



A REVOLUTIONARY META-SEARCH TOOL FOR EFFICIENT LITERATURE SEARCH

C. Swaraj Paul¹, S. Sridevi², R. Balakrishna³

Assistant Professor, Department of CSE, Vels University, Chennai, Tamil Nadu, India¹

Assistant Professor, Department of CSE, Vels University, Chennai, Tamil Nadu, India²

Teaching Assistant, Department of CSE, Vels University, Chennai, Tamil Nadu, India³

ABSTRACT : A rigorous and systematic search that uncovers relevant literature is a crucial part of any research project. However, finding literature in the information systems (IS) field is a complex, time-consuming and error-prone task. Due to the interdisciplinary nature of the IS field, research contributions are published in a wide variety of outlets (i.e., journals and conference proceedings) from a diverse set of disciplines (e.g., computer sciences, economics, management, sociology, medical sciences). These outlets are dispersed over numerous literature databases, each with its own functionalities, peculiarities, and constraints. To address this issue, we developed LitSonar. LitSonar is a revolutionary meta-search engine for academic literature which consolidates search results from several literature databases. LitSonar aims to improve the quality of literature reviews by enhancing rigour and efficiency of literature searches. Following the design science research paradigm, LitSonar is developed with an incremental development approach consisting of multiple design cycles of artefact creation/refinement and qualitative/quantitative evaluation. In this prototype paper, we present the overall design objectives as well as implementation details of the current prototype. In doing so, this paper can help researchers in evaluating approaches towards developing novel solutions to improve efficiency and rigour of literature searches.

KEYWORDS: Literature search, Systematic literature review, Meta-search engine, Information retrieval, Academic search engine.

1 INTRODUCTION

The core purpose of any research endeavour is the creation of new knowledge. Before extending our understanding of the world, researchers need to be aware of existing knowledge and earlier research carried out by others. A literature review is therefore essential for any academic project (Webster and Watson, 2002). The effectiveness of such a review depends on the comprehensiveness and quality of the analysed literature. If the literature base is fragmented or inadequate, the review results will be weak regardless of the effort put into the analysis, or in the words of Levy and Ellis (2006), “garbage-in/garbage-out” (p. 185). Hence, rigorous and systematic literature searches are crucial for assessing and extending the current body of knowledge (Vom Brocke et al., 2009, Webster and Watson, 2002).

Conducting a rigorous literature search becomes increasingly difficult due to the huge number of scientific documents available (over 114 million according to Khabisa and Giles (2014)). Utilizing information retrieval systems becomes indispensable for the literature search process. General guidelines for literature reviews often advise to use literature databases, such as EBSCOhost or ScienceDirect (e.g., Bandara et al., 2011, Okoli and Schabram, 2010, Webster and Watson, 2002, Wolfswinkel et al., 2013). Due to the limited coverage of individual literature databases, an exhaustive search usually involves multiple databases from different vendors, each with its own features and particularities (e.g., query syntax, port options, availability of search fields). As a

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 6, September 2015

result, the search process becomes complex, time-consuming and error-prone, especially for students and novice researchers (Levy and Ellis, 2006, Rowley and Slack, 2004, Vom Brocke et al., 2009). This is an even bigger issue for researchers in the information systems (IS) field, because of the field's interdisciplinary nature. IS-related articles are published in a wide variety of outlets (i.e., journals and conference proceedings) (Barnes, 2005, Levy and Ellis, 2006, Vom Brocke et al., 2009) from a diverse set of disciplines (e.g., computer sciences, economics, management, sociology, medical sciences). As a consequence, such articles are dispersed over numerous literature databases. For example, we found that covering the 50 top-ranked IS journals (Association for Information Systems, 2014) requires at least six different databases.

Academic search engines, such as Google Scholar (GS) or Microsoft Academic Search (MAS), could constitute an alternative. These search engines autonomously crawl the Internet for scientific documents and are therefore not limited to a single literature database. As a result, academic search engines typically have a higher coverage of scientific documents than individual literature databases. In recent years, much research has been dedicated exploiting the potential of these search engines for systematic literature reviews (Beckmann et al., 2012, Boeker et al., 2013, Bramer et al., 2013, Falagas et al., 2007, Hoff and Mundhenk, 2001, On and Lee, 2004, Samadzadeh et al., 2013). However, their suitability as the only source for literature reviews is controversial. For example, several studies found that GS has a higher coverage of scientific outlets in comparison to individual literature databases, such as Web of Knowledge or Scopus (Boeker et al., 2013, Bramer et al., 2013, Samadzadeh et al., 2013). On the other hand, GS has a very low search precision (Boeker et al., 2013, Bramer et al., 2013). Users have to check about 20 times more references on relevance compared to the normal approach using multiple searches in literature databases, which makes the review process much slower (Boeker et al., 2013). Other drawbacks on GS as an academic search engine are its undocumented and fluctuating search index, the limited number of retrievable results (i.e., a maximum of 1,000 results per search), and the lack of a bulk result export (Beckmann et al., 2012, Boeker et al., 2013, Bramer et al., 2013, Falagas et al., 2007). Similar issues have been reported for MAS (Jacsó, 2011, OrduñaMalea et al., 2014). Even though detailed information on the indexed outlets are available, the coverage of certain outlets is weak and fragmented. Furthermore, OrduñaMalea et al. (2014) report that the search index of MAS is no longer updated and advise against using MAS as source for literature reviews due to the unclear validity of search results. Hence, despite their limited coverage, literature databases are still considered as the best choice when conducting a systematic literature search (Boeker et al., 2013).

II DESIGN OBJECTIVES

In order to facilitate systematic literature searches, LitSonar is designed to achieve the following design objectives: validity, reliability, and efficiency. Validity and reliability are the two main aspects characterizing rigour of literature searches and are essential features for any high-quality literature review (Vom Brocke et al., 2009). First, validity refers to the degree to which the search uncovers all contributions that are relevant to the reviewed topic (Vom Brocke et al., 2009). A valid literature search has to be both strict enough to identify relevant articles and comprehensive enough not to overlook vital contributions (Levy and Ellis, 2006). Different search strategies can be applied to achieve these goals, such as database-focused or outlet-focused strategies. In the former case, literature databases containing contributions related to the topic area under review are queried for relevant keywords (e.g., Levy and Ellis, 2006, Okoli and Schabram, 2010, Pateli and Giaglis, 2004). This approach usually involves multiple databases in order to achieve an appropriate literature coverage. A more outlet-focused strategy starts by searching for key articles contributing to the researched topic in high-ranked outlets (Webster and Watson, 2002). Once the key articles are identified, the literature base is complemented by an iterative backward search (i.e., reviewing cited articles) and forward search (i.e., reviewing articles that have cited the previously found articles) (Vom Brocke et al., 2009, Webster and Watson, 2002). Suitability of a strategy depends on the specific literature search (e.g., research area, novelty of the reviewed topic) (Levy and Ellis, 2006). Hence, a system which seeks to support valid literature searches, not only has to provide comprehensive coverage of potential literature sources, but also has to support different search strategies.

III DESCRIPTION OF THE PROTOTYPE

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 6, September 2015

The overall goal of LitSonar is to offer an easy-to-use entry point to scientific literature, which supports reliability, validity, and efficiency of literature searches. In order to achieve this goal, we designed LitSonar as meta-search engine which queries high-quality scholarly content in literature databases and presents the results in a homogenous list. The remainder of this section describes the overall system architecture and physical infrastructure of our prototype implementation, LitSonar's graphical user interface (GUI) and the web service responsible for the result retrieval process.

3.1 System Architecture and Physical Infrastructure

The architecture of LitSonar follows a service-oriented design with three loosely coupled components: a stateful web service, a database, and a GUI. The service-oriented architecture pattern was chosen due to its high scalability and flexibility (Endrei et al., 2004). The web service constitutes the central component of LitSonar as it implements the entire result retrieval process (see Section 3.3). The web service provides its functionality to clients (i.e., service consumer) through an open application programming interface (API). Communication between clients and the web service is implemented via HTTP (transport-layer) and SOAP (messaging-layer). As a result of this, the service can be consumed by different clients, such as websites or reference managers. The current prototype implementation of the service is consumed by a web-based GUI (see Section 3.2). Information about literature databases that are required by the web service during the retrieval process (e.g., indexed outlets, connection parameters) as well as user resources (e.g., search histories, preferences) are stored in a central MySQL data- base. The physical infrastructure of LitSonar is depicted in Figure 1. In order to deliver search results as fast as possible, regardless of the number of simultaneous users, each of the three components (i.e., web service, GUI, and database) is deployed to an individual physical server cluster. Each cluster comprises multiple nodes as well as an upstream load balancer. The amount of available computing resources can be adjusted to match the demand by adding extra nodes to or removing nodes from the clusters.

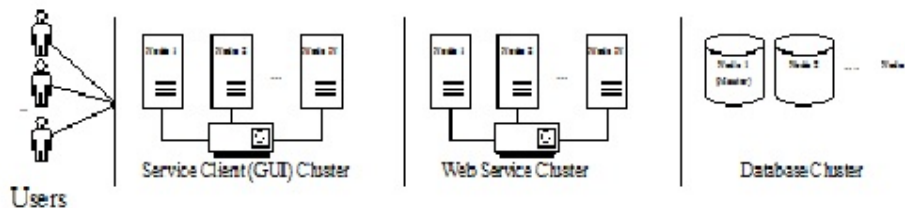


Figure 1. Physical infrastructure of the LitSonar prototype.

3.2 Graphical User interface

LitSonar's GUI helps users to formulate complex search requests and browse through the retrieved results. It is developed using the server-side scripting language PHP and the open source web application framework CakePHP (<http://cakephp.org>). The generated web pages only employ technologies standardized by the World Wide Web Consortium (i.e., HTML, CSS). As a result, the GUI ensures barrier-free access for most internet enabled devices with web browsing capabilities, including assistive devices (e.g., screen readers, Braille terminals).

To support the validity of search requests, an interactive search form guides users through a structured process to define the query parameters. The process comprises three steps: (1) definition of search terms, (2) selection of data sources, and (3) setting of additional search parameters (e.g., timespan, type of articles).

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 6, September 2015

In the first step, users are asked to provide and structure their search terms. To this end, we developed an interactive input mask supporting complex nested query structures. Figure 2 showcases a nested query containing four search terms and five synonyms which are to be found in documents' title, abstract, or keywords. Each panel of the same colour represents a (sub-) expression containing at least one child node (i.e., search term or subpanel). Siblings are connected with a Boolean operator (i.e., AND, OR) which can be selected over a switch on top of their parent panel. For each search term (e.g., "cloud") users can select a database field to be queried as well as provide synonyms (e.g., "SaaS", "PaaS", "IaaS"). All terms (i.e., search term and its synonyms) within a search term field are automatically joined with the Boolean operator OR. Even though the depicted example looks simple, an equivalent representation in form of an EBSCOhost search query reveals the actual complexity: `"((TI(decision* OR adoption) OR SU(decision* OR adoption) OR AB(decision* OR adoption)) AND ((TI(web) OR SU(web) OR AB(web)) AND (TI(service*) OR SU(service*) OR AB(service*)) OR ((TI(cloud OR outsourcing OR SaaS OR PaaS OR IaaS) OR SU(cloud OR outsourcing OR SaaS OR PaaS OR IaaS) OR AB(cloud OR outsourcing OR SaaS OR PaaS OR IaaS))))))"` By providing an interactive mask that hides such complex structures from the user, the risk of malformed queries is reduced significantly and it becomes much easier to formulate precise and valid search requests.

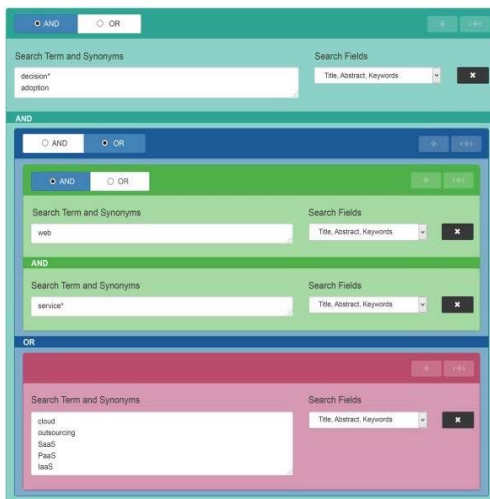


Figure 2 Interactive search term mask.



Outlet	ISSN	Issues
Management Information Systems Quarterly	0276-7783	1977 - 2014
Information Systems Research	1047-7047	1990 - 2014
Communications of the ACM	0001-0782	1958 - 2014
Management Science	0025-1909	1954 - 2014
Journal of Management Information Systems	0742-1222	1984 - 2014
Artificial Intelligence	0004-3702	1970 - 2014
1970 - 1994: Not covered		
2002 - 2014: EBSCO Academic Search Complete		
1995 - 2013: ScienceDirect		
Decision Sciences	0011-7315	1970 - 2014
Harvard Business Review	0017-8012	1922 - 2014
AI Magazine	0738-4602	1980 - 2014
European Journal of Information Systems	0960-085X	1991 - 2014
Decision Support Systems	0167-9236	1985 - 2014
IEEE Software	0740-7459	1984 - 2014
Information & Management	0378-7206	1977 - 2014
Journal of Computer and System Sciences	0022-0000	1967 - 2014
MIT Sloan Management Review	0019-848X	1960 - 2014
Communications of the AIS	1529-3181	1999 - 2014

Figure 3 Outlet coverage report.

In the second step, users can select the data sources they intend to query. LitSonar supports both database-focused and outlet-focused search strategies. Users can either pick multiple databases or compile a list of journals and conference proceedings. These ranked lists help users to assess the quality of outlets, which can be a difficult task for novice researchers (Levy and Ellis, 2006).

In the third step, users can apply additional filter options to further limit the search scope (e.g., document type, timeframe), choose between two result modes (i.e., search strings or search results) and, finally, submit their search request to the web service. Depending on the selected output-mode, either a list of database-specific search strings is returned or the retrieved search results are presented in a homogenous list. In the latter case, users can browse through the list, download articles, compose individual result lists, and export article references. Additionally, LitSonar provides an extensive report on the coverage of literature databases and outlets.

3.3 Web Service

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 6, September 2015

The main purpose of the web service is to dispatch the users' search requests to multiple literature databases and compile a homogenous list of results. The web service is developed using the Spring Framework (<http://spring.io>), an open source application framework for the Java Platform, Enterprise Edition. Since the web service runs in the Java Virtual Machine, no specific hardware is required as long as an appropriate Java Runtime Environment is available. Figure 4 illustrates the implemented result retrieval flow as well as the involved software modules. The retrieval flow implements the three typical phases of MSEs: *pre-processing*, *result-retrieval*, and *post-processing* (e.g., Keyhanipoor et al., 2005, Meng et al., 2002, Srinivas et al., 2011).

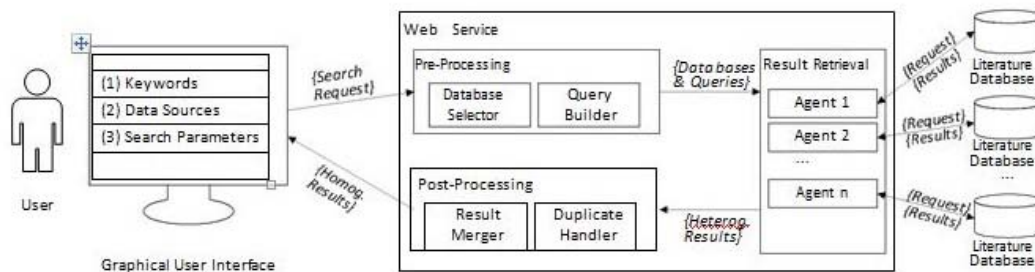


Figure 4 Result retrieval flow of LitSonar

In the pre-processing phase, search requests are prepared for the following retrieval phase. After receiving a search request, the *database selector* compiles a list of literature databases that need to be queried. If the request contains an outlet filter, a minimal set of databases with maximum outlet coverage is selected. The mapping of databases to outlets is provided by the LitSonar's frequently updated MySQL database. If more than one literature database covers the same outlet over the same time period, LitSonar selects the database with the highest full-text accessibility. Next, the *query builder* translates the search request into database-specific queries. This process incorporates, for instance, specific wildcard symbols, Boolean operators, outlet restrictions, and identifiers for search fields. To ensure the reliability of the search, we adopted the Top-Down Query Mapping algorithm (Chang and García-Molina, 1999), which maps queries across heterogeneous information sources without changing their meaning.

To provide results within a reasonable response time, LitSonar follows a pull approach. When a user sends a search request to the web service, only a limited number of results from each source are retrieved and subsequently processed (currently 100 SRRs). The retrieved results are stored in a session cache, and only the number of SRRs necessary to fill the first page is returned to the GUI. When a user wants to see the next page, a request for the next page is sent to the web service. If the cache contains a sufficient number of results, the results are returned directly. Otherwise, the retrieval process will be continued to refill the cache. The retrieval process is performed by several preconfigured and concurrently operating software modules, called *agents*. Each literature database is assigned to a specific agent which establishes a connection to the database's API, dispatches search requests, retrieves responses, and extracts the SRRs.

In the post-processing phase, the web service compiles the heterogeneous results from different literature databases into a homogeneous ordered list. In the first step, the *duplicate handler* groups redundant SRRs. Potential duplicates are grouped and not removed for lack of a reliable matching-methods due to frequent spelling errors within SRRs (Silva et al., 2009). To prevent the removal of false duplicates, it remains for the user to decide whether it is duplicate or not. Next, the *result merger* consolidates the retrieved SRRs into a homogenous list in descending order by relevancy. Relevancy is determined based on a similarity score that approximates the similarity between a SRR and the user's search query. To this end, we implemented a similarity function which combines evidence from multiple data fields (i.e., title, keyword, abstract, author), as outlined by Rasolof et al. (2003). For each field a scoring value is computed and linearly aggregated into a



International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 6, September 2015

single similarity score. The scoring values incorporates the document frequency (DF) of each search term (i.e., a measure of whether the term is common or rare across all SRRs). A high DF suggests that a term is more common and therefore a less significant indicator for the relevance of the SRR. After the retrieved results are ordered and merged into a homogenous list, the list is returned to the GUI.

IV CONCLUSION AND FUTURE RESEARCH

A rigorous and systematic literature search is often complex, time consuming, and error-prone. This paper addresses the issue by presenting LitSonar, a novel MSE for academic literature. LitSonar's design is guided by the goal to enhance the efficiency of the search process while maintaining and improving validity and reliability of the search results. Table 1 summarizes LitSonar's characteristics which contribute to the achievement of these objectives. In conclusion, the proposed system has the potential to enhance systematic literature searches and, ultimately, reviews in several ways. First, by providing a systematic search process users are encouraged to use a systematic search approach more often, which eventually leads to a more comprehensive knowledge base (Levy and Ellis, 2006, Vom Brocke et al., 2009). Second, by creating a central access point to multiple databases, LitSonar makes extensive literature searches that incorporate several databases much faster, especially when applying an outlet-focused search strategy.

A faster search process enables users to conduct more search iterations of refinement and reassessment of search constraints (e.g., keywords, search fields, time-span). LitSonar can therefore improve search queries and facilitate finding more relevant literature. Third, users only need to formulate a search query once, even if multiple databases are queried. This prevents inconsistencies between search queries and improves the reproducibility of search results. Fourth, LitSonar's reports on the coverage of databases and outlets allows users to assess the exhaustiveness of their search and, if necessary, complement the results. This contributes to the comprehensiveness and traceability of literature searches. Finally, fellow researchers are able to reproduce results with the same rigour as the initial search in a reasonable time as well as encourages them to use and extend prior results.

However, the current prototype implementation has some limitations. First of all, search results are retrieved successively. This approach has both advantages and drawbacks. On the upside, a successive approach is efficient and enables users to see the most relevant results long before the retrieval process has finished. On the downside, functions that require a complete result list, such as bulk export of results or individual sorting and filtering options, can only be provided after the lengthy retrieval process has finished, making such features impracticable for MSEs. Especially the lack of meta-information is a major issue. Information, like the most frequent keywords within a result set, allow users to quickly determine the quality of the executed search query and refine their requests accordingly. These features are well-known and provided by almost any modern literature database. Their absence might discourage users from employing an academic MSE.

In future work, we will therefore explore how these features can be incorporated into the MSE concept in order to enhance functionality and user experience of academic MSEs. Another limitation of the current prototype is its dependency on commercial literature databases (e.g., EBSCOhost, ScienceDirect). Such databases are often financed by a subscription model and retrieved results may only be provided to subscribers or entitled members of subscribing institutions (e.g., universities). Hence, the coverage of LitSonar currently depends on the user's affiliation. To address this issue, we will investigate whether openly accessible bibliographic data can be integrated into our approach without reducing the reliability, viability and efficiency of literature searches. Furthermore, we will integrate interlibrary loan services which enable users to acquire physical full-text copies even if their access to digital resources is limited. Both the field test and the controlled experiment will help us not only to improve LitSonar but also to generate and share new knowledge on designing tools that support rigorous and efficient systematic literature searches.

REFERENCES



International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 6, September 2015

- [1] Association for Information Systems. (2014). MIS Journal Rankings. URL: <http://aisnet.org/?JournalRankings> (visited on 03/12/2015).
- [2] Bandara, W., Miskon, S. and Fieft, E. (2011). "A Systematic, Tool-Supported Method for Conducting Literature Reviews in Information Systems." In: Proceedings of the 19th European Conference on Information Systems. Helsinki, Finland, pp. 1–13.
- [3] Barnes, S. J. (2005). "Assessing the Value of IS Journals." Communications of the ACM 48 (1), 110–112.
- [4] Beckmann, M., Wehrden, H. V. and Palmer, M. (2012). "Where You Search is What You Get: Literature Mining." Journal of Vegetation Science 23 (6), 1197–1199.
- [5] Boeker, M., Vach, W. and Motschall, E. (2013). "Google Scholar as Replacement for Systematic Literature Searches: Good Relative Recall and Precision are not Enough." BMC Medical Research Methodology 13, 131.
- [6] Boell, S. and Cecez-Kecmanovic, D. (2014). "A Hermeneutic Approach for Conducting Literature Reviews and Literature Searches." Communications of the AIS 34 (1), 257–286.
- [7] Bramer, W. M., Giustini, D., Kramer, B. M. and Anderson, P. (2013). "The Comparative Recall of Google Scholar Versus PubMed in Identical Searches for Biomedical Systematic Reviews." Systematic Reviews 2, 115.
- [8] Chang, C.-C. K. and García-Molina, H. (1999). "Mind Your Vocabulary: Query Mapping Across Heterogeneous Information Sources." In: Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data. New York, USA, pp. 335–346.
- [9] Lu, Y., Meng, W., Shu, L., Yu, C. and Liu, K.-L. (2005). "Evaluation of Result Merging Strategies for Metasearch Engines." In: 6th International Conference on Web Information Systems Engineering. New York, USA, pp. 53–66.
- [10] Meng, W., Yu, C. and Liu, K. (2002). "Building Efficient and Effective Met search Engines." ACM Computing Surveys 1 (34), 48–84.