Automating Search Strings for Secondary Studies

104

Francisco Carlos Souza, Alinne Santos, Stevão Andrade, Rafael Durelli, Vinicius Durelli, and Rafael Oliveira

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.

104.1 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

F.C. Souza (✉) • A. Santos • S. Andrade • V. Durelli
University of São Paulo – USP, São Carlos, São Paulo, SP, Brazil
e-mail: fcarlos@icmc.usp.br; alinne@icmc.usp.br; stevao@icmc.usp.br; durelli@icmc.usp.br

R. Durelli
Federal University of Lavras, Lavras, MG, Brazil
e-mail: rafael.durelli@dcc.ufla.br

R. Oliveira
Federal Technological University of Parana, Dois Vizinhos, PR, Brazil
e-mail: raoliveira@utfpr.edu.br

S. Latifi (ed.), *Information Technology – New Generations*, Advances in Intelligent Systems and Computing 558, DOI 10.1007/978-3-319-54978-1_104

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, 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 Sect. 104.2 SSs, HC, and fitness function are described. Section 104.3 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 Sect. 104.4. Discussions, general impressions, and threats to validity are presented in Sect. 104.5. Section ?? discusses related work. Finally, Sect. 104.6 presents concluding remarks.

104.2 Background

104.2.1 Secondary Studies

According to Kitchenham et al. [10] "primary studies" are experiments and empirical validations with qualitative or quantitative results to an specific research field. In other hand, "Secondary Studies" (SSs) in SE represent a compilation of several primary studies gathered to find relevant research evidence and answer specific research question. There are two types of SS, they are: SLR and SLM. Both SSs rely upon the use of an objective, transparent and rigorous approach for the entire research process in order to minimize bias and ensure future replicability.

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

In spite of the growing importance that SSs have been achieving nowadays, they are still new topics to the SE community. Many challenges appear as to how to create a suitable search string. Usually, the search string is generated and calibrated based on a set of terms, keywords, synonyms, etc. Moreover, this whole process is done manually by the researchers, which is time-consuming and error-prone. In addition, if researchers are new in a particular field, then they need to spend more time and dedicate efforts to generate and calibrate search strings. Another challenge is the different rules employed by the digital scientific databases that make the search string calibration almost a process of trial-and-error. Usually, researchers need to rework on setting the same search string in several databases source to identify good studies, avoiding missing papers. We argue that this whole process of creating a suitable search string could be semi-automated by means of HC algorithm.

104.2.2 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, contributing to reduce efforts and costs associated to the software development [4, 6]. SBSE consists of search-based algorithms used in SE, such as generic algorithms, generic programming, simulated annealing, and 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 it finds an optimal solution or when no neighbor has a better value [12].

The main benefits of HC algorithm are: (*i*) it requires low memory due to fact that only the current state is stored; (*ii*) it is easy to be implemented; and (*iii*) whether 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 is a solution to be optimal. Metaheuristics are high level strategies to efficiently explore the search space in order to find optimal or near 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.

There are many guidance on setting a FF, however it is not a general task due to the fact that each function depends on the specific problem and its features. The idea is employing heuristics information regarding to 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.

104.3 The SBSG Approach

Researchers must generate and calibrate a search string totally manually, which is an error prone activity, time consuming, and labor-intensive. Aiming at applying the concepts of 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 assess strategy for searching and generating suitable search strings.

Our approach provides a semi-automatic method to produce a good 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 Fig. 104.1. According to 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.

Through those parameters, SBSG starts its process with an initial string (S_0) based on PICO criterion. It means that it is necessary at least one keyword to fill out the population, intervention, comparison and outcome. Figure 104.2 shows an illustrative example of PICO criterion.

SBSG employs an HC algorithm that works performing small changes in each part of the string to create a

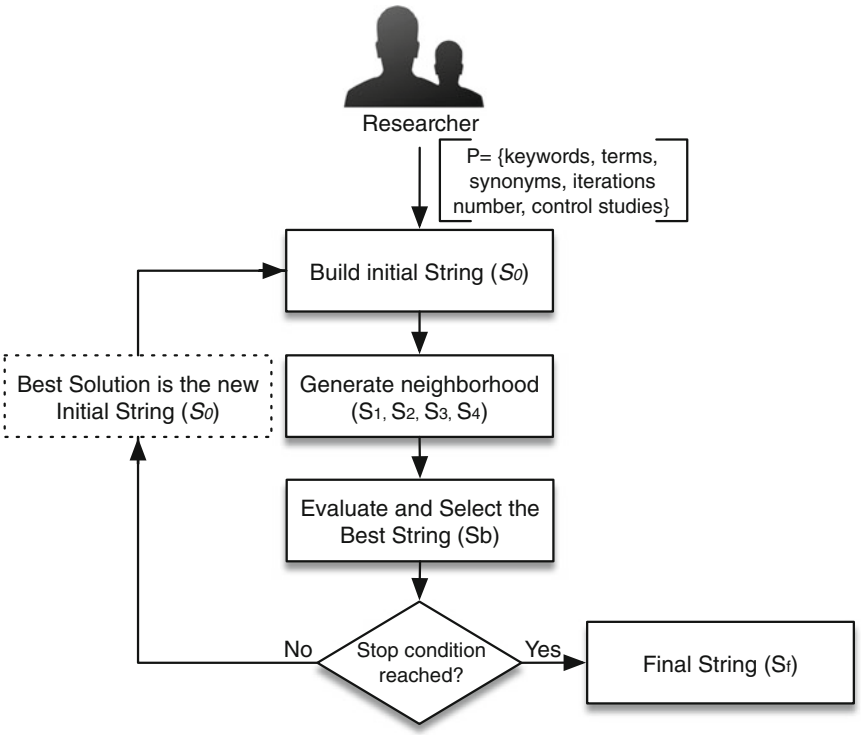

Fig. 104.1 SBSG's workflow

Fig. 104.2 PICO criterion (Kitchenham et al. [9])

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 104.1.

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 & 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 104.3 represents the measures aforementioned.

The *sensibility* on search strategy context is a measure to identify all of the relevant studies (**A** – Fig. 104.3) for a specific domain from retrieved studies (**R** – Fig. 104.3) supported by a set of relevant studies previously defined (**C** – Fig. 104.3). 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** – Fig. 104.3), where $\mathbf{B} = \mathbf{R} - \mathbf{A}$. 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 104.1 and 104.2, respectively.

$$S = \frac{A}{A + C} * 100. \tag{104.1}$$

$$P = \frac{A}{A + B} * 100. \tag{104.2}$$

The overall process by which a candidate string is evolved also provides an option to reduce researchers' efforts during the study selection stage. This alleviation of human efforts is provided enabling the assess to the quality of each study based on abstract, keywords, and control list. Then, researchers are able to eliminate those studies with zero or a very low fitness value. Therefore, the total fitness (**F**) is computed through Equation 104.3. When **F** is zero, the string returns only irrelevant studies and the higher **F** is, the higher the adequacy of search string will be.

$$F = \frac{(S) + (P)}{2} \quad (104.3)$$

Once no further improvements can be achieved for 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 & 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 104.2 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.

Table 104.1 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

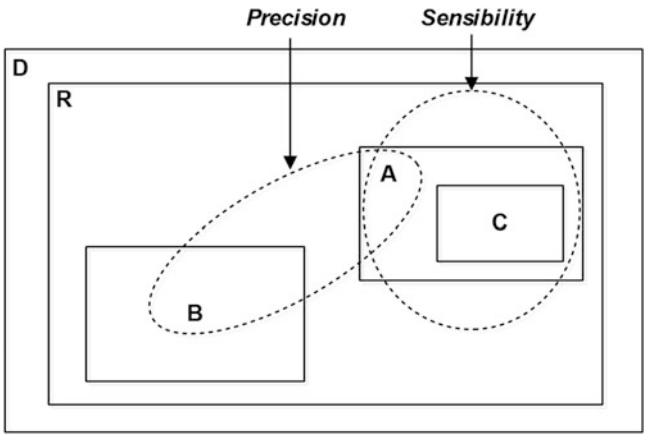

Fig. 104.3 Sensibility and precision search strategies

104.3.1 Proof-of-Concept Implementation

A tool to fully support SBSG was devised. It owns 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 pre-defined 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 real search strings.

The latter module contains a set of scripts devised in Python. 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.

104.4 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.

104.4.1 Research Questions

We designed our experiments in a proof-of-concept format aiming to answer the following *Research Questions* (RQ): (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

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

Table 104.2 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

approach?; and (iii) **RQ_3: In practice, how efficient is our approach to generate the best search strings?**

104.4.2 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.

104.4.3 Experimental Design and Execution

To provide empirical evidences to answer our RQs, we have used five SS's search strings that have already been published. During the selection of these SSs we have focused on covering a broad class of SS's search string from different field of research in SE. Summing up, have chosen SS's search string that aimed to identify primary studies of different contexts, such as software testing, software reusability, and model-driven development, etc.

Zhang et. al. [14] identified 11 digital scientific databases used more than once in SS for searching relevant studies in SE. Among them, IEEE Xplore is seen as the main digital source for SSs in SE. The content in IEEE Xplore comprises over 180 journals, over 1,400 conference proceedings, etc. Approximately 20,000 new documents are added to IEEE Xplore each month. Therefore, we have decided to use the IEEE Xplore digital source in our experiments. Furthermore, four reasons contribute 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, selected five set of terms, keywords and control list. Secondly, we performed the search with 5, 15, and 30 iterations. Then we performed 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) repeat 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. We explored the features of descriptive statistics provided by IBM SPSS Statistics tool.[3] They summarize data using four numbers: the mean; minimum and maximum values; and standard deviation. Section 104.5 provides the results collected from our proof-of-concept validation.

Table 104.3 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

104.5 Results and Discussion

104.5.1 Sensibility and Precision

First, we performed the descriptive statistics for Sensibility (RQ_1) of the generated search strings using our approach. Furthermore, 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 there were no increases in the sensitivity's mean. These results indicate that search strings with optimum sensibility can be generated through of the first iterations (Table 104.3).

Figure 104.4 shows that, for a small number of iterations the approach obtained a high sensibility for all cases, it means that a string can identify the most of relevant studies. In terms of SLR and SLM, a high sensibility is usually more desired. Therefore, the higher it is, more studies related to the domain must be returned in the search.

We also performed the descriptive statistics for precision (RQ_2) according to each iteration ($i = 5$, $i = 15$ and $i = 30$). Most of search strings had good results for five iterations (see Table 104.4). String 1 and String 2 achieved an optimum precision (20%, 21%, respectively), while the String 3 and String 5 had very high precision (41% and 35%, respectively). On the other hand, String 2 reached a low precision (12%). We noticed from the fifth iteration there were no increases in the mean of precision. Therefore, our approach is able to generate through of the first search strings with optimum and high precision.

The precision value for optimal strings must be low compared with the sensibility. Figure 104.5 shows a low value for all cases, this measures gives a proportion of relevant studies retrieved regarding to the number of

[3] see: http://www-01.ibm.com/software/analytics/spss/

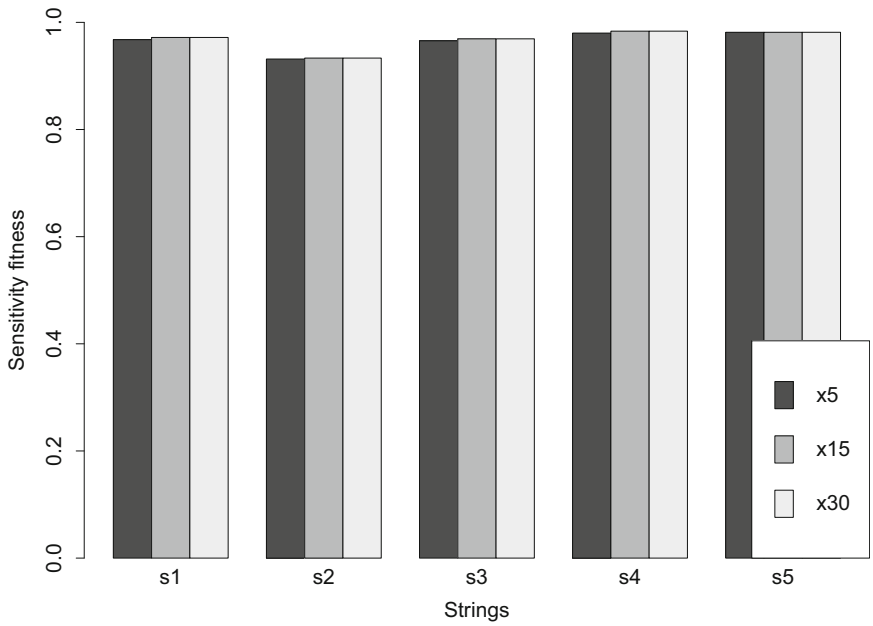

Fig. 104.4 The best sensitivity by iteration

Table 104.4 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

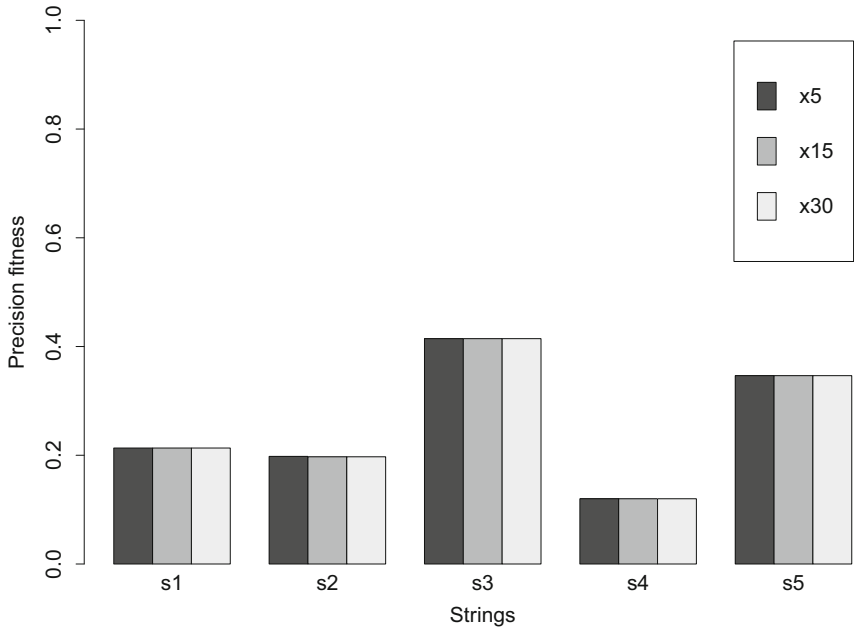

Fig. 104.5 The best precision by iteration

irrelevant studies from the search. Therefore, to our approach these results can be considered satisfactory, since that all strings achieve good precision in a few iterations.

Additionally, we can notice that in the Figs. 104.4 and 104.5 have been obtained similar results. We believe that it occurs due to the initial string be structured to cover the domain as a whole, thus, is almost impossible to start with an initial string with a low fitness. Assuming a good initial strings, the HC needs too less efforts and improvements to find the best one. Therefore, based on this assumption the results tend to be similar from a number of iterations.

104.5.2 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 104.6 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.

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. We can notice in Fig. 104.7 that the time depends of some important factors, such as, size of string and complexity of it, but even for the worst cases the approach is faster than a string generated by hand. However, as already mentioned, the HC had a 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): they took on average 116 second. Often, when manually fine-tuning search strings, many changes are needed. Therefore, 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.

104.5.3 Threats to Validity

Conclusion validity: a possible threat associated to the conclusion of our study is intimately 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.

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

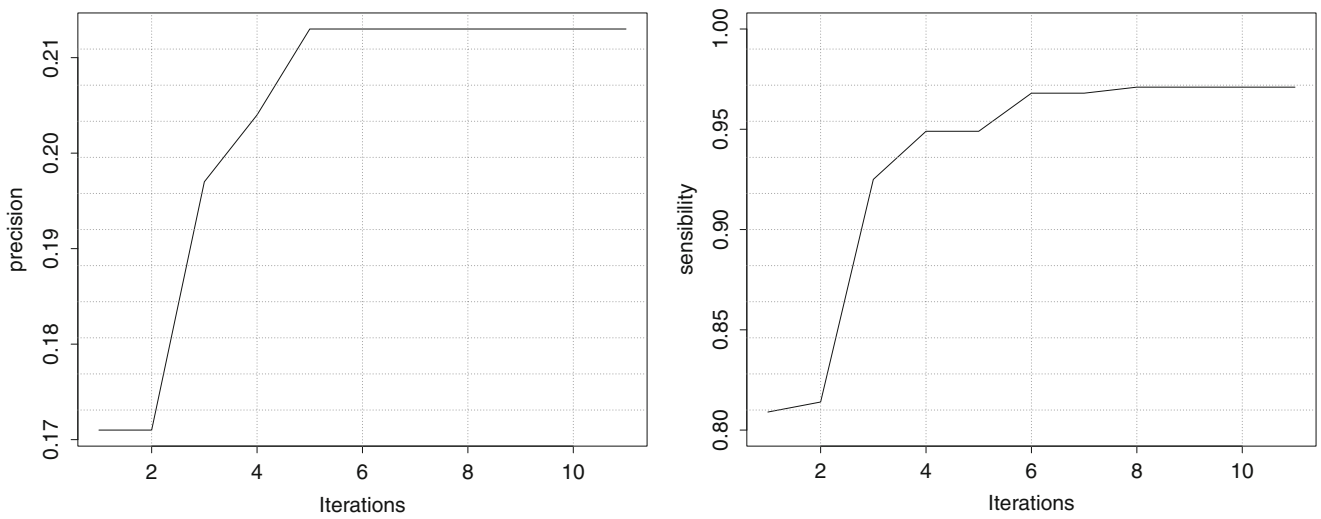

Fig. 104.6 Evolution of the sensibility and precision

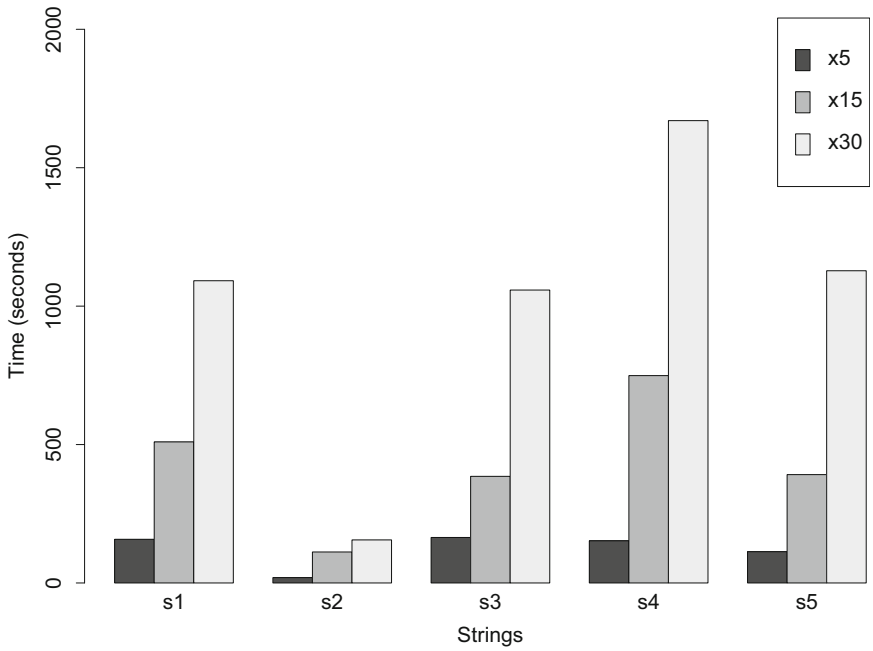

Fig. 104.7 Time to generate the search strings

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 the unknown drawbacks on using SBSG would be better observed with subjects conducting their systematic reviews. Moreover, we believe that the fact the we already known the SBSG method could contribute to better results.

104.6 Conclusion and Future Work

In this paper we propose the SBSG approach that applies a search-based algorithm called Hill Climbing to automate thr 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 previously known, SBSG automatically provides a string in the appropriate format according to a pre-determined digital database.

We also carried out an experiment using five real SSs that have already been published. Our study consisted of using SBSG to find search strings with good accuracy in accordance to measures of sensibility, precision, and efficiency. We considered the results satisfactory, once the strings were automatically generated/calibrated and the primary studies identified contained few losses compared to the actual results of the five real SSs. The results shown that the precision and the sensibility our approach are 25,2% and 96,2%, respectively. We believe SBSG is a contribution once it represents a first step to automatically support the generation of search strings for systematic reviews.

References

1. Basili, V., Caldiera, G., & Rombach, H. (1994). *The goal question metric paradigm* (1st ed.). Wiley.
2. Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys, 35*(3), 268–308.
3. Dieste, O., & Padua, A. (2007). Developing search strategies for detecting relevant experiments for systematic reviews. In *ESEM 2007*, Madrid (pp. 215–224).
4. Gay, G. (2010). A baseline method for search-based software engineering. In *PROMISE 2010*, PROMISE '10, Timisoara (pp. 2:1–2:11). New York, NY: ACM.
5. Hannay, J. E., Dybå, T., Arisholm, E., & Sjøberg, D. I. K. (2009). The effectiveness of pair programming: A meta-analysis. *Information and Software Technology, 51*(7), 1110–1122.
6. Harman, M., McMinn, P., de Souza, J. T., & Yoo, S. (2012). Search based software engineering: Techniques, taxonomy, tutorial. *Empirical software engineering and verification* (pp. 1–59). Berlin/Heidelberg: Springer.
7. Haynes, R. B., Wilczynski, K. A., McKibbon, C. J., & Sinclair, J. C. (1994). Developing optimal search strategies for detecting clinically sound studies in medline. *Journal of the American Medical Informatics Association, 1*, 447–458.
8. Kitchenham, B. (2011). Chapter three – What we can learn from systematic reviews. In *Making software what really works, and why we believe it* (Vol. 1, 1st ed.). Gravenstein Highway North, Sebastopol, CA: O'Reilly Media
9. Kitchenham, B., Mendes, E., & Travassos, G. H. (2007). A systematic review of cross vs. within-company cost estimation studies. *IEEE Transactions on Software Engineering, 33*(5), 361–329.
10. Kitchenham, B. A., Budgen, D., & Pearl Brereton, O. (2011). Using mapping studies as the basis for further research – A participant-

observer case study. *Information and Software Technology, 53*(6), 638–651.
11. Kitchenham, B. A., Dyba, T., & Jorgensen, M. (2004). Evidence-based software engineering. In *ICSE 2004*, ICSE '04, Edinburgh (pp. 273–281). Washington, DC: IEEE Computer Society.
12. Russell, S. J., & Norvig, P. (2003). *Artificial intelligence: A modern approach* (2nd ed.). Upper Saddle River, N.J: Pearson Education.
13. Straus, S., & Richardson, W. (2010). *Evidence-based medicine: How to practice and teach it* (4th ed.). Edinburgh: Churchill Livingstone.
14. Zhang, H., Babar, M. A., & Tell, P. (2011). Identifying relevant studies in software engineering. *Information and Software Technology, 53*(6), 625–637.