

Analyzing Citation Frequencies of Leading Software Engineering Scholars

Oliver Hummel¹, Alexander Gerhart¹ & Bernhard Schäfer¹

¹ Software Engineering Group, University of Mannheim, Mannheim, Germany

Correspondence: Oliver Hummel, Software Engineering Group, University of Mannheim, Mannheim 68131, Germany. Tel: 49-621-181-3911. E-mail: hummel@informatik.uni-mannheim.de

Received: July 19, 2012 Accepted: October 17, 2012 Online Published: November 15, 2012

doi:10.5539/cis.v6n1p1

URL: <http://dx.doi.org/10.5539/cis.v6n1p1>

Abstract

It is understandable that sponsors of research activities are interested in assessing the work of scholars they (plan to) support although this is not a simple undertaking. Until today, there is no obvious approach for objectively measuring and comparing the quality of research results from different disciplines. Hence, counting publications has been used for long time as a substitute to deal with this challenge and only recent technological advances have fostered the usage of so-called citation indices (such as the h-index) for this purpose. Although this approach is as disputed as all previous ideas in this context, we feel it is about time to investigate the expressiveness of modern citation analysis approaches in computer science more closely. In order to do that, we have chosen the area of software engineering and created a first comprehensive ranking, illustrating citation values of world class scholars by analyzing the work of almost 700 researchers in this field. We have found that top h-index scores in software engineering are around 60 while top-notch g-indices start at around 130 when Google Scholar and Publish or Perish, the quasi standard tools for this purpose are used. Clearly, the results of our study are influenced by the coverage of these tools so that we have also analyzed Google Scholar and found it having a very high coverage of software engineering publications. Hence we are convinced to have collected good quality results that will allow our community to better judge and use citation numbers in the future.

Keywords: citations analysis, software engineering, h-index, g-index, research quality assessment

1. Introduction

Evaluating the versatile work of a scientist is not a simple undertaking. Research, teaching and administrative duties are the three main pillars that consume for example the time of a typical university professor. Due to the limited amount of resources in our society it is understandable that sponsors of scientists try to assess the quality of scholars they plan to hire or to support otherwise. However, condensing the professional life of a scientist into a single numeric quality indicator is probably as difficult as describing the quality of a software system in the same way. Nevertheless, a simple idea to better deal with this challenge is assessing the three areas mentioned before individually; since research is usually considered the most important aspect in appointment procedures, approaches for evaluating the quality of research achievements have a long tradition.

However, the quality or more generally the value of research and research results is not easy to measure either, because it is depending on a multitude of factors such as its complexity, innovational strength or the impact of published results to mention just a few. Even more, most of these factors are rather “soft” factors and as such difficult to assess objectively. Therefore, the evaluation of publications is so far the only approach found practical enough to evaluate the research performance of scientists. Consequently, a number of prominent works in software engineering and computer science have tried to address this issue by counting publications in order to identify the most productive researchers of a given time period (Wong et al., 2011; Ren & Taylor 2007). However, due to their limitation on a small number of examined publication venues (around seven) the results of these studies are certainly interesting, but not genuinely representative as they ignore a large quantity of publications. Furthermore, by definition it is not possible to detect the best researchers in a field by merely investigating a limited period in time. Although getting results published in peer reviewed journals or conferences of high quality is clearly a challenge in itself (sometimes even more than it should be as nicely summarized by Santini (2005)), the mere acceptance of an article does not guarantee that its content will turn out as being highly relevant for a research community in the future. Thus, counting the references (i.e. citations) to a

publication in later publications seems to be a better way to assess the long-term impact of scientific work and to identify researchers with large influence on a field. After all, this so-called citation analysis is also not a new idea since it has already been proposed by Garfield in the mid 1950s (Garfield, 1955). However, due to technical limitations, it has been largely constrained on assessing the impact of journals until very recently. The emergence of and public awareness for citation indices of individual researchers (such as the h-index proposed by Hirsch (2005)) was only possible on the basis of technological advances exploited by a new breed of search engines for scientific publications (such as Google Scholar). Obviously, numerous citations of a publication do still not indicate high quality research per se (as discussed by e.g. Merton, 1968; Parnas, 2007; Meyer et al., 2009), it appears that citation analysis is a clear and widely recognized advancement in this field. We hence believe that it is about time to create a first comprehensive citation benchmark for software engineering scholars that allows comparing their citations indices within the field and with those of colleagues from other (computing) disciplines as e.g. listed on the Web by Palsberg (2012).

After briefly describing important foundations of citation analysis in the following section 2, we outline the goals of our study in more detail in section 3. Section 4 presents how we approached the challenge of collecting and sampling several hundred internationally renowned software engineering researchers with the help of de-facto standard. Subsequently, section 5 discusses the results of our benchmarking effort for almost 700 software engineering researchers. Furthermore, it presents interesting insights from a comprehensive coverage analysis of Google Scholar, conducted for papers authored by a representative sample comprising 20 authors from the set of 700, underlining the applicability of Google Scholar (and therewith Publish or Perish) for this purpose. An outlook on potential future work and some concluding remarks finally round off our contribution in section 6.

2. Foundations

Although citation analysis has been successfully applied for more than five decades for deriving the impact factor of journals (Garfield, 1955) it has gained the attention of a larger audience only recently after the so-called h-index has been proposed by Hirsch (2005) for assessing the publication strength of individual researchers. Powerful scientific search engines that have become publicly available are certainly another factor explaining this new trend since they easily allow individual researchers to compare their citation indices with other scholars. Given the list of publications of a researcher, sorted in descending order of citation frequency, the h-index is defined as follows: “A scientist has index h if h of his/her N_p papers have at least h citations each”. The following table illustrates this with the help of citation numbers for Barry Boehm’s publication record retrieved with Publish or Perish (Harzing, 2010) in November 2011.

Table 1. Exemplary calculation of Barry Boehm’s h-index

Rank	Citations
1	6,319
2	3,684
...	...
52	54
53	53
54	52
...	...

Accordingly, as of November 2011 his h-index was 53. Assuming that correct data is used, the h-index can apparently only grow over time when additional references to an author’s publications are published somewhere. In common interpretations of the h-index self-citations that occur whenever authors cite their own previous work are counted as well and it also seems to be common understanding that sustainably increasing one’s h-index by self-citations is difficult if not impossible as e.g. discussed by Engqvist and Frommen (2008). Another characteristic of the h-index and a more common reason for criticizing it is that groundbreaking publications (with potentially thousands of citations) are not valued sufficiently as in principle 53 publications with exactly 53 citations each would suffice to reach the same h-index as Barry Boehm above. Consequently, relatively soon after the h-index was presented, the so-called g-index was proposed by Egghe (2006). It tries to mitigate this disadvantage by summing up the citations and comparing this number to the squared rank of the publications as shown in the following table, again for the publication record of Barry Boehm in November 2011.

Table 2. Exemplary calculation of Barry Boehm's g-index

Rank	Rank ²	Citations	Sum
1	1	6,319	6,319
2	4	3,684	10,003
...
156	24,336	11	24,886
157	24,649	11	24,897
158	24,964	11	24,908
...

Since on rank 157 the sum of all citations is for the last time larger than the squared rank, Barry Boehm's g-index is 157. Given the results of other SE scholars presented later, this is an exceptionally large number that actually seems to represent his groundbreaking work in various areas of software engineering far better than his doubtlessly strong but not particularly exceptional h-index of 53.

Other recently presented citation indices include Jin's AR-index, taking the age of h-indexed publication into account, as well as its predecessors the so-called A-index and the R-index (Jin et al., 2007). With Claes Wohlin's w-index (Wohlin, 2009) there even exists a proposal for another citation index from a prominent software engineering researcher. However, since most of these indices can be calculated from the original h-index with only slight variations and have not nearly gained the popularity of the g- and the h-index, we will focus on the latter two as the most prominent citation indices for the remainder of this article.

2.1 Common Criticisms

Beyond the fundamental question whether research can be judged by merely counting publications or citations (Parnas, 2007), there are a number of further criticisms that briefly need to be raised in this subsection in order to illustrate the limitations of citation indices. First of all, it is clear that a plain citation does not indicate whether it was used to refer to a valuable contribution or because it is seen as a "bad example". Furthermore, it seems that well known researchers get cited more often, a fact that is known as the "Matthew Effect" (Merton, 1968) in literature. In addition, as already recognized by Hirsch in his seminal publication, citation numbers are not normalized in regard to the number of authors that may have co-authored a publication (Hirsch, 2005). However, the impact of normalizing a publication's weight (i.e. counting it only with e.g. 1/3 if three authors have contributed) is unclear anyhow, since at least within the same community author teams seem to have similar sizes (as also shown in section 5 of this article). Last, but not least, all publication- or citation-based evaluation approaches are prone to targeted manipulations such as "citation cartels", adding non-participating authors to papers or the publishing of "half-baked ideas" in non-reviewed publication venues (Parnas, 2007) to a certain degree.

2.2 Scientific Indices and Search Engines

There exist several different scientific indices and search engines for scholarly literature that include citation numbers. Some of them are freely available, such as Google Scholar, Scirus from Elsevier, CiteSeerX, Microsoft Academic Search or even the ACM's digital library. Among the well-known subscription-based search engines are Scopus from Elsevier and Thomson ISI's Web of Science (WoS). Prior to 2004, the latter held a quasi-monopoly on scientific research involving citation frequencies and only covered a selected number of mostly English-speaking journals (Nounzi, 2005). However, that comfortable position for Thomson Scientific changed in late 2004, when both Scopus and Google Scholar were introduced. Today, according to information from Thomson Scientific (Thomson, 2012), WoS covers about 12,000 journals in 256 categories starting from the year 1900 and over 148,000 proceedings of conferences, workshops, symposia, seminars, colloquia and conventions. According to information on SciVerse's Scopus website (Elsevier, 2012), they provide the largest abstract and citation database of peer-reviewed literature worldwide. It includes 46 million records, 70% of them including abstracts, nearly 19,500 titles from 5,000 publishers worldwide, and more than 4.6 million conference papers. Google Scholar also covers all previously mentioned ways of publishing (Falagas et al., 2007), but unfortunately does not provide any information regarding the numbers of covered journals, proceedings etc. (Jasco, 2008). According to a recently published study by Wainer et al. (2011), however, the WoS indexing service is missing about 66% of published works in computer science, while Scopus on average only misses 33%.

According to various previous studies (which we will discuss in section 4.2) and our own investigations presented later, Google Scholar has by far a higher coverage and thus has become the search engine of choice for conducting this study as described in the following section. Furthermore, in contrast to similar offerings such as Scopus, Google Scholar is free of charge so that it is easy to replicate our results, has already been used by various other works and last but not least it is used by Publish or Perish (Harzing, 2010), the de-facto standard tool when it comes to citation analysis.

2.3 Goal of This Study

The desire of sponsors of research work to objectively assess the performance of researchers is all too understandable, not only in times of financial crises and budget cuts. As of today, however, most approaches proposed for this purpose are highly disputed amongst researchers and also citation analysis based on the recently introduced citation indices is no exception to this debate. However, given the increasing public attention for citation indices, it is certainly helpful (and necessary) to gain a better understanding of the mechanisms at work and their specific impact on the computer science community. The “numbers game”, as Parnas (2007) has strikingly phrased it, is perhaps not (yet) played very much in our area, but has at least gained so much attention that the h-index has already become part of numerous scientific CVs all over the world.

Even worse from a software engineering (SE) perspective is the fact that the h-index seems to be extremely community specific as already a shallow look on Palsberg’s well-known list (Palsberg, 2012) of computer scientists reveals: top-notch researchers achieve an h-index of over 100 there, while no software engineer can be found amongst its enlarged top. A devil’s advocate might claim that other disciplines achieving higher values are merely more mature or even more important than software engineering, our feeling, however, is that one should not compare apples and oranges as customs in different communities are varying too much. Nevertheless, beyond Wohlin’s brief “en passant” analysis from 2009 that lists the top-notch SE researchers with an h-index of around 30 we are not aware of any investigation of citation indices that could be used for comparisons with other research areas (Wohlin, 2009). Thus, we believe it is about time to create comprehensive citation benchmarks for all important communities that allow a better comparison of researchers within a community and even beyond community boundaries. Obviously, this is a non-trivial effort so that we decided to start with a pilot study for the field of software engineering that can be used as a template for similar efforts in other communities in the future.

The comparability (and competitiveness) of computer science with many other disciplines also suffers from another difference: while journal publications are considered being of highest importance in most other fields, computer science is usually different: due to the fast pace of technological advancement, a large part of the academic work there is not published in journals, but in conference proceedings instead. Consider the publication record of Gail Murphy as an example: out of the 112 publications listed on her homepage, “merely” 22 were published in a journal. Out of the 44 publications that are counted for her h-index, and thus can be seen as her most important publications (see Table 4), 24 were actually presented during a conference and published in conference proceedings. Therefore, to grasp as much of the work of software engineering researchers as possible, purely counting journal publications, as for example carried out in older versions of a researcher rankings initiated by Robert Glass (Wong et al., 2011), obviously rules out a large number of important publications. Although it is clear that such a manual ranking must constrain the number of publication venues somehow (and choosing the most prominent ones is certainly a logical solution) this also misses a large body of relevant work that has been not published in high-profile venues due to negative peer reviews. Although, in general, peer reviews seem to be an effective strategy for assuring the quality of submitted work they sometimes seem to be too strict when it comes to innovative new ideas, as illustrated by Santini’s entertaining collection of initially rejected works (Santini, 2005). Actually, some of the most prominent ideas in computer science (such as Dijkstra’s famous article on the goto statement (Dijkstra, 1968)) were initially rejected and had to find different ways for publication. Such reports are certainly thought-provoking and underline the necessity of having a performance measure that also takes publications coming from the outside of often relatively closed communities of prestigious journals and conferences into account. More details on possible issues with peer reviews are e.g. discussed by Birukou et al. (2011) that also propose potential countermeasures, such as ranking papers as opposed to merely reviewing them independently in conference program committees.

Given these aspects, we believe that the expressiveness of citation based measures is clearly superior to mere publication counting and define the creation of a comprehensive collection of citation performances for software engineering researchers as the central goal of the study presented in this article. Accordingly, the central research question (RQ1) is: What are the highest g- and the h-indices reachable by leading software engineering scholars based on de-facto standard Publish or Perish (that is using Google Scholar)? Furthermore, we want to know: How high is the coverage of publications in Google Scholar for a representative sample of software engineering

researchers (RQ2) and what percentage of their publications has appeared in journals (RQ3)? And finally we are interested in finding out whether the coverage of Google Scholar is better for newer publications (RQ4).

3. Study Design

Any study aiming on the reproducible creation of a citation benchmark is confronted with three fundamental questions, namely –

- exactly defining the research area to be analyzed
- collecting a list of researchers for the study
- obtaining comprehensive and reliable citation data

Although all three challenges sound relatively simple at a first glance, yet a number of subtle details, which we are briefly discussing in the following subsections, need to be considered.

3.1 Definition of Software Engineering

The literature contains a multitude of definitions for software engineering and most of them are similar to the following one taken from the IEEE (2004): software engineering is “(1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software. (2) The study of approaches as in (1)”. However, in practice, software engineering has many overlaps with neighboring disciplines so that it will probably always be impossible to filter out publications that are not genuinely software engineering related since it is impossible to define a sharp boundary between disciplines.

3.2 Selection of Researchers

The same constraints certainly apply for the selection of researchers themselves, since scholars are often working in several different fields or in areas at the boundary between two fields so that it will probably remain impossible to find researchers that work in software engineering only. In order to build a first representative set of renowned researchers for our analysis despite these concerns, we decided to collect the members of program and organization committees (PC resp. OC) of three leading software engineering conferences in the last ten years (i.e. 2002–2011): the International Conference on Software Engineering (ICSE), the International Conference on Object-Oriented Programming, Systems, Languages & Applications (OOPSLA resp. SPLASH), and the joint meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE) have been selected for this purpose. Unfortunately, not all data was available on the Internet. Only for ICSE both lists were accessible for all conferences in the desired timeframe. For OOPSLA, all PC and OC members from 2002 to 2011 were available at the time of conducting our study, except for the 2003 and 2007 OC lists that were missing. FSE PC members were available from 2003 to 2011 with 2002 missing and FSE OC members were available since 2004, with 2002 and 2003 missing. The overall amount of researchers collected is shown in Table 3.

Table 3. Overview of amount of committee members investigated per conference

Conference	Program Committee	Organization Committee	PC + OC unified and cleared of duplicates
ICSE	430	319	317
OOPSLA	290	206	294
FSE	254	124	190
Total	974	649	670

As illustrated in the table, the total number of individuals who were member of at least one PC or OC of one of the three selected software engineering conferences over the last decade is 670. It is reasonable to assume that this set comprises a large proportion of the main contributors to the world’s software engineering research in this timeframe, although it still does not solve the challenge of deciding whether someone is a “genuine” software engineering researcher. It is also obvious that e.g. taking authors of these conferences and perhaps of leading software engineering journals into account would have further increased the coverage of researchers. However, it would have also increased the workload for this study so that we decided to postpone this to future work.

In order to also cover renowned software engineering scholars that have not served in one of these conferences

within the last ten years anymore (such as Barry Boehm, for example), we decided to collect data from an additional list, namely from the list of all ACM Sigsoft research award winners (ACM, 2011). As of November 2011, 21 individuals have received this award out of which 11 were not contained in the “conference list” from before so that we eventually yielded a sample of 681 researchers to investigate.

3.3 Search Engines and Tools

As hinted before, the number of search engines reasonably usable for a citation analysis remains quite small. Traditionally, this was the domain of specialized citation databases (such as Scopus or Web of Science) that, however, as latest studies (Wainer et al., 2011) reveal, are rather incomplete from a computer science perspective. After all, the emergence of Google Scholar seems to have changed the premises considerably, as it, according to further previous studies (such as by Bosman, 2006) provides a reasonable coverage for computer science publications. Moreover, Harzing’s well-known Publish or Perish tool (PoP, (Harzing, 2010)) that can be seen as a quasi-standard in citation analysis, also uses Google Scholar for collecting its data and can be seen as a graphical user interface for Google Scholar. PoP is free for personal non-profit use and as of 2012 is downloadable from the website of its creator Anne-Wil Harzing, which can be found under www.harzing.com. Just like Google Scholar it offers the possibility to include resp. exclude names from searches, or to limit timeframe and/or subject area of the desired search results.

Unfortunately, Google is very restrictive in disclosing even basic information on the mechanics and the contents of Google Scholar. Although the ranking of Google Scholar can be fully ignored for the calculation of the g- and the h-index, it is indeed a serious problem that Google does neither reveal which publications are indexed nor that it discloses any assessments of the quality of its data. Completeness and correctness of publications and their citations are of course a fundamental prerequisite for citation analysis and hence several third-party studies examined the quality of Google Scholar in this regard. In a study conducted by Walters (2007), Google Scholar outperformed seven other databases by covering 93 % from a sample of 155 articles. Bar-Ilan (2008) was able to find all but one article she used for a computer science literature review with Google Scholar. Moreover, she compared the h-indices of a group of 40 highly cited researchers from Israel based on Google Scholar, Scopus and World of Science for publications released between 1996 and 2006. Ten of these 40 researchers are computer scientists. Only for one of those ten computer scientists, the h-index calculated from Google Scholar was lower than the h-indices from either Web of Science or Scopus. According to Chen (2010), Google Scholar covered 98 to 100 % from eight databases of both subscription based and publicly available journals in 2010. Five years earlier, the coverage of the same databases was only between 30 and 88 % while Meier and Conkling (2008) found a coverage of over 90 % in engineering literature.

Thus, despite a number of flaws still identified in recent studies (Jasco, 2010), for the time being, Google seems to be the best search engine available for the purpose of citation counting and hence was used for obtaining the results presented in the following.

4. Results

4.1 RQ1: Software Engineering Citation Benchmark

In order to gain a better overview of the individual research performance of the 681 individuals in our unified list, we developed a little program that allowed querying Google Scholar automatically for an initial ranking. With the help of this tool, it was possible to determine h-index, g-index, total number of publications and total number of citations for each researcher on the list.

Once this was accomplished, we took the 50 entries with the highest h-indices from this list (actually due to identical h-indices we used 51) and analyzed them manually with Publish or Perish in order to allow direct comparability with this quasi standard. The results presented in this section have been acquired in November 2011, with PoP under the following prerequisites: We have used the full names of the authors as listed on the conference websites (i.e. firstname middle initial lastname) and limited queries to the category Engineering, Computer Science and Mathematics. As a further broad sanity check, all queries were restrained to the time period between 1940 and 2012.

Based on this effort, the following table presents the 26 most-cited software engineering researchers from the overall collection comprising 681 individuals, sorted by their h-index as determined by Publish or Perish. Furthermore the table contains the respective affiliations, the g-index, the total numbers of publications and citations, and finally the average number of authors per paper for each researcher.

Table 4. Benchmark for citation frequencies of leading software engineering scholars

Pos.	Researcher	Affiliation	h-Ind.	g-Ind.	Publ.	Cit.	Auth./Pap.
1	Victor Basili	U Maryland	65	134	420	19,323	2.69
2	Douglas C. Schmidt	Vanderbilt U	63	143	751	23,498	2.83
3	Luca Cardelli	Microsoft Res.	63	133	255	17,928	2.09
4	John Mylopoulos	U Toronto	63	118	519	16,104	3.20
5	Elisa Bertino	Purdue U	63	109	810	16,172	3.15
6	John C. Mitchell	Stanford	58	107	198	11,766	2.78
7	<i>David Walker</i>	<i>Princeton</i>	<i>55</i>	<i>123</i>	<i>848</i>	<i>17,331</i>	<i>2.85</i>
8	David Harel	Weizmann Inst.	54	156	272	24,642	2.26
9	David Garlan	Carnegie Mellon	54	126	336	16,486	2.81
10	Jeff Kramer	Imperial College	54	113	264	13,243	2.80
11	Barry Boehm	U South. Calif.	53	157	480	25,802	2.61
12	Mary Jean Harrold	Georgia Tech	53	93	191	9,098	2.60
13	Lionel Briand	U Luxembourg	50	99	230	10,386	3.00
14	Rachid Guerraoui	MIT	50	94	457	10,772	2.85
15	Benjamin Pierce	U Pennsylvania	49	101	243	10,614	2.33
16	James Larus	Microsoft Res.	49	99	235	10,129	3.07
17	Craig Chambers	U Washington	49	96	171	9,341	2.81
18	Roy Campbell	U Illinois	49	95	549	11,065	3.08
19	Don Batory	UT Austin	48	88	266	8,503	2.50
20	David Parnas	U Limerick	46	122	246	15,033	1.86
21	Somesh Jha	U Wisconsin	46	94	188	9,073	3.28
22	Gregg Rothermel	U Nebraska	46	84	189	7,338	3.11
23	Gail Murphy	U Br. Columbia	44	76	159	6,050	2.53
24	<i>David Evans</i>	<i>U Virginia</i>	<i>43</i>	<i>83</i>	<i>700</i>	<i>9,215</i>	<i>2.64</i>
25	Kathryn McKinley	UT Austin	43	78	202	6,526	3.04
26	Gail Kaiser	Columbia U	43	72	333	6,348	2.61

As visible in the table, an h-index of more than 42 was required to become part of this list. However, as a manual sanity check has revealed, the names of two researchers (printed in red and italics) are so common that their results are highly questionable since Google Scholar does not seem to be able to differentiate them from colleagues with identical first and last names.

In general, the table reveals a clear dominance of North American researchers, which is not a surprise since the three selected conferences (and the ACM of course) regularly have a clear majority of their participants and (PC) members from the US. It nevertheless underlines that the best reachable “lifetime” h-indices in software engineering currently are around 60 with Victor Basili achieving the highest value of 65. While he and Douglas Schmidt on the second position can certainly be seen as genuine software engineering researchers, the case is not as clear for Luca Cardelli, John Mylopoulos and Elisa Bertino. They have at least clear influences from the theoretical computer science resp. database communities and published a significant number of papers there as well. The highest g-index contained in our list is the one of Barry Boehm with 157 while David Harel is almost head to head with 156. These results clearly confirm that the g-index better values groundbreaking publications than the h-index where Boehm and Harel are still strong but not exceptional. Another interesting aspect is the number of authors that on average participated in the publications as it rarely exceeds 3 and supports the assumption that successful researchers normally do not publish in larger author teams (what would probably merely increase the publication count).

It is also interesting to see the extremely large publication numbers that are attributed to some of the researchers by Google Scholar. Although the biography of Elisa Bertino confirms more than 500 reviewed papers and articles with her name on it, it seems likely that at least some of her 810 publications found by Google Scholar are caused by parsing errors or duplicates. However, since those normally do not get cited in other publications we expect the impact on citation numbers to be widely negligible. Beholding this from another perspective, one might also conclude that Google Scholar is indeed not very reliable in terms of absolute publication numbers (and hence rankings based purely on automated publication counting are practically worthless), but the usage of citations numbers for calculating citation indices, seems to provide an effective means for filtering the “noise” created by ghost publications and other indexing errors.

Microsoft has recently been trying to catch up with Google Scholar with its Academic Search engine (MSAS) that amongst various other features offers an automatic recognition of research interests and a calculation of citation indices limited to the specific field. The first five authors listed there for the field of software engineering in November 2011 were as follows (values are for software engineering only, the values in brackets indicate the values measured over all fields).

Table 5. Researcher sample taken from Microsoft academic search

Pos.	Researcher	Affiliation	h-Ind.	g-Ind.	Publ.
1	Victor Basili	U Maryland	54 (56)	104 (104)	300 (108)
2	Edmund M. Clarke	Carnegie Mellon	53 (70)	155 (171)	165 (401)
3	Barry Boehm	U South. Calif.	47 (47)	113 (113)	290 (471)
4	Marry Jean Harrold	Georgia Tech.	47 (48)	76 (77)	183 (218)
5	David Garlan	Carnegie Mellon	44 (48)	116 (120)	196 (297)

It is interesting to see that the values from MSAS are all significantly smaller than the ones provided by Google Scholar. Since information on the coverage or the algorithms used by Microsoft is also not available it can only be speculated that the coverage of MSAS is currently lower than that of Google Scholar. It is also interesting to see that there is a significant difference between filtered and unfiltered publication numbers, while this difference does not seem to influence citation indices in most cases.

4.2 RQ2: Google Scholar Coverage Analysis

Although we believe that the ranking just presented is already an interesting contribution on its own and can easily be used for a “self-assessment” of interested software engineering researchers not in our list, for instance, we were also interested in the quality of the numbers ascertained. In order to get an idea of the coverage of Google Scholar for software engineering publications, we performed a coverage analysis based on the data we have collected. For that purpose, we created two different sample groups containing ten researchers each. “Group 1” contains nine high-profile researchers from the “conference list” and one researcher from the “Sigsoft award list”. “Group 2” contains ten researchers with h-indices between 11 and 9 again taken from the “conference list”. All names were chosen randomly, but only those researchers who had a comprehensive list of publications on their homepages were finally considered. The following procedure was then performed for each researcher in both sample groups:

1. The researcher’s h-index and g-index was taken from Publish or Perish.
2. The total numbers of publications were taken from a) Google Scholar (search term: author: “author name”; all subject areas), b) Publish or Perish (search term “author name”; subject area “Engineering, Computer Science, Mathematics”) and c) the researchers homepage.
3. The results of GS were compared to the publication list on the researcher’s homepage.
4. The h-index of each researcher was re-calculated, this time based on the author’s publications list for all publications for which citation numbers were found within Google Scholar.
5. In the last step, Google Scholar’s coverage for every researcher was calculated, based on her/his publication list.

Publication types manually excluded from this analysis were the following: Editorships of books and proceedings, non-scientific publications, publications that are explicitly listed as unpublished and publications in languages other than English. The results of this effort are shown in the following table.

Table 6. Coverage analysis for Google Scholar (GS) in comparison to Publish or Perish (PoP) and author's homepages

Name	Publ. GS	Publ. PoP	Publ. WWW	h-Ind. PoP	h-Ind. WWW	Cov. GS
Mary Jean Harrold	216	191	134	53	53	97.76
Gregg Rothermel	237	189	129	46	47	98.45
Gail Murphy	289	159	112	44	44	99.11
Kathryn McKinley	258	202	123	43	42	100.00
Martin Rinard	301	261	150	41	42	98.00
Jeff Magee	220	196	112	41	37	93.75
Richard N. Taylor	204	184	151	41	42	89.40
Gerard Holzmann	199	162	128	39	41	88.28
Jens Palsberg	243	214	103	39	39	98.06
Bashar Nuseibeh	234	186	72	38	37	95.83
					Avg.	95.72
Amiram Yehudai	87	77	65	11	12	96.92
Yvonne Coady	118	95	45	12	6	82.22
Eda Marchetti	46	35	44	11	10	77.27
Eli Tilevich	73	64	49	11	11	97.96
Giovanni Denaro	55	44	34	11	11	97.06
Ali Mesbah	54	27	18	10	10	100.00
Adrian Kuhn	54	48	39	12	12	100.00
Julie McCann	95	66	63	10	10	95.24
Apostolos Zarras	64	57	50	10	11	90.00
Sebastian Burckhardt	43	38	22	10	8	90.91
					Avg.	92.54

The table reveals that GS and PoP (columns Publ. GS resp. Publ. PoP) indeed tend to attribute more publications to a researcher as have actually been published by him or her according to the respective homepage (column Publ. WWW). On the other hand the coverage of actually written papers by Google Scholar (Cov. GS) is in the range of clearly over 90 % with a slight advantage for the sample of highly successful (and usually experienced) researchers over mostly younger ones with a lower h-index.

We consider this being good and bad news at the same time, as it underlines that GS has further improved its coverage compared to earlier publications that reported lower numbers (Chen, 2010; Meier & Conkling, 2008). On the other hand, it also supports the provisos of other researchers (Jasco, 2010) against GS as it still seems to index a lot of “ghost publications”. However, as mentioned before, these publications are usually not cited anywhere so that they do not influence citation index values (presented in column h-Index PoP). Moreover, GS still seems to have issues with the quality of its parsing algorithm since a manual search for a paper listed on the author's homepage, but not within GS's results, finds them in practically all cases: they are just not part of the result list in GS or PoP. There are two plausible reasons for a publication being found by GS, but not part of the PoP search results: first, the publication is not correctly indexed in the subject area Engineering, Computer Science, Mathematics. For example, this was the case for the technical report “Aristotle: A system for research on and development of program-analysis-based tools” (Harrold & Rothermel, 1997) authored by Mary Jean Harrold and Gregg Rothermel; which explains that Rothermel's h-index based on his homepage is higher than the one calculated by PoP based on the GS search results. The second plausible reason is that Google Scholar does not know that a researcher in question is one of the authors of a publication. An example for this is a paper authored by Donglin Liang and Mary Jean Harrold (Liang & Harrold, 1999). Google Scholar included it in the search results for the search terms “Efficient Points-To Analysis for Whole-Program Analysis” author: “Donglin Liang”, but did not include it when “Efficient Points-To Analysis for Whole-Program Analysis” author: “Mary Jean Harrold” was searched. Therefore, this paper was neither included in the Publish or Perish search results, nor Harrold's h-index. However, its place in the h-index core was filled by another publication so that the actual value was not impacted.

We have found only two cases where the manually counted h-index was significantly lower than the automatically created version by GS/PoP namely Jeff Magee and Yvonne Coady. The reason for that was that the publication lists on their homepages are obviously not complete. Jeff Magee's list misses 11 publications that all have his name in the byline and have been cited more than 40 times. The same applies for Yvonne Coady, whose publication list misses 7 papers that were counted in her PoP-based h-index.

4.3 RQ3: Journal Coverage Analysis

Most previous studies that analyzed Google Scholar's coverage or compared it to results from Web of Science or Scopus focused on coverage of journal articles since they are the main publication venues in other scientific disciplines. Since our coverage analysis so far did not distinguish between journal papers and other publications, such as conference or workshop proceedings or even technical reports that are common in computer science, we present appropriately divided results in this subsection.

The numbers reveal that the selected software engineering researchers have only published about 20% of their total publications in journals, which underlines the relatively low importance of journal publications in the fast moving research in software engineering once more. Therewith, it once again highlights the limitations of previous studies that mostly focused on journal publications (cf. Wong et al., 2011). Note that publications presented at a conference or in a technical report and later published in a journal (or vice versa in the latter case), were counted in both categories for the purpose of our analysis.

Table 7. Coverage rate of GS for journal articles

Name	Publ. WWW	Publ. Jour.	% in Jour.	Miss.	% GS Jour. Cover.
Mary Jean Harrold	134	26	19.40	0	100.00
Gregg Rothermel	129	36	27.91	0	100.00
Gail Murphy	112	22	19.64	0	100.00
Kathryn McKinley	123	28	22.76	0	100.00
Martin Rinard	150	21	14.00	0	100.00
Jeff Magee	112	25	22.32	0	100.00
Richard N. Taylor	151	33	21.85	0	100.00
Gerard Holzmann	128	33	25.78	0	100.00
Jens Palsberg	103	42	40.78	0	100.00
Bashar Nuseibeh	72	20	27.78	0	100.00
Amiram Yehudai	65	23	35.38	0	100.00
Yvonne Coady	45	5	11.11	0	100.00
Eda Marchetti	44	5	11.36	1	80.00
Eli Tilevich	49	8	16.33	0	100.00
Giovanni Denaro	34	2	5.88	0	100.00
Ali Mesbah	18	3	16.67	0	100.00
Adrian Kuhn	39	4	10.26	0	100.00
Julie McCann	63	17	26.98	0	100.00
Apostolos Zarras	50	11	22.00	0	100.00
Sebastian Burckhardt	22	1	4.55	0	100.00
Total		365	20.14	1	99.73

Out of 365 journal publications found for this analysis as shown in Table 7, only one single paper was missing in Google Scholar. Thus, the results of this analysis show an impressive coverage for software engineering journal articles of almost 100 %.

4.4 RQ4: Age-based Coverage Analysis

In addition to this basic coverage analysis, all 1,643 publications found on author homepages were sorted by publication year and Google Scholar was searched for them. The results, including publications from 1978 until 2012, are shown in the following table.

Table 8. Publications missing in GS according per publication year

Year	Publ.	Miss.	Cov. %	Year	Publ.	Miss.	Cov. %
1978	3	1	66.7%	1996	76	6	92.1%
1979	5	0	100.0%	1997	63	5	92.1%
1980	6	1	83.3%	1998	81	3	96.3%
1981	9	1	88.9%	1999	71	4	94.4%
1982	7	1	85.7%	2000	70	2	97.1%
1983	13	5	61.5%	2001	86	6	93.0%
1984	17	2	88.2%	2002	75	1	98.7%
1985	5	1	80.0%	2003	92	4	95.7%
1986	10	3	70.0%	2004	101	1	99.0%
1987	8	1	87.5%	2005	106	4	96.2%
1988	15	1	93.3%	2006	93	2	97.8%
1989	16	2	87.5%	2007	97	5	94.8%
1990	18	1	94.4%	2008	98	6	93.9%
1991	22	2	90.9%	2009	80	1	98.8%
1992	40	1	97.5%	2010	77	0	100.0%
1993	47	5	89.4%	2011	59	1	98.3%
1994	44	3	93.2%	2012	2	1	50.0%
1995	43	2	95.3%	Sum	1,643	84	94.9%

While older papers from the 1980s still have a relatively high miss rate it continuously decreases from the 1990s onwards where practically all years are covered with at least 90%. After looking at the graphical representation of the data shown in Figure 1, it can be assumed that there still is an ongoing trend towards higher coverage rates for papers published in more recent years. To verify this assumption, the data from Table 9 was used to perform a linear regression analysis. The resulting regression line is also shown in Figure 1, it has a slope of: $y = 0.003564 * x - 6.212212$. Excel's CORREL function shows a correlation coefficient of $r = 0.319$. Removing the outlier result of upcoming publications in 2012 delivers a correlation coefficient of $r = 0.648$.

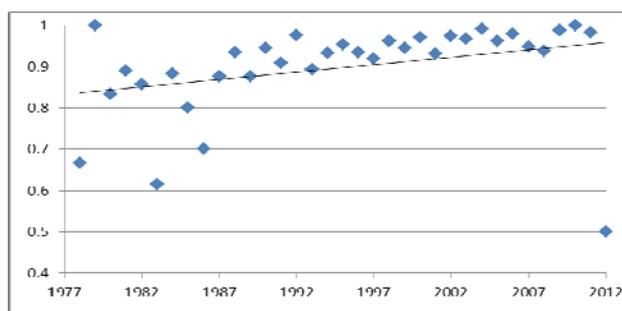


Figure 1. Scatter plot and trend line of data from Table 8

5. Threats to Validity

Although our benchmarks are not a full-grown empirical experiment there still exist threats to the validity or better the expressiveness of the presented results. Hence, it is helpful to briefly discuss potential issues that can be adapted from common checklists such as the one provided by Wohlin et al. (2000) for that purpose. According

to that reference there exist four groups of such threats, namely threats to internal, external, construct and conclusion validity.

As the name implies, the latter is concerned with factors that may negatively influence the validity of the conclusion of an investigation. Since we did not perform any statistical analyses, the only issue applicable from this group is the reliability of measures, i.e. the reliability of numbers provided by Google Scholar. As discussed in section 3.3., Google Scholar is indeed known for having some issues that affect the quality of its content. However, we have verified by manual inspections that repeated identical queries delivered identical results and that citation values are at least plausible and replicable for manual samples. Since all numbers presented have been collected with the same measures and tools, treatment implementation did also not influence the results. However, since our data collection required several days, we cannot fully rule out potential external influences such as changes in Google's database, for instance. Since Google Scholar is freely available and our approach is well documented, however, it is easy to replicate our investigation and to validate resp. update its results if doubts about their correctness should appear. Another threat to the conclusion validity is related to the diversity of software engineering and its scholars as overlaps with neighboring areas occur frequently (John Mylopoulos is a prominent example for this). Consequently, not all researchers that have published in software engineering and helped organizing ICSE, OOPSLA or FSE are necessarily software engineers to the core and hence publications and citations from other areas might have positively influenced the numbers determined for them.

Threats to internal validity are concerned with problems caused by the selection of subjects or their human idiosyncrasies and hence do not apply in our context since we did not directly interact with human subjects. However, the selection of the subjects certainly influenced the outcome of the benchmarks and it is highly likely that we missed some researchers with high citation numbers as well as that Google Scholar probably does not cover all publications and citations perfectly. Nevertheless, we believe that we have worked with the best data available today and that citations numbers are likely to be even higher if Google Scholar's coverage and citation recognition would be even better. Furthermore, the organizing committees of three major software engineering conferences and the ACM Sigsoft Award Winners should have yielded a comprehensive initial sample that covers most of the top scholars in the software engineering community. This issue is closely related with the idea of external validity, i.e. the generalizability of the results. The population we have chosen is of course highly selective so that citation numbers cannot be generalized to other communities, for example, and similar studies should be conducted there in order to allow a comparison beyond community boundaries. It is also questionable whether the good coverage results found for Google Scholar are valid for other (potentially non-technical or non-English speaking) researchers or research communities as well, as it is certainly reasonable to assume that computer science publications are more affine to electronic publication and indexing than other disciplines. Finally, since we did not execute an empirical experiment, threats to construct validity do not apply for our study.

6. Conclusion and Future Work

In this paper, we have presented results of a study aiming on creating a benchmark of citation performances from software engineering scholars as an important subfield of computer science. Such a collection can serve numerous purposes and since we have used freely available tools and clearly documented our approach, it can be used as a foundation for further investigations. Even researchers not listed here can get a good idea of the quality of their current lifetime publication record as they can simply compare themselves with the h-indices of around 60 and g-indices of around 150 achieved by top-notch software engineering scholars towards the end of their careers. Of course, it also allows funding bodies or potential employers to better classify individual citation indices and software engineering in comparison with other disciplines where significantly higher h-indices have been reached (as demonstrated by Palsberg's list mentioned before). It is also worth mentioning that h- and g-index results seem relatively reliable even when different result filtering techniques are applied for most scholars, unless they have very common names (such as David Evans, for example). However, as a concluding word of warning we want to cite Meyer et al. (2009): "Numerical measurements such as citation counts must never be used as the sole evaluation instrument (for researchers). They must be filtered through human interpretation [...]".

In addition to this initial citation benchmark, we have shown for a selective sample of 20 software engineering researchers that Google Scholar has reached a coverage level that goes beyond 95% for contemporary software engineering publications and can be expected to increase further in the future if it follows the trend of recent years. Furthermore, we have found that Google Scholar achieves an even higher coverage for journal publications as this has practically reached 100 % already. However, as often circulated before, the overall share of journal publications in computer science seems to be relatively low, as only about 20 % of all publications in the investigated sample have been published in journals. Other publication venues such as workshop or

conference proceedings still seem to be of high significance in this quickly developing discipline.

Although the general tendency indicates that Google Scholar delivers relatively reliable numbers, the absolute citation numbers and especially the publication numbers found in our study should be taken with a grain of salt. The reason is that the data collection and quality assurance approaches of Google Scholar still remain dubious and numerous quality concerns raised in the literature have rather been confirmed than falsified by our work. These concerns especially include the indexing of ghost publications and incorrect author lists for papers. However, since such ghost publications are practically not cited and a missing authorship rather decreases g- and h-indices for affected authors we still believe to have contributed a first valuable citation benchmark.

Based on the experience we have presented in this study it is certainly worthwhile for future work to increase the amount of considered researchers, e.g. by also taking authors of main conferences and journals and not only their program committee members resp. editors into account. Furthermore, we realized that a number of the claimed top-notch software engineering researchers are mainly working in related disciplines so that a clearer definition of “software engineering researcher” should be found. Perhaps the ongoing development of Microsoft’s Academic Search engine that clusters publications in research areas could give helpful input in this direction and can also be used for a comparison with Google Scholar and an in-depth comparison of delivered results. Finally, we feel that the coverage analysis we have presented can also be extended in numerous ways, including its size or the coverage and distribution of different publication venues such as conferences or workshops. Furthermore, it should also be interesting to analyse citations for certain time periods in order to compare the results of such a citation-based ranking with a publication-based ranking such as the one presented by Wong et al. (2011). From a national perspective it should also be interesting how well Google Scholar covers publications not written in English (and how often they get referenced) and how good national citation numbers are in comparison to the results found for world class researchers in this study.

References

- ACM. (2011). *ACM SIGSOFT Outstanding Research Award*. Retrieved from <http://www.sigsoft.org/awards/outResAwd.htm>
- Bar-Ilan, J. (2008). Which h-index? A comparison of WoS, Scopus and Google Scholar. *Scientometrics*, 74(2), 257-271. <http://dx.doi.org/10.1007/s11192-008-0216-y>
- Birukou, A., Wakeling, J., Bartolini, C., Casati, F., Marchese, M., Mirylenka, K., ..., Wassef, A. (2011). Alternatives to peer review: novel approaches for research evaluation. *Frontiers in Computational Neuroscience*, 5, 56. <http://dx.doi.org/10.3389/fncom.2011.00056>
- Bosman, J., Mourik, I., Rasch, M., Sieverts, E., & Verhoeff, H. (2006). Scopus reviewed and compared. *Technical Report*, Utrecht University.
- Chen, X. (2010). Google Scholar's Dramatic Coverage Improvement Five Years after Debut. *Serials Review*, 36(4), 221-226. <http://dx.doi.org/10.1016/j.serrev.2010.08.002>
- Dijkstra, E. (1968). Letters to the editor: go to statement considered harmful. *Communications of the ACM*, 11, 147-148. <http://dx.doi.org/10.1145/362929.362947>
- Egghe, L. (2006). Theory and Practice of the g-index. *Scientometrics*, 69(1), 131-152. <http://dx.doi.org/10.1007/s11192-006-0144-7>
- Elsevier. (2012). *About Scopus*. Retrieved from <http://www.info.sciverse.com/scopus/about>
- Engqvist, L., & Frommen, J. (2008). The h-index and self-citations. *Trends in Ecology and Evolution*, 23(5), 11270-11274. <http://dx.doi.org/10.1016/j.tree.2008.01.009>
- Falagas, M., Pitsouni, E., Malietzis, G., & Pappas, G. (2007). Comparison of PubMed, Scopus, Web of Science, and Google Scholar: Strengths and weaknesses. *FASEB Journal*, 22(2), 338-342. <http://dx.doi.org/10.1096/fj.07-9492LSF>
- Garfield, E. (1955). Citation Indexes for Science: A New Dimension in Documentation through Association of Ideas. *Science Magazine*, 122(3159), 108-111. <http://dx.doi.org/10.1126/science.122.3159.108>
- Harrold, M., & Rothermel, G. A. (1997). A System for Research on and Development of Program-Analysis-based Tools. *Technical Report OSU-CISRC-3/97-TR17*, Ohio State University.
- Harzing, A. (2010). *The Publish or Perish Book*. Tarma Software Research.
- Hirsch, J. (2005). An Index to quantify an Individual’s scientific Research Output. *Proceedings of the National Acad. of Sciences*, 102(46), 16569. <http://dx.doi.org/10.1073/pnas.0507655102>

- IEEE Computer Society. (2004). *Guide to the Software Engineering Body of Knowledge*. Retrieved from <http://computer.org/portal/web/swebok/html/ch1>
- Jacso, P. (2008). Google Scholar revisited. *Online Information Review*, 32(1), 102-114. <http://dx.doi.org/10.1108/14684520810866010>
- Jacso, P. (2010). Metadata mega mess in Google Scholar. *Online Information Review*, 34(1), 175-191. <http://dx.doi.org/10.1108/14684521011024191>
- Jin, B., Liang, L., Rousseau, R., & Egghe, L. (2007). The R- and AR-indices: Complementing the h-index. *Chinese Science Bulletin*, 52(6), 855-863. <http://dx.doi.org/10.1007%2Fs11434-007-0145-9>
- Liang, D., & Harrold, M. (1999). Efficient Points-To Analysis for Whole-Program Analysis. *Proceedings of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering* (pp. 199-215). http://dx.doi.org/10.1007%2F3-540-48166-4_13
- Meier, J., & Conkling, T. (2008) Google scholar's coverage of the engineering literature: An empirical study. *The Journal of Academic Librarianship*, 34(4), 196-201.
- Merton, R. K. (1968). The Matthew Effect in Science. *Science Magazine*, 159(3810), 56-63. <http://dx.doi.org/10.1126/science.159.3810.56>
- Meyer, B., Choppy, C., Staunstrup, J., & van Leeuwen, J. (2009). Viewpoint research evaluation for computer science. *Communications of the ACM*, 52, 31-34.
- Noruzi, A. (2005). Google Scholar: The New Generation of Citation Indexes. *Libri*, 55(4). <http://dx.doi.org/10.1515/LIBR.2005.170>
- Palsberg, J. (2012). *The h Index for Computer Science*. Retrieved from <http://www.cs.ucla.edu/~palsberg/h-number.html>
- Parnas, D. (2007). Stop the Numbers Game. *Communications of the ACM*, 50(11), 19-21. <http://dx.doi.org/10.1145/1297797.1297815>
- Ren, J., & Taylor, R. (2007). Automatic and versatile Publications Ranking for Research Institutions and Scholars. *Communications of the ACM*, 50(6), 81-85. <http://dx.doi.org/10.1145/1247001.1247010>
- Santini, S. (2005). We are sorry to inform you. *IEEE Computer*, 38(12), 128-127. <http://dx.doi.org/10.1109/MC.2005.423>
- Thomson Scientific. (2012). *About Web of Science*. Retrieved from <http://scientific.thomson.com/products/wos>
- Wainer, J., Goldenstein, S., & Billa, C. (2011). Invisible Work in Standard Bibliometric Evaluation of Computer Science. *Communications of the ACM*, 54(5), 141-146. <http://dx.doi.org/10.1145/1941487.1941517>
- Walters, W. H. (2007). Google Scholar coverage of a multidisciplinary field. *Information Processing & Management*, 43(4), 1121-1132. <http://dx.doi.org/10.1016/j.ipm.2006.08.006>
- Wohlin, C. (2009). A new Index for the Citation Curve of Researchers. *Scientometrics*, 81(2), 521-533. <http://dx.doi.org/10.1007/s11192-008-2155-z>
- Wohlin, C., Runneson, P., Höst, M., Ohlsson, M., Regnell, B., & Wesslén, A. (2000). *Experimentation in Software Engineering*, Kluwer. <http://dx.doi.org/10.1007/978-3-642-29044-2>
- Wong, W., Tse, T., Glass, R., Basili, V., & Chen, T. (2011). An Assessment of Systems and Software Engineering Scholars and Institutions (2003-2007 & 2004-2008). *Journal of Systems and Software*, 84(1), 162-168. <http://dx.doi.org/10.1016%2Fj.jss.2010.09.036>