Automating Search Strings for Secondary Studies

Francisco Carlos Souza[1], Alinne Santos[1], Stevão Andrade[1], Rafael Durelli[2], Vinicius Durelli[1], and Rafael Oliveira[3]

[1]University of São Paulo – USP, São Carlos, SP - Brazil, email:{fcarlos, alinne, stevao, durelli}@icmc.usp.br
[2]Federal University of Lavras, Lavras, MG - Brazil, email: rafael.durelli@dcc.ufla.br
[3]Federal Technological University of Parana, Dois Vizinhos, PR - Brazil, email: raoliveira@utfpr.edu.br

***Abstract*—Background: secondary studies (SSs), in the form of systematic literature reviews and systematic mappings, have become a widely used evidence-based methodology to to create a classification scheme and structure research fields, thus giving an overview of what has been done in a given research field.**
Problem: often, the conduction of high-quality SSs is hampered by the difficulties that stem from creating a proper "search string". Creating sound search strings entails an array of skills and domain knowledge. Search strings are ill-defined because of a number of reasons. Two common reasons are *(i)* insuffient domain knowledge and *(ii)* time and resource constraints. When ill-defined search strings are used to carry out SSs, a potentially high number of pertinent studies is likely to be left out of the analysis.
Method: to overcome this limitation we propose an approach that applies a search-based algorithm called Hill Climbing to automate this key step in the conduction of SSs: search string generation and calibration.
Results: we conducted an experiment to evaluate our approach in terms of sensibility and precision. The results would seem to suggest that the precision and the sensibility our approach are 25.2% and 96.2%, respectively.
Conclusion: The results were promising given that our approach was able to generate and calibrate suitable search strings to support researchers during the conduction of SSs.

***Keywords*—Secondary Studies, Search String, Hill Climbing.**

I. Introduction

Systematic literature reviews (SLRs) and systematic literature mappings (SLMs), also known as secondary studies (SSs), have been widely used in medical research and in the natural sciences since the 1970s and early 1980s. SLR and SLM are considered rigorous methods to map the evidence base in an unbiased way, evaluate the quality of the existing evidence, and synthesize and give an overview of a given research field. Based on the guidelines of these methods, Kitchenham et. al. [11] proposed evidence-based Software Engineering (EBSE) in hopes of fostering the adoption of SSs in Software Engineering (SE). In the context of Software Engineering, SSs rely on the use of an objective, transparent, and rigorous approach for the entire research process in order to minimize bias and ensure future replicability. Rigour, transparency, and replicability are achieved by following a fixed process for all reviews. The fixed process is one of the characteristics that distinguish SSs from traditional literature reviews (TLR).

The conduction of SSs usually include the following steps: first, the research question is deconstructed by considering Population, Intervention, Comparison, and Outcome (PICO) criterion. Terms from this criterion form of the basis of search strings that are used in the literature search. Then a protocol is produced to describe definitions, search strings, search strategy, inclusion and exclusion criteria, and the approach that will be used to synthesize data. This protocol is often peer-reviewed and piloted. This may lead to several revisions in the search strategy. Next, a systematic search is conducted; studies are retrieved from digital scientific databases sources (e.g., IEEE Xplore, ACM, Scopus, Scirus, etc.) [8].

A remarkable problem regarding SSs is the generation of suitable search strings. Ill-defined search string can hinder the search process by returning a significant amount of irrelevant studies. In addition, another problem stems from the different rules employed by digital databases, which may render the search step into a process of trial and error. For instance, to generate a suitable search string researchers need to grasp a set of keywords and synonymous, which is a time and resource consuming task. Also, usually researches need to adjust the same generic search string to several digital databases. According to Kitchenham et. al., [11], four common aspects are associated with having to analyze a high number of irrelevant papers: (*i*) unsatisfactory search string creation; (*ii*) digital databases of studies have different interfaces; (*iii*) digital databases deal with Boolean formulas [5, 8] in a slightly different way from each other; and (*iv*) digital databases have different methods to search the body of the manuscript or even some indexing elements (e.g., title, keywords, and abstract).

In order to overcome these four issues we propose an approach called Search-based String Generation (SBSG) that applies an Artificial Intelligence (AI) technique called Hill Climbing (HC) [12]. The main goals of SBSG is to produce and recommend a suitable search string to be used during the conduction of an SS. In this study we prototype the SBSG approach in a proof-of-concept tool. Specifically, SBSG runs as following: (*i*) researchers need to define a set of parameters: terms, keywords, synonyms, number of iterations (how many times SBSG will run), and a list of control studies[1]; (*ii*) these parameters are used as input to an HC algorithm to create an initial search string, i.e., initial solution; (*iii*) from the initial search string HC generates a set of neighbor search strings, i.e., similar search strings; and (*iv*) if a neighbor improves the value of the initial solution, i.e., a better suitable search string is generated, then this new search string is selected and becomes the current search string. This process runs interactively until SBSG finds the best suitable search string or reaches the specified number of iterations.

In order to provide some evidence of the applicability of our approach we performed an experiment. More specifically,

[1]Control studies are papers that must be retrieved when search in a given database.

we used terms, keywords, synonyms, and lists of control studies of five published SSs. Afterwards, we assessed if the search strings generated by our approach were as suitable as the ones used by the published SSs. Two metrics were used to gauge the quality of the generated search strings: precision and sensibility. Experimental results that our approach improved the search strings.

The main contributions of this paper are fourfold: (*i*) a well-defined approach to improve a key step of SSs – search string generation and calibration; (*ii*) an approach to tap into the strengths of a search-based algorithm and improve the applicability of EBSE; (*iii*) a proof-of-concept tool that automatically supports the generation of search strings for SSs; and (*iv*) an experiment to evaluate our approach and implementation thereof in terms of precision and sensibility.

This remainder of this paper is structured as follows: in Section II SSs, HC, and fitness function are described. Section III details some technical issues concerning the use of the SBSG approach as an effective solution to improve automated searches in SSs. The experiment we carried out is presented in Section IV. Discussions, general impressions, and threats to validity are presented in Section V. Finally, Section VI presents concluding remarks.

II. Background

A. Secondary Studies

According to Kitchenham et al., [10] *primary studies* are studies that present qualitative or quantitative results concerning an specific research field. Secondary studies (SSs), on the other hand, are a compilation of several primary studies gathered to investigate a given research area and answer specific research question. There are two types of methodologies that yield SS: SLR and SLM. Both methodologies are transparent and rigorous, aimed at minimizing bias and ensuring replicability.

In spite of the growing importance that SSs have achieved, many aspects of the conduction of SLR and SLM are still challenging. One of the these challenges is the creation of proper search strings. Usually, search strings are created and calibrated based on a set of terms, keywords, synonyms, etc. Oftentimes, this process is carried out manually by the researchers, thus the creation of search strings turns out to be time-consuming and error-prone. In addition, if researchers are new to a particular field, they need to spend more time to generate and calibrate search strings. Another challenge is the different rules employed by the digital scientific databases that makes the search string calibration almost a process of trial-and-error. Usually, researchers need to adjust the same search string for several databases in hopes of identifying relevant studies. We argue that this whole process of creating a suitable search string could be semi-automated by means of Hill Cimbling algorithm.

B. Hill Climbing and Fitness Function

Search Based Software Engineering (SBSE) is the field of software engineering research and practice that applies search based techniques to solve different optimization problems from diverse SE areas. SBSE approaches allow software engineers to automatically obtain solutions for complex and labor-intensive tasks, thereby contributing to reduce the effort and cost associated to SE activities [6, 4]. SBSE consists of search-based algorithms used in SE, such as generic algorithms, generic programming, simulated annealing, and Hill Cimbling (HC) [12].

HC is a local search algorithm that combines a general search method with either objective functions or fitness functions for evaluating the states generated by the method [12]. HC aims to identify the best path to be followed in the search. As outcome the technique returns a satisfactory result for a given problem. HC algorithm consists in selecting an initial solution randomly, evaluating, and improving it step by step, from the investigation of the neighborhood of the current solution. If a neighbor improves the value of the initial solution, this new solution is then selected and becomes the current solution. This process is performed until an optimal solution is found or when no neighbor has a better value [12].

The main benefits of HC algorithm are the following: (*i*) it doest require much memory due to fact that only the current state is stored; (*ii*) it is easy to be implemented; and (*iii*) if the best or optimal solution exists in the search space, HC is able to find that solution in a feasible computational cost. Regarding SE issues, HC algorithm is a simple and powerful SBSE strategy to search optimal solutions in combinatorial search spaces. Then, it is feasible to employ HC algorithm in different SE applications.

HC algorithms must be combined with a proper Fitness Function (FF) that is able to measure the quality of the solutions found [4]. FFs are important components for search and metaheuristics techniques since they predict how close a solution is to be optimal. Metaheuristics are high level strategies to efficiently explore the search space in order to find optimal or near optimal solutions by using different methods, techniques which constitute metaheuristic algorithms range from complex learning processes to simple local search procedures such as a HC algorithm [2]. Thus, in general terms, a FF is an expression that measures the goodness of a candidate solution for solving a given problem.

Setting a FF poses some challenges, it is not a trivial task due to the fact that each function depends on the specific problem and its features. The idea is employing heuristics information regarding the features of an specific problem into a function so that it can be able to assess the adequacy of candidate solutions. Then, there are cases in which the FF is fairly trivial and there are cases in which the designer needs to dedicate efforts to figure which FF is more suitable for a given problem.

III. The SBSG Approach

As mentioned, researchers must generate and calibrate a search string manually. Aiming at applying the concepts of the HC algorithm to alleviate the problem of generating suitable search strings for different digital scientific databases, we propose the SBSG approach. Our approach is semi-automatic and combines an HC algorithm with an assessment strategy for searching and generating suitable search strings.

Our approach provides a semi-automatic method to produce a sound string reducing the aforementioned problems

about the manual process. The idea behind the SBSG approach does not replace researchers in the string generation process, on the contrary, it assists them. A generic workflow of SBSG is shown in Figure 1. As shown in the figure, researchers must provide the following parameters: (*i*) a list of keywords, (*ii*) a list of control studies, (*iii*) the set of terms of the string and their respective synonymous, and (*iv*) the number of iterations, i.e, how many times SBSG will run.

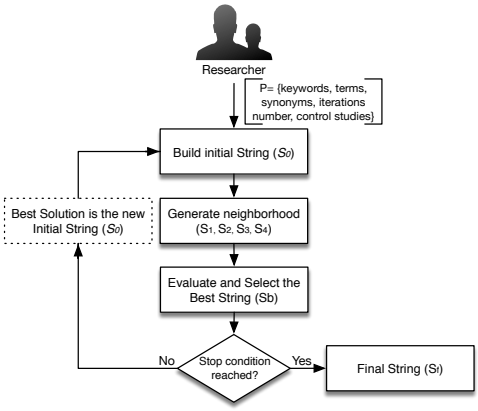

Fig. 1. SBSG's workflow

Through those parameters, SBSG starts its process with an initial string (S_0) based on the PICO criterion. Threfore, it is necessary at least one keyword to fill out the population, intervention, comparison, and outcome. Figure 2 shows an illustrative example of PICO criterion.

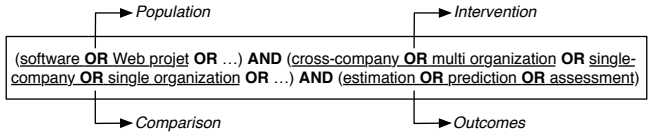

Fig. 2. PICO criterion (Kitchenham et al. [9])

SBSG employs an HC algorithm that works performing small changes in each part of the string to create a neighborhood of string candidates. First, HC generates the string S_0 though of a set of terms previously defined by the user. Then, a neighborhood (S_1, S_2, S_3, S_4) from S_0 is expanded, which it is based on strategies presented in the Table I.

TABLE I. STRATEGIES FOR STRINGS NEIGHBORHOOD

Functions	Changes	Where
Adds or deletes	Synonymous	Population
Adds or deletes	Synonymous	Intervention
Adds or deletes	"-"	in compound words
Adds or deletes	plural	of a synonymous
Replaces	the suffix of a Synonymous	Population

Each string is assessed through a special FF to search the best one (S_b). This FF is based on an optimal search strategy introduced by Straus and Richardson [13] and Haynes et al. [7] in the context of SSs. The FF proposed is composed of two measures called *sensibility* and *precision* (Figure 3).

The *sensibility* on search strategy context is a measure to identify all of the relevant studies (**A** – Figure 3) for a specific domain from retrieved studies (**R** – Figure 3) supported by a set of relevant studies previously defined (**C** – Figure 3). The higher **C** is, the higher will be the sensibility score or the opposite. On the other hand, the *precision* is an ability to identify the amount of irrelevant studies (**B** – Figure 3), where **B = R-A**. When **B** is zero, i.e., no irrelevant study is detected, the precision score is greater. A search string with low precision will lead a lot of irrelevant studies retrieved. *Sensibility* and *precision* are computed using the equations 1 and 2, respectively.

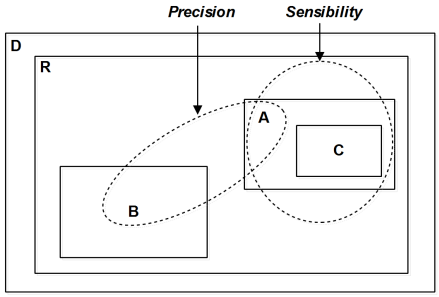

Fig. 3. Sensibility and Precision search strategies

$$S = \frac{A}{A+C} * 100. \quad (1) \qquad P = \frac{A}{A+B} * 100. \quad (2)$$

The overall process by which a candidate string evolves reduces the effort of string adaptation. This frees researchs up to do other activities as assessing the quality of the returned studies based on their abstracts and keywords, for instance. This way researchers are able to eliminate those studies with zero or a very low fitness value. Therefore, the total fitness (**F**) is computed through tEquation 3. When **F** is zero, the string returns only irrelevant studies and the higher **F** is, the higher the adequacy of search string will be.

$$F = \frac{(S) + (P)}{2} \quad (3)$$

Once no further improvements can be made to a search string, the search continues exploring the next neighborhood, starting over with the new current string, until no neighbor leads to improvements. At this point the search restarts at another randomly chosen location in the search space. This is known as a strategy to overcome local optima, enabling the search to explore a wider region of the possible strings for a specific topic.

The quality of the SBSG approach is quantified through search strategies scale using in Dieste and Padua [3], which was inferred from the sensitivity and precision ranges of SLRs in medicine. We have adopted this search strategy because it can qualify how relevant a study is to a particular domain. Table II provides a scale to measure the quality of search based on the amount of relevant and irrelevant studies. Assuming the scales for adequate strings, we considered a threshold between 80% and 99%, 20%, and 60% as references for sensitivity and precision, respectively.

A. Proof-of-Concept Implementation

We developed a tool to automate SBSG. This tool has two modules: (*i*) a Java module and (*ii*) a Python module. The

former is used to generate search strings following the rules of the IEEE search engine. Based on a parameter defined by hand, this module reads a text file that must follow a predefined syntax and then it identifies the PICO's terms in the file. From this starting point, the first module creates a data structure that is able to apply rules from IEEE Xplore[2] to generate valid search strings.

TABLE II. SEARCH STRATEGY SCALES.

Strategy type	Sensitivity	Precision	Goal
High sensitivity	85-90%	7-15%	Max sensitivity despite low precision
High precision	40-58%	25-60%	Max precision rate despite low sensitivity
Optimum	80-99%	20-25%	Maximize both sensitivity and precision
Acceptable	72-80%	15-25%	Fair sensitivity and precision

The Python module contains a set of scripts. These scripts are used to call the first module. As outcome, a object in Python is obtained. After, a script is used to request and send the generated search string to the IEEE Xplore digital database. The outcome is an XML (eXtensible Markup Language) file containing all fetched primaries studies's data such as: title, keywords, abstract. Then these data are parsed and analyzed in terms of *sensibility* and *precision*. This second module runs interactively until it finds the best suitable search string or it reaches the specified number of iterations.

IV. EMPIRICAL EVALUATION

This section presents the search strings that were used as subject in our evaluation, all of the decision we took when designing our experiment to address three research questions, and the results obtained.

A. Research Questions

We set out to answer the following research questions (RQ) through our proof-of-concept implementation: (i) **RQ_1:** How good is the sensibility of search strings generated when using our approach?; (ii) **RQ_2:** How good is the precision of search strings generated when using our approach?; and (iii) **RQ_3:** In practice, how efficient is our approach to generate the best search strings?

B. Goals

The goal of our experimentation can be therefore defined by using the GQM (Goal Question Metric) [1], which can be summarized as: **Analyze** the SBSG approach, **for the purpose of** evaluating it sensibility and effectiveness, **with respect to** improvement of SS's search string, **from the point of view of** researchers, **in the context of** heterogeneous subject secondary studies' search string.

C. Experimental Design and Execution

To provide empirical evidence and answer the aforementioned RQs, we have used the search strings of five SSs that have already been published. During the selection of these SSs we focused on covering a broad class of SSs, selecting SSs from different SE fields. We chose SS whose search strings contain keywords from the following fields: software testing, software reusability, and model-driven development.

IEEE Xplore is deemed as the one of main digital electronic databases research in SE. Therefore, we decided to use the IEEE Xplore digital source in our experiment. Also, four other reasons contributed to our choice on using IEEE Xplore to automate the SBSG strategy in this study: *(i)* satisfactory search algorithm; *(ii)* bibliographic resources are not limited; *(iii)* recognition of plurals; and *(iv)* IEEE retrieves the largest number of studies with abstract and complete texts.

This experiment was carried out in four steps. Firstly, we selected five set of terms, keywords and control list. Secondly, we performed the search with 5, 15, and 30 iterations. Then we performed the search for the best string according to three settings: (i) measure the sensibility, precision, and time to the generation; (ii) measure how many iterations needs to find the best one; and (iii) repeated this process 10 times. Finally, the average of the sensibility and precision were computed by the sum obtained in the previous sub-steps.

In order to analyze the gathered data, descriptive statistics have been used: the mean; minimum and maximum values; and standard deviation. Section V describes the results collected from our proof-of-concept validation.

V. RESULTS AND DISCUSSION

A. Sensibility and Precision

First, we looked at the descriptive statistics for the sensibility (RQ_1) of the generated search strings using our approach. We computed the mean for each sensibility's iterations ($i = 5$, $i = 15$ and $i = 30$). All search strings had optimum results. The mean sensibility of all evaluated strings with five iterations was high, ranging from 93% to 98%. We noticed from the fifth iteration that there were no increases in the sensitivity's mean. These results indicate that search strings with optimum sensibility can already be generated in the first iteration.

TABLE III. STATISTICS FOR SENSIBILITY WITH 5 ITERATIONS

Subj.	Minimum	Maximum	Mean	Std. Deviation
String 1	,92	,93	,9316	,00562
String 2	,95	,97	,9677	,00873
String 3	,95	,97	,9656	,00770
String 4	,95	,98	,9800	,01146
String 5	,98	,98	,9815	,00000

Figure 4 shows that, for a small number of iterations the approach obtained a high sensibility for all cases, it means that a string can identify most relevant studies. In terms of SLR and SLM, a high sensibility is usually desired. Therefore, the higher it is, more studies related to the domain must be returned in the search.

We also looked at descriptive statistics for precision (RQ_2) according to each iteration ($i = 5$, $i = 15$ and $i = 30$). Most search strings achieved good results for five iterations (see Table IV). String 1 and String 2 achieved optimum precision (20%, 21%, respectively), while String 3 and String 5 had very high precision (41% and 35%, respectively). On the other hand, String 2 reached low precision (12%). We noticed from

[2]see: http://ieeexplore.ieee.org/

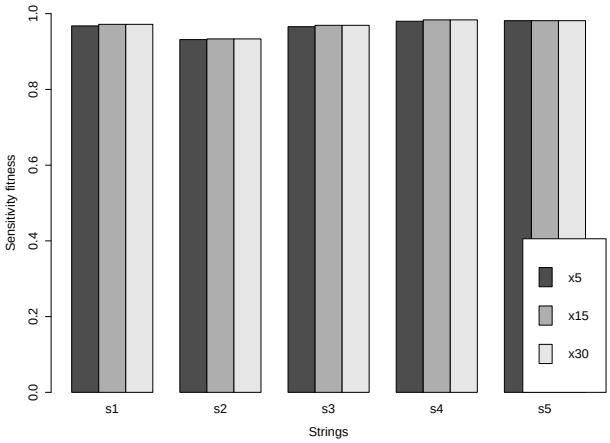

Fig. 4. The best sensitivity by iteration

the fifth iteration that there were no increases in precision. As a result, we surmise that our approach is able to generate sound search strings with good sensibility and optimum and high precision in the first interactions.

TABLE IV. STATISTICS FOR PRECISION WITH 5 ITERATIONS

Subj.	Minimum	Maximum	Mean	Std. Deviation
String 1	,20	,20	,1979	,00232
String 2	,21	,21	,2133	,00000
String 3	,41	,41	,4145	,00000
String 4	,12	,12	,1195	,00000
String 5	,35	,35	,3464	,00000

The precision value for optimal strings must be low compared with the sensibility. Figure 5 shows low values for all cases, this measures gives a proportion of relevant studies retrieved regarding the number of irrelevant studies from the search. Therefore, these results can be considered satisfactory, since all strings achieved good precision in a few iterations.

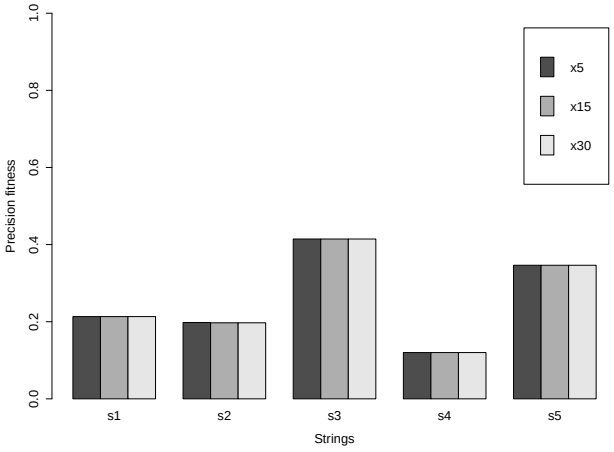

Fig. 5. The best precision by iteration

Additionally, it is worth noting that the results in the Figure 4 and 5 are similar. We believe that this occurred due to the fact that the initial string was structured to cover the domain as a whole, thus, it is almost impossible to start with an initial string with a low fitness. Assuming sound initial strings, the HC needs significantly less effort and demands less improvements to find the best search string. Consequently, based on this assumption, the results tend to be similar from a number of iterations.

B. *Efficiency*

The efficiency (RQ_3) of our approach was measured using the time for the search string generation and the number of iterations. The time includes the time of each execution to generate and calibrate the search string and their improvement. Figure 6 depicts how sensibility and precision improves in 10 iterations. One can mention that the approach has a fast convergence, since in five or six iterations the optimal solution has been found, i.e it was evaluated only five or six neighborhoods.

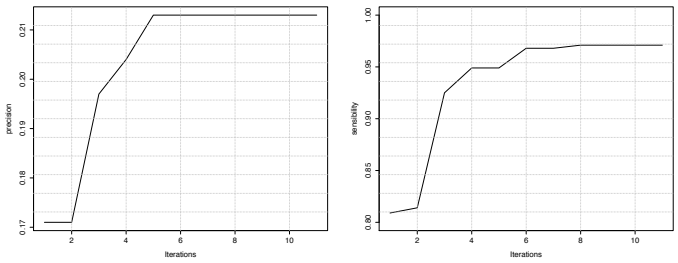

Fig. 6. Evolution of the sensibility and precision

In our approach, to avoid results being skewed by the randomness inherent in search techniques, we conducted the experiment using three settings to measure the time and how many iterations were necessary for the search to converge for an optimal solution. As shown in Figure 7, the amount of time depends on some important factors, such as, size and complexity of the string, but even for the worst cases the approach is faster than generating a string by hand. However, as already mentioned, the HC had fast convergence, it means that a few iterations were necessary to find a good solution, in general, performing the experiment with the first setting (5 iterations) has been enough to obtain an acceptable string.

Our approach generates an initial string based on the set of parameters provided by the researcher. Based on the first string, our approach generates four additional strings, one in each iteration. These strings are similar to the initial string. This process of deriving more strings from the initial string is analogous to the many steps that researchers usually take to manually fine-tune their search strings. As for our results, we evaluated 25 different strings (in five iterations), which took on average 116 seconds. Often, when manually fine-tuning search strings, many changes are needed. So, given that this process of improving a given search string by hand might take hours, we conjecture that using our HC-based algorithm is able to significantly speed up the fine-tuning of search strings.

C. *Threats to validity*

Conclusion validity: a possible threat to the conclusion of our study is associated with the fact that we have used only IEEE Xplore as the subject database. Generally, researchers use at least three digital libraries as source of information to select primary studies.

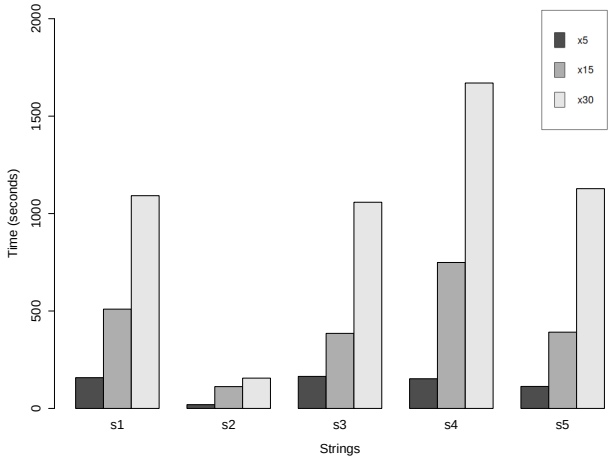

Fig. 7. Time to generate the search strings

Construct validity: regarding the theory behind our experiment and the observations, we believe the main threat on using SBSG is the fact that the approach must be implemented and adapted to the search engine of each digital scientific database. This means that if some update or new technology is implemented on the database, our implementation must be adapted to work properly on it.

External validity: considering the generalization of our findings, we believe that the main threat is the fact that we believe we would have been able to evaluate the drawbacks of SBSG better if we had used subjects in our experiment. More specifically, it would be beneficial for our approach if we could get feedback from individuals carrying out their own systematic reviews. Moreover, we cannot rule out the fact that we are already familiar with SBSG, which might have contributed to better results.

VI. Conclusion and Future Work

In this paper we presented SBSG, an approach that applies a search-based algorithm, i.e., Hill Climbing, to automate a key step in the conduction of SSs: search string generation and calibration. Using a list of terms, synonyms, keywords, and a set of control studies, SBSG automatically provides a sound search string in the appropriate format according to a pre-determined digital database.

We also carried out an experiment using five SSs that have already been published. Our experiment used SBSG to find search strings with good accuracy in accordance to measures of sensibility, precision, and efficiency. The gathered evidence would seem to suggest that SBSG yields sound search strings. The resulting search trings were automatically generated/calibrated and the primary studies identified by them returned most relevant studies: there were only a few studies missing in comparison to the actual results of the five real SSs. The results indicate that the precision and the sensibility of our approach are 25,2% and 96,2%, respectively. We believe SBSG is a significant contribution towards automating the conduction of SLRs and SLMs. In the future, we plan to evaluate SBSG using a bigger set of digital databases.

References

[1] V. Basili, G. Caldiera, and H. Rombach. *The Goal Question Metric Paradigm*. John Wiley and Sons, 1^{St} edition, 1994.

[2] C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, 35(3):268–308, 2003.

[3] O. Dieste and A. Padua. Developing search strategies for detecting relevant experiments for systematic reviews. In *ESEM 2007*, pages 215–224, Sept. 2007.

[4] G. Gay. A baseline method for search-based software engineering. In *PROMISE 2010*, PROMISE '10, pages 2:1–2:11, New York, NY, USA, 2010. ACM.

[5] J. E. Hannay, T. Dybå, E. Arisholm, and D. I. K. Sjøberg. The effectiveness of pair programming: A meta-analysis. *Inf. Softw. Technol.*, 51(7):1110–1122, July 2009.

[6] M. Harman, P. McMinn, J. T. de Souza, and S. Yoo. Empirical software engineering and verification. chapter Search Based Software Engineering: Techniques, Taxonomy, Tutorial, pages 1–59. Springer-Verlag, Berlin, Heidelberg, 2012.

[7] R. B. Haynes, K. A. Wilczynski, C. J. McKibbon, and J. C. Sinclair. Developing optimal search strategies for detecting clinically sound studies in medline. *Journal of the American Medical Informatics Association*, 1:447–458, 1994.

[8] B. Kitchenham. *Making Software What Really Works, and Why We Believe It*, volume 1, chapter Chapter three – What we can learn from systematic reviews. O'Reilly Media, Gravenstein Highway North, Sebastopol, CA, first edition, 2011.

[9] B. Kitchenham, E. Mendes, and G. H. Travassos. A systematic review of cross vs. within-company cost estimation studies. *IEEE Transactions on Software Engineering*, 33(5):361–329, May 2007.

[10] B. A. Kitchenham, D. Budgen, and O. Pearl Brereton. Using mapping studies as the basis for further research - a participant-observer case study. *Inf. Softw. Technol.*, 53(6):638–651, June 2011.

[11] B. A. Kitchenham, T. Dyba, and M. Jorgensen. Evidence-based software engineering. In *ICSE 2004*, ICSE '04, pages 273–281, Washington, DC, USA, 2004. IEEE Computer Society.

[12] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 edition, 2003.

[13] S. Straus and W. Richardson. *Evidence-Based Medicine:How to Practice and Teach It*. Churchill Livingstone, 4^{th} edition, 2010.