music, Religion, etc. Due to the (i) huge number of queries (in the millions) in the query log, (ii) the diversity of users who formulated the queries, and (iii) the general subject areas covered in the queries, the HBLL query log is an ideal dataset for our empirical study.

To the best of our knowledge, there are no benchmark datasets/measurements that can be used for evaluating the retrieval performance of library systems. Hence, in assessing the performance of *SoLibS,* we plan to process and manually analyze/compare the *results* of a significant number of queries (determined statistically as detailed in Section 3.2.2) in the HBLL query log retrieved by both the HBLL system and *SoLibS,* separately.

**3.2.2** ***The Controlled Experiment.*** In order to objectively assess the performance of *SoLibS*, we will conduct a controlled experiment in which each of the participants (i.e., appraisers) involved in the empirical study will be asked to evaluate the retrieved results generated by *SoLibS*, as well as the ones generated by the HBLL system, on a set of randomly selected queries from the HBLL query log. In determining the *ideal* (i.e., most appropriate) *number of queries* to be included in the controlled experiment, we will rely on two different variables: the *average attention span* of an adult [Rozakis 02], and the average number of *search queries* that a person creates when using a search engine [Jansen 00] in a single session. The purpose of this controlled experiment is to determine the *gold standard* for comparing the performance of *SoLibS* and the HBLL system.

In designing a controlled experiment, we will adapt the cross-over experiment [Jones 03] which requires each subject, i.e., appraiser, to conduct a sequence of experiments individually. A cross-over experiment that evaluates two alternatives is called *two-period two-treated design* [Jones 03], which requires each subject to evaluate two different alternatives: *A*, which is the set of query results generated by *SoLibS* in our empirical study and *B*, which is the set of query results generated by the HBLL system on the same set of queries in the study. Half of the subjects will be given *A* first and then *B*, whereas the other half will receive *B* first and then *A*. According to [Jones 03], cross-over experiments can be used for comparing performance measures obtained by the same subject on different alternatives, which are the set of query results generated by *SoLibS* and the HBLL system, respectively in our study.

To estimate the *proper number of participants* to be included in the cross-over experiment, we will use statistically sound approaches ([Jones 03, Hinton 04]). The equations applied in these approaches are commonly-used in statistical investigations, which are based on well-established parameters, such as *standard deviation* and errors of Type I (i.e., *false positives*) and Type II (i.e., *false negatives*), to generate highly confident results that are representative of the population under evaluation, i.e., library patrons performing library catalog searches in our study. The participants involved in this empirical study, who are volunteers, will be recruited from BYU undergraduate/graduate students and faculty members from different disciplines, and they will be randomly sampled which should yield a representative group of the population at BYU who use the HBLL system.

Since our empirical study involves human subjects, we will apply for the Institutional Review Board (IRB) approval, which complies with the requirements imposed by BYU. The establishment of the IRB is to "assure that research with human subjects is conducted in accordance with federal regulations and basic ethical principles." (See http://orca.byu.edu/irb/.)

**3.2.3** ***The Performance Measures***. In assessing the performance of *SoLibS* in terms of its effectiveness and efficiency in processing library queries, we will first consider measures commonly used for verifying the effectiveness of an IR system, which include *Precision*, *Mean Average Precision*, *Mean Reciprocal Rank*, and *Spearman Rank Correlation Coefficient*. For each evaluation measure, we compare the performance measure of *SoLibS* with the corresponding one of the HBLL system. Regardless of which performance evaluation measure we consider, the *(ir)relevance* of the library records retrieved by *SoLibS* will be determined by the independent appraisers in the controlled experiment detailed in Section 3.2.2.

***Precision*** determines the *fraction* of retrieved results that are *relevant* to the corresponding query. In our empirical study, precision quantifies the set of (relevant) library records retrieved by *SoLibS* (the HBLL system, respectively) for each test query in the controlled experiment. Since in our study the appraisers will be asked to evaluate only the top 10 retrieved results, which is the number of results usually viewed by users when performing an online search [Hoscher 00], we will adapt two variations of the precision measure that are widely used: precision@1 and 10-precision. ***Precision@1*** [Cong 08], denoted *P@1*, yields the number of queries, *n*, in a (test) set of queries such that the *first* retrieved record for each one of the *n* queries is *relevant* to its corresponding query, which as mentioned earlier is determined by the independent appraisers in the controlled experiment. The higher the *P@1* value is, the better the retrieval system is in retrieving and ranking relevant results. The ***10-Precision*** value [Goncalves 04], on the other hand, quantifies the number of top 10 retrieved results in terms of their relevance with respect to the corresponding query that are also determined by the appraisers. Same as the

*P@1* value, the higher the 10-precision value, the more effective the system is in locating relevant results.

Besides the precision measures, we will also consider the ***Mean Average Precision*** (*MAP*) [Aslam 06] to provide additional performance evaluation of *SoLibS*. The ideal value of the chosen $r$ ($\geq 1$) results to be examined in *MAP(r)* is 1, which denotes that all the top $r$ retrieved records are relevant, and the closer *MAP(r)* is to 1, the better the corresponding IR system performs. By using *MAP*, we will not only evaluate the *average precision* of the retrieved results of a set of (test) queries, but also the effectiveness of our *ranking* approach which should position higher in the rank the library records with higher degrees of relevance with respect to the corresponding query.

Another measure we will adapt for further verifying the effectiveness of *SoLibS* in retrieving relevant results is the ***Mean Reciprocal Rank*** *(MRR)* [Cong 08]. As stated in [Cong 08], *MRR* determines the reciprocal rankings averaged over a set of $n$ ($\geq 1$) queries. Using *MRR* we can estimate the number of library records a library patron *must* examine in the ranked list of retrieved results of a query before finding the *first* relevant library record. The closer to 1 *MRR* is, the better the corresponding library system is in retrieving and ranking results that are relevant. We will determine the ranking accuracy of *SoLibS* using the Spearman Rank Correlation Coefficient [Callan 01].

According to Callan et al. [Callan 01], the *Spearman* ***Rank Correlation Coefficient*** is widely used for comparing two different orderings, i.e., rankings in our empirical study. The coefficient is a value between -1 and 1, where 1 indicates that the two given orderings are *identical*, 0 means that the two orderings are *not related*, and -1 implies that the two orderings are in *reversed order*. Hence, using the Spearman Rank Correlation Coefficient, we can measure whether our *ranking* strategy, which is based on the *similarity measure* and the *degrees of resemblance* between a given query and the retrieved library records, is indeed ideal for ranking retrieved library records. Using the coefficient, we determine how closely matched each ranking generated by *SoLibS* (the HBLL system, respectively) and the corresponding ranking anticipated by library patrons (through the individual appraisers) is. The ranking on a given set of retrieved results created by independent appraisers will serve as the *gold standard*, based on which we will determine the ranking performance of *SoLibS*. The definition for computing the Spearman Rank Correlation Coefficient is given in [Callan 01], and it contemplates the case when two or more elements (i.e., library records in our study) in a ranking share the *same* ranking position. The *unique* ranking of library records retrieved for a given query $Q$ will be obtained by *averaging* the ranking positions of the retrieved results of each test query determined by the appraisers.

**3.2.4** ***Queries with zero-hits.*** As discussed earlier, one of the shortcomings of existing library systems is the generation of a large percentage of zero-hits, i.e., library patrons' queries that yield no results. With that in mind, *SoLibS* is designed for minimizing the number of zero-hits using *word-similarity matching* and *folksonomies* from LibraryThing. To verify that this minimizing-zero-hits design goal is achieved, we will compare the number of test queries in the HBLL query log $G$ that yield zero-hits using the HBLL system with the number using *SoLibS*. At the same time, we will verify that among the results retrieved by *SoLibS* for each zero-hits test query in $G$, there exist relevant results.

**3.2.5** ***Query processing time.*** Besides minimizing the number of zero-hits, it is essential to design a library system $L$ that processes library patron's queries within a time frame comparable with existing library search engines, which measures the *efficiency* of $L$. We will measure the average query processing time in evaluating the test queries in the HBLL query log using *SoLibS* and compare it with the average processing time to evaluate the same set of queries by the HBLL system.

**3.2.6 Initial Experimental Results.** We have conducted an initial empirical study using the HBLL query transaction log created from July 2006 and January 2007, which includes approximately one million library catalog queries in a 144 MB file. The average size of each entry in the log is 180 bytes. Due to the huge number of queries in the HBLL query log, we have randomly chosen a subset of 500 queries, denoted *HBLL-log*, to perform the initial empirical study, which provides the preliminary measures of the *effectiveness* and *efficiency* of *SoLibS* in enhancing the quality and quantity of the library search results using library catalogs. We have conducted a controlled experiment using 48 independent appraisers, and the number was determined by the statistical methods discussed in Section 3.2.2. (For further details regarding the design and implementation of the controlled experiment, see [Thesis 08].) The experiments were conducted on an Intel Dual Core workstation with dual 2.66 GHz processors, 3 GB RAM size, and a hard disk of 300 GB running under Windows XP.

The searches of which their results were included in the HBLL query log were performed on the HBLL system that is powered by Unicorn. As stated in *SirsiDynix*'s *Unicorn* Web site (http://www.sirsidynix.com/Solutions/Products/integratedsystems.php), Unicorn is installed on many library systems around the world, such as Carnegie Mellon University, Kansas City Public Library, Princeton City Schools, Natural Resources Canada Library, Supreme Court of Canada Library, Gribskov Community Library-Denmark, etc. While *Unicorn* includes necessary features such as modules for circulation, acquisitions, outreach, materials booking, or reserves, the major design faults of existing library search engines are applicable to *Unicorn*.

We used the test queries in the *HBLL-log* to assess *SoLibS* in terms of (i) its *accuracy* in retrieving relevant results of each query in *HBLL-log,* (ii) its effectiveness in *reducing* the number of searches that retrieve no results, (iii) its *ranking* on the retrieved relevant results according to their degrees of resemblance with respect to its test query in *HBLL-log,* and (iv) its query processing *speed*.

Initial experimental results compiled by using *SoLibS* on the queries in the *HBLL-log* show that

- *SoLibS* achieves a 0.84 *P@1* value, which means that on an average 84% of the queries processed by *SoLibS* yield a relevant result at the *top* position in the ranking of the retrieved library records, whereas the HBLL system yields a 0.61 *P@1* value, i.e., 61% of the queries processed by the HBLL system come up with a relevant result positioned *first* in its ranking, which is more than 20% lower than *SoLibS*.

- The 10-*Precision* values of the retrieved results generated by *SoLibS* are consistently higher than the 10-*Precision* values of the results generated by using the HBLL system. The initial results show that *SoLibS* achieves an average of 0.67 10-*Precision* as opposed to an average of 0.58 10-*Precision* obtained by the HBLL system, which indicates that on the average 67% of the top 10 results retrieved and ranked by *SoLibS* are relevant, compared with the 58% achieved by the HBLL system.

- The *first* library record retrieved by *SoLibS* for each of the test queries in the controlled experiment is *relevant* (since its *MRR* value is 1.13), whereas when using the HBLL system, *two* library records are expected to be retrieved before locating the first *relevant* one (since its *MRR* value is 2.09). Based on the *MRR* value, it is clear that the *SoLibS* is more effective in locating relevant results earlier in the ranking than its counterpart.

- The *MAP* values obtained by *SoLibS* are *highe*r than the corresponding ones obtained by the HBLL system. A *higher MAP v*alue m*eans tha*t *less* library records are expected to be access*ed o*r *ex*amined by library patrons in finding the desired number (i.e., *r*) of *relevant rec*ords. Accor*din*g *to* the experimental results, *on the* average *SoLibS* locates t*he* $r^{th} \in (r \in \{3, 5, 7, 10\})$ desired relevant record between the $r^{th}$ and $r + 3^{th}$ record, whereas the HB*LL s*ystem requires an a*verage* between the $r + 2^{th}$ and $r + 7^{th}$ record. These results indicate that *SoLibS is hig*hly effective in *retri*eving relevant results and presenting them higher in the ranking.

- In terms of ranking, *SoLibS* yield a ranking that is more *related to*, i.e., alike, the ranking generated by the appraisers involved in the controlled experiment than the ranking created by HBLL, since the ranked, retrieved results in the controlled experiment correlate with the ranking provided by the appraisers much more often using *SoLibS* than the HBLL system. Furthermore, we have observed that the HBLL system ranks retrieved results more often in an order that tends to be the *opposite* to the ones suggested by the appraisers. In 30% of the test queries, the HBLL system yields a *Spearman Rank Correlation Coefficient* lower than 0, which did not occur even once using *SoLibS*.

- The average time required for processing each one of the test queries using the HBLL system is 6.1 seconds, whereas by using *SoLibS* the average time is 7.0 seconds. Although the average query processing time of *SoLibS* is higher than the average query processing time of the HBLL system, the difference, which is less than one second, is not significant, especially when the results retrieved by *SoLibS* are more accurate, in terms of relevancy and ranking, than the results generated by the HBLL system.

- Using the retrieved results of the 500 queries in *HBLL-log*, the percentage of zero-hits searches is reduced from 16.2% (using the HBLL system) to 1% (using *SoLibS*), i.e., from 81 to 5 zero-hit queries, which is a significant improvement. Most importantly, the (top 10) results retrieved by *SoLibS* for each of the queries for which the HBLL library system retrieved no results at all were manually examined, and the examination showed that *SoLibS* retrieved *relevant results* for the test queries that the HBLL library system yielded zero-hits.