8-2013

How many researchers does it take to make impact? Mining software engineering publication data for collaboration insights

Subhajit DATTA
Singapore Management University, subhajitd@smu.edu.sg

Santonu SARKAR

Sajeev A. S. M.

Nishant KUMAR

How Many Researchers Does it Take to Make Impact? Mining Software Engineering Publication Data for Collaboration Insights

Subhajit Datta
Singapore University of Technology and Design
subhajit.datta@acm.org

Santonu Sarkar
Infosys Labs, Bangalore, India
santonu_sarkar01@infosys.com

A.S.M.Sajeev
University of New England, Australia
sajeev@une.edu.au

Nishant Kumar
Appnomic Systems, India
Kumar.nishant1@gmail.com

ABSTRACT

In the three and half decades since the inception of organized research publication in software engineering, the discipline has gained a significant maturity. This journey to maturity has been guided by the synergy of ideas, individuals and interactions. In this journey software engineering has evolved into an increasingly empirical discipline. Empirical sciences involve significant collaboration, leading to large teams working on research problems. In this paper we analyze a corpus of 19,000+ papers, written by 21,000+ authors from 16 publication venues between 1975 to 2010, to understand what is the ideal team size that has produced maximum impact in software engineering research, and whether researchers in software engineering have maintained the same co-authorship relations over long periods of time as a means of achieving research impact.

Categories and Subject Descriptors

H.3.4 [**Information System**]: Information network-Social Information Mining; D.2.9 [**Software Engineering**]: Collaboration

General Terms

Benchmarking,Virtualization

Keywords

software engineering research, DBLP, topic analysis, T Test, Annova, Collaboration

1. INTRODUCTION

Empirical research is significantly more collaborative than the theoretical. The very nature of empirical sciences makes it necessary for numerous individuals to work together on research problems that involve planning, executing and evaluating experiments. This wider collaboration is reflected in the co-authorship trends of papers. While it is usual for papers in mathematics and theoretical sciences to have few authors, papers in the empirical sciences have many authors, in recognition of their varied contributions to the research process [1].

It is widely believed that the phrase "software engineering" (SE) was used in public discourse for the first time at a NATO conference in 1968 [2]. In 1975, the first dedicated venue for publishing software engineering research was introduced– the IEEE Transactions on Software Engineering (TSE). Subsequently, several other specialized publication venues came up. In the three and a half decades since 1975, software engineering has accumulated a significant body of research. This has also been the time of deepening penetration of software engineering- software artefacts dominate almost every aspect of our lives today. Software engineering as a sub-discipline of computer science, stands at the crossroads of formalism and empiricism. According to Shaw, software engineering started as adhoc craftsmanship, and gradually evolved into an empirical science [3]. Anecdotal evidence seems to support the increasingly empirical nature of SE; over the last decade premier publication venues have started expecting considerable experimental validation of research results [4].

In view of this we want to study how highly contributing SE researchers collaborate and whether collaboration facilitates the research impact. By collecting the publicly available data related to SE publication, content (abstract), author, and citation we built a model to understand the collaboration among researchers. In this paper report we report results from an empirical study involving 19,000+ papers written by 21,000+ authors from 16 software engineering research publication venues, over a 35 year time period from 1975 to 2010. We examine the following two questions involving the relation between team dimension and impact in software engineering research:

COMPUTE'13 Aug 22-24, Vellore, Tamil Nadu, India

- *What is the team size that has produced the maximum impact?*
- *Have team members collaborated together for long to maximize impact?*

We assume that the members of research teams get represented in paper authorship, so that the number of a co-authors of a paper indicate the size of the team which conducted the research. In the second question's context, collaboration is assumed to result in co-authorship relations between the individual researchers. Finally, we measure impact of a paper in terms of the number of citations it receives. We recognize some of the valid concerns that are raised against "counting" citations by way of judging a paper's impact [5]. However, there is little consensus on what constitutes a more valid measure of impact, and much of research evaluation in the academia and industry continues to rely on citation counts. So we believe our way of measuring impact – though not the only way – is largely consistent with the praxis.
The paper has been organized as follows. In the next section we describe the data that is necessary for our study. In Section 3 we describe our study setting. We explain our analysis method in Section 4. We have proposed two measures to analyze the nature of collaboration, namely the congruence of affinity and optimal team-size for maximum impact. These two measures have been explained in Section 5 and 6 respectively. Next we describe the impact and the instinct for collaboration over time in Section 7. We describe limitations of our analysis method in Section 8 and describe practical usage of this approach in Section 9. Finally we conclude the paper.

2. DATA IDENTIFICATION

To understand SE research discipline's characteristics along the three dimensions described in the previous section, we first need to identify the required data that can reflect on parameters of our interest.

1. First and foremost, we require the details of software engineering research papers that have been published in various venues, right from their inception. Alongwith this, we require the list of all authors (interchangeably called as researchers) of these papers.

2. In a research context, it is evident that an idea, or a topic of interest, is associated with a collection of papers. The question we need to answer is the mechanism to obtain a set of appropriate "SE-ideas" and associate them with the research papers. Trying to manually map a set of papers to a particular idea is not useful since it will be highly laborious, and prone to subjectivity and errors.

3. For measuring the contribution of a researcher, we consider two basic measures – publication count and citation count. We assume that the former reflects the amount of research published by a researcher, while the latter indicates the extent to which the researcher's work has been recognized. Though such count based metrics are not without controversy [**?**], much of academic and industrial research evaluation continues to rely on these measures.

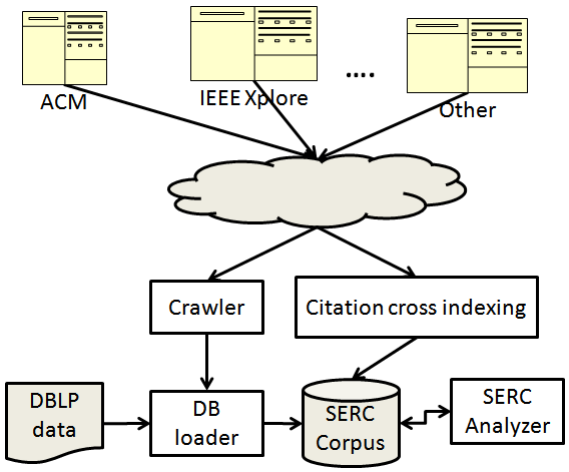

Figure 1: Schematic diagram of the SERC Tool

4. In order to measure interactions among highly contributing researchers, we consider their co-authorship information (which can be derived from a paper and its co-authors). Even though researchers can interact among themselves in other ways, we assume their interaction over research results in co-authored papers.

3. STUDY SETTING

The framework for our study is called the software engineering research corpus (SERC). We have implemented a tool as shown in Figure 1 that illustrates our approach to instantiate the SERC framework. We have chosen 16 major publication venues (alongwith their companion conferences) that focus on software engineering research as shown in Table 1.

3.1 Data Source

Our primary data source for papers and authors is from the DBLP site[1] that maintains the computer science bibliographic data for most of the conferences and journals with more than 2 million records. We have collected this database in MySQL format. The structure of DBLP data has been explained in [**?**]. The database dump dated April 23, 2011[2] was used for our study. In addition, we have considered publicly available information from ACM Digital Library[3], IEEE Xplore[4], Springer LNCS and other sites.
Our data-set is summarized in Table 1. Information around papers published in these venues is available at DBLP[5]. The database dump dated April 23, 2011[6] was used for our study. The citation cross indexing between papers and the citation count for authors was constructed using information available in the public

9[1]http://www.informatik.uni-trier.de/ ley/db/
9[2]http://dblp.l3s.de/dblp++.php
9[3]http://dl.acm.org
9[4]http://ieeexplore.ieee.org
9[5]http://www.informatik.uni-trier.de/ ley/db/
9[6]http://dblp.l3s.de/dblp++.php

Table 1: Publication Venues and Other Details

TSE - IEEE Transactions on Software Engineering
TOSEM - ACM Transactions on Software Engg. & Methodology
JSS - Journal of Systems and Software
IEEE SW - IEEE Software
ICSE - Intl. Conference on Software Engineering
OOPSLA/SPLASH - Object-Oriented Progg, Systems, Lang. & App.
FSE - Intl. Symposium on the Foundations of Software Engg.
ECOOP - European Conference on Object-Oriented Programming
FASE - Intl. Conf on Fundamental Approaches to Software Engg.
ASE - Intl. Conference on Automated Software Engineering
APSEC - Asia-Pacific Software Engineering Conference
ISSTA - Intl. Conference on Software Testing and Analysis
KBSE - Knowledge-Based Software Engineering Conference
WICSA - Working Conference on Software Architecture
CBSE - Component-Based Software Engineering
ISSRE - Intl. Symposium on Software Reliability Engineering
Total number of years (1975 to 2010, both inclusive) - 36
Total number of venues - 16
Total number of papers - 19,731
Total number of authors - 21,282

domain at ACM Digital Library[7], IEEE Xplore[8], and Microsoft Academic Search[9].
Paper abstracts were also extracted from these bibliographic repositories. We implemented specialized web crawlers to search each source in turn and store the data in a MySQL database[10]. A set of Java based components was developed to further process and analyse the data. Latent Dirichlet Allocation (see Sidebar-2) based topic analysis was performed using Mallet[11], while SPSS Statistics[12] was used for all statistical analysis.

3.2 Data Collection Process

We have collected and uploaded DBLP data in our SERC database in MySQL[13], shown in the topmost part of Figure 2. We show two main tables namely `dblp_author_ref_new` and `dblp_pub_new` that come from the DBLP site. For SE specific analysis, we have created a view of `dblp_pub_new` that contains only software engineering related publications based on the venues shown in Table1.

3.2.1 Paper Abstract

It is not sufficient to have the DBLP data only, as it does not contain information related to paper citation, author's H index information, topics of a paper, and how a topic evolves over time. To understand a research topic, we decided to use the abstract of a paper as the primary source since other means such as keywords or category descriptors are often not found. For the abstract of the software engineering related

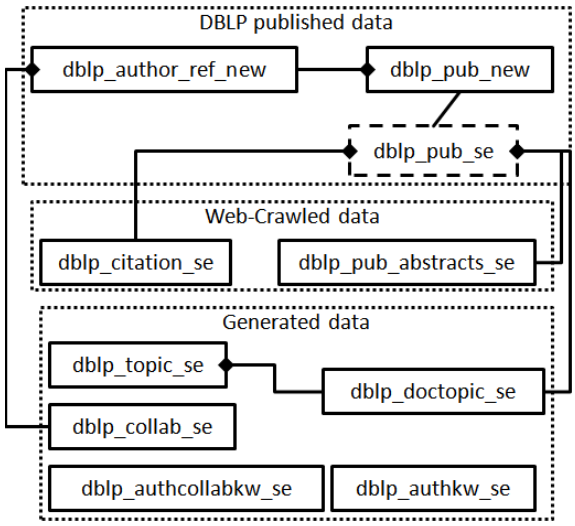

Figure 2: SERC Database schema overview

[7] http://dl.acm.org
[8] http://ieeexplore.ieee.org
[9] http://academic.research.microsoft.com
[10] http://www.mysql.com
[11] http://mallet.cs.umass.edu
[12] http://www-01.ibm.com/software/analytics/spss/products/statistics/
[13] http://www.mysql.com

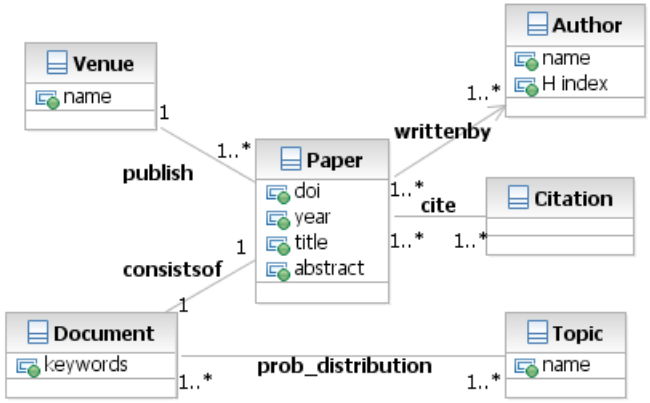

Figure 3: SERC metamodel overview

paper, we implemented specialized crawler (as shown in Figure 1) that searches each source in turn and store the data into additional database table `dblp_pub_abstracts_se`, as shown in the middle part of Figure 2. Each paper has a unique internal id generated by DBLP. From the crawled data we take the paper title and then search in the DBLP table to obtain the unique id and establish the referential integrity with `dblp_pub_se`.

3.2.2 *Citation*

The citation cross indexing module, shown in Figure 1, builds a citation cross reference database between papers in SERC using Once the this cross referencing for all papers whose citation information could be accessed is constructed, the citation count for authors is computed. Citation data collection for each paper happens in the following manner.

1. Use the cited paper title and *doi* to crawl from ACM, IEEE Xplore and other sites. We have developed site specific web scrapping methods for this purpose to get the citation link of this paper. This link is first stored in a file.

2. Next, for each link corresponding to the citing paper, the paper title is obtained

3. The cited paper title, id, the citing paper title are stored in `dblp_citation_se`

3.3 Data Representation

The metamodel for SERC has been shown in Figure 3. The metamodel

$$SERC =< \mathcal{V}, \mathcal{P}, \mathcal{A}, \mathcal{C}ref, \Gamma >$$

has five main elements. The central element is the set $\mathcal{P}$ of papers related to software engineering published in different venues from the inception 1975, till 2010. Each paper has attributes such as the *year of publication, paper title, abstract, doi* and so on. The complete set of attributes are defined in the DBLP schema [**?**]. In addition to the DBLP defined attributes, we introduce an additional attribute called *abstract* that contains the abstract of the paper. We have described the process of collecting the abstract in Sec. 3.2.1.
The next element is the set software engineering publication venues denoted by $\mathcal{V}$. The list of all the venues has been shown in Table 1). As shown in the metamodel (Figure 3), a venue publishes one or more papers whereas one paper is published in one venue. Next, $\mathcal{A}$ denotes the set of authors of these papers. Like P, attributes of the author is defined in the DBLP schema. Additionally, we define a new attribute called *H-index* for an author. The process of collecting the H-index has been highlighted in Sec. 3.2.1. As shown in the metamodel, an author can write one or more papers and one paper is written by one or more authors.
The next relation of the metamodel is the relation $\mathcal{C}ref \subset \mathcal{P} \times \mathcal{P}$ captures citation information between the publications.
The element Γ denotes the set of topics, represented as a set of probability distributions over documents. These topics are generated using a well-known topic discovery algorithm known as Latent Dirichlet Allocation (LDA)[6]. The input to this algorithm is a collection of documents where a document is treated as a *bag-of-words*, created from a paper by selecting a set of meaningful keywords from its title and abstract. Obviously, a paper has one-one relationship with a document as shown in Figure 3.

4. ANALYSIS METHOD

Following are the statistical tests – along with the underlying assumptions – we have used for our analysis in this paper.

4.1 T-Test

We wanted to test whether there is a statistically significant difference between citation counts of single-author papers and multiple-author papers. An independent samples t-test [7] can determine whether there is a statistically significant difference in the mean value of a variable between two groups of data. From the $SERC$ corpus, we have considered the set of single author papers ($\mathcal{P}_1 \subseteq \mathcal{P}$) form the first group and multi-author papers ($\mathcal{P}_{1+} \subseteq \mathcal{P}$ and $\mathcal{P}_{1+} \cup \mathcal{P}_1 = \mathcal{P}$) form the second group. For our test we set the alpha-value to be 0.05 which is standard. In a t-test, if the resultant p-value is less than or equal to the alpha value, the difference between the two groups will be considered as statistically significant.

4.2 ANOVA

We also wanted to test whether there is a statistically significant difference between citation counts of different groups of data; that is between single author papers, two-author papers, three-author papers etc. A one-way analysis of variance (ANOVA) test was used to check for this difference. Unlike the t-test where we had two groups of data, here we divided the dataset for the ANOVA into seven groups, the first six groups contained papers of one to six authors respectively and the last group with papers of more than seven authors. As in the previous test, alpha was set at 0.05. An ANOVA would give a p-value less than or equal to alpha-value if it found statistically significant differences between the groups' citation counts, however, it will not identify the particular groups that are significantly different from others. To identify this, a

post-hoc test such as Tukey's HSD [8] is needed to find out which of the groups are actually showing the significant differences.

4.3 Assumptions

The above tests are called parametric statistical tests for which the data need to satisfy certain assumptions. For these tests, normality of the data is not critical if the dataset is large enough [9], as in our case. The tests also assume similarity of variances in the different groups of the dependent variable which can be determined using Levene's test of homogeneity of variances. For our data, the Levene's test showed that this assumption was not met; when the size of samples in the different groups are vastly different, as in our case, it is quite possible for the variances to be different. SPSS's t-test handles such situations. For one way ANOVA, however, we used the Welch's F statistic to deal with this situation [8].

5. CONGRUENCE OF AFFINITY

For a given researcher we can compute his/her propensity or affinity towards one or more research topic(s) based on past publications. Discerning research "topics" from our corpus of 19,000+ papers is far from a trivial problem. Any manual process, in addition to being very tedious, is also prone to subjective bias and errors. In view of this situation, we decided to use a Latent Dirichlet Allocation (LDA) based technique to discover a set of topics from the corpus of all the papers [6]. LDA has been widely used to identify topics from large text corpora, specially in the context of research publications [10], [11].

Briefly, LDA considers a document to be a mixture of a limited number of topics and each word in the document can be attributed to one of these topics. Given a corpus of documents, LDA discovers a set of topics, keywords associated with each of the topics and the specific mixture of these topics for each document in the corpus. In our case, the set of papers $\mathcal{P}$ has been used as the text corpus (each document in this corpus is a stemmed set of keywords obtained from the paper title and abstract) from which LDA discovers a set of topics $\Gamma = \{\tau_1 \cdots \tau_k\}$. From a text corpus LDA creates two sets of probability distributions. One of these sets models topic mixture over documents (denoted as $\Theta = \{\theta_p | p \in \mathcal{P}\}$) and the other set models keyword mixture over topics. For a paper p, we get a probability distribution θ_p over topics, and for a given topic, we get a probability distribution of keywords. In LDA, these two are taken to be Dirichlet distributions with parameters α and β respectively. Arriving at the optimal number of topics for a given corpus is an empirical process. We need to vary α, β, the number of iterations (N) and the number of topics (K) to get the log likelihood value for the model which indicates its highest level of effectiveness [12]. Iterating over these parameters several thousand times, we selected 80 topics for our study. Beyond this number, we noticed that instances of repetitions in the keywords across the topics increased substantially, thus indicating a low possibility of identifying further distinguishable topics.

Having obtained the topic mixture model Θ over papers $\mathcal{P}$ using LDA. we now define the congruence of affinity for authors in our data-set as follows:

Definition: As mentioned earlier, let $\Theta = \{\theta_p | p \in \mathcal{P}\}$, where for a paper p, θ_p is the probability distribution over topics. Let $a.P$ be the set of all papers published by the author a, and $PaperTopic(\tau)$ for a topic τ is the set of all papers p whose topic mixture probability $\theta(p)$ is above certain threshold.

Now we define the affinity of the author a on a topic τ denoted as $aff(a, \tau)$, as:

$$aff(a, \tau) = \sum_{p \in (a.P \cap PaperTopic(\tau)} \theta_p(\tau)$$

Having defined $aff_a(\tau)$ we can now define a k-dimensional affinity vector for an author, considering all the k topics as:

$$\widehat{affvec}(a) = < aff(a, \tau_1), \cdots, aff(a, \tau_k) >$$

.

The *congruence* of past research interests between two authors a_1 and a_2 – $Congr(a_1, a_2)$ – is defined as the Euclidean distance between their affinity vectors. Thus:

$$Congr(a_1, a_2) = \sqrt{\sum_{i=1}^{k} (aff(a_1, \tau_i) - aff(a_2, \tau_i))^2}.$$

6. TEAM SIZE FOR MAXIMUM IMPACT

The conventional wisdom is that working in teams is more effective than working individually. From this point of view, team assembly mechanisms have been studied in creative enterprises [13], and corporate management is deeply engaged with the idea of effective teaming [14]. Even popular self-help books talk about "synergy" being "1+1>2" [15]. There is also significant literature on the benefits and challenges of collaboration in software development [16], [17]. In the context of software engineering research, we examined if there is indeed empirical evidence that multi-author papers (indicating teamwork) achieve higher impact than single author papers.

As mentioned earlier, we take the citation count of a paper as a proxy for its impact. The mean citation count ($\pm$ standard error of the mean) for single-author papers was 14.95 ± 0.77 and that for multi-author papers was 16.35 ± 0.37. An independent samples t-test showed that the difference between the two was not statistically significant (t(11452)=-1.76, p=0.078). This indicates that there has not been significant benefit for multiple authorships compared to single authorships in terms of research impact in software engineering, which is contrary to the conventional wisdom.

As we observe, there are many more multi-author papers than single author ones (Table 2). Accordingly, for a deeper understanding of how team size relates to research impact, we divided the data-set into seven groups, the first six groups contained papers of one to six authors respectively; since the number of papers with more than seven authors was relatively small (see Table 2), we decided to put them together with

Number of authors	Frequency
1	2684
2	4093
3	2637
4	1222
5	451
6	189
7	85
8	48
9	24
10	13
11	3
12	1
13	1
14	2
20	1

Table 2: Number of papers in the data set with different number of authorships

those of seven authors as the last group. Figure 4 shows the mean citation counts for papers with different number of authors. ANOVA demonstrated significance (Welch's F(6, 1142.87)=2.4, p-value=0.026). The post-hoc analysis showed that only the difference in mean citations between papers of four authors and seven or more authors was statistically significant.
As can be seen in Figure 4, papers with four authors had the least mean citation count and papers with seven or more authors had the highest mean number of citations. *Thus, even though the two-group analysis showed that there is no significant difference between single author and multiple author papers, the results here demonstrates that within multiple author papers, team work with seven or more authors pays off in impact compared to team work of four authors.*

7. IMPACT AND THE INSTINCT FOR COLLABORATING OVER TIME

In software engineering, as in some other disciplines, it is not uncommon for groups of like minded people to work together and publish jointly for a considerable period of time. Using our data-set, we are able to validate whether such a close relationship have been necessary to produce high impact outcomes in software engineering research. In order to test this question, we defined the following variables between each *collaborative pair* – that is, pairs of researchers who have co-authored at least one paper together:

- Number of joint papers published
- Number of overlapping years in publishing
- Number of common venues published in
- Number of common co-authors
- Congruence of affinity based on past publications

Intuitively, we see congruence of affinity, defined in Section 5 as the level of commonality between the research interests of two researchers based on past publications.

	Correl coeff	p-value
No of years of overlap	-0.011	0.011
No of common co-authors	0.012	0.004
Congruence of affinity	0.034	<0.001

Table 3: Impact of Closely Working Groups

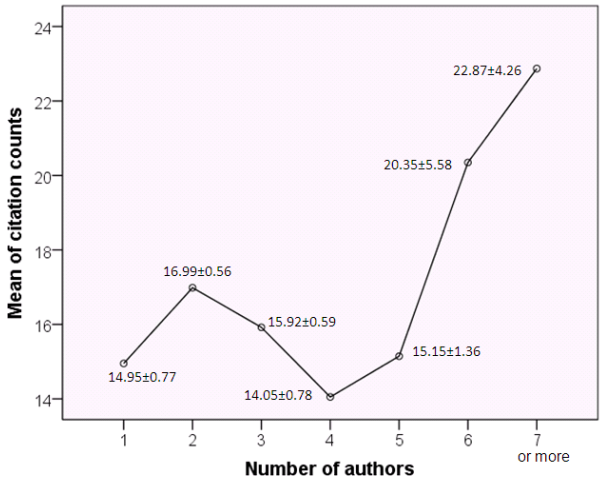

Figure 4: Mean citations versus number of authors (Mean and standard error are shown in the graph)

To understand whether and how collaborating over time relates to impact, we checked the Pearson correlation between the five variables identified above for each collaborative pair and the citation count of the papers for which the pair collaborated. Table 3 shows the correlations that are statistically significant.
The very weak correlations indicate that the impact a paper achieves is affected very little by the history of the collaborators in terms of common prior co-authorships, number of years they have published together or even congruence of their past research interests. *Thus to achieve impact, collaborators in general did not have worked together for long periods of time.*
In order to further confirm this result, we selected 100 papers with the highest citation counts in the data-set and determined their correlations. We found that in this case, only congruence of affinity was statistically significant (p-value=0.001) and its correlation with citation count was 0.129. That means, only 1.66% (i.e., 0.129^2) of the impact is explained by affinity, which being a very weak relationship, further confirms our result.

8. LIMITATIONS

Let us outline the following threats to the validity that limit the scope of our results:

Construct validity. implies that variables are measured correctly. In areas where there is considerable theoretical work, it usually involves establishing that the measurements are constructed in accordance with

theoretical foundations in the area. In our case, we have not been able to identify in the literature a metric which captures the extent to which two researchers can be said to share common research interests based on past publications. Hence the congruence of affinity metric (Sidebar-2) was defined. Measuring impact of a publication by counting the number of citations is widely practiced though not without controversy as we mentioned earlier [5]. Our results are bound by the limitation of the citation count measure.

Internal validity. addresses whether a study is free from systematic errors and biases. Since our data set is derived from all accessible publications in a predefined set of venues, issues that can affect internal validity such as mortality and maturation do not arise in our case. However, selection bias can occur from the manner in which venues of publications are selected for the study. Based on the established bibliographic sources considered, we believe we have accessed the maximal amount of data in our scope that is available in the public domain. The citation counts were based on citation cross indexing between papers that we constructed across several of our data sources. Papers in our corpus for which citation information is not available in the public domain, could not naturally be included in our analysis. Removal of self citations and disambiguation of author names are critical data cleansing activities for studies of this kind [18].

External validity. indicates the generalizability of the results of the study. The population for our study is all software engineering publications from the venues considered. Our sample size and the sampling method are unlikely to be a threat to external validity.

Reliability. of a study is related to reproducibility of the results. As we have minimized subjective bias in our analysis, and used automated approached whenever possible, our results can be easily reproduced.

9. BENEFITS AND TAKE-AWAYS

For an empirical study such as this one, it is important to establish the practical significance of our results. It is widely perceived that software engineering research is increasingly becoming empirical in nature [3]. In the empirical disciplines, research activity is largely driven by teams of researchers. How many researchers make the "optimal" team is question that is widely debated, with little consensus. In this paper, we examine this and related questions from a statistical viewpoint using a large data-set. We bring in a level of objective analysis to questions which are commonly confronted by opinions and subjectivity.

Our results have strong implications for the nurturing of software engineering research teams. In the academia, graduate students often collaborate with their advisers as well as other graduate students as they pursue their thesis completion. They are often faced with a dilemma as to whether it would be beneficial to work on a problem alone or seek out ideas and inspiration from others. While there is no one-size-fits-all answer to this question, our results would help discern some of the trade-offs inherent in going alone vis-a-vis working in a team. In the industry, research problems are often calibrated by their potential business impact. How many researchers to put on a problem is thus a question of much tactical importance. The findings from our study will inform the choices research managers in the industry frequently need to make.

One of the most widely cited conjectures in software engineering – canonized as Brooks' Law – states that adding more developers to an already delayed project makes it more delayed. This is taken as a reflection on how the overheads of interaction between increasing number of team members negatively impacts the team's deliverable. While on the development side of software engineering Brooks Law holds nearly universal sway, on the research side effects of the number of people working together on a problem is far more nuanced. Our results reveal this dichotomy; this is an interesting insight on the dynamics of SE research.

All researchers seek impact. Research impact comes out of a chemistry – often unknown – of a number of factors. In this paper we have systematically analyzed the impact of some such factors. We believe the insights gained on the influence of the number of researchers and co-authorship relationships lay the foundation for deeper analysis in future.

10. CONCLUSION

We observed no significant difference in impact between single-author and team based multi-author papers. However, when publications were grouped based on the difference in the number of authors, the work produced by large teams showed significantly higher impact. So in answer to the title question we can say, it needed at least seven researchers to work as a team to produce papers of significantly higher impact in software engineering. As we remarked earlier, papers in empirical sciences usually have higher number of co-authors than those in mathematics or theoretical sciences [1]. The fact that SE papers are found to achieve highest impact when there are seven or more collaborators seems to corroborate one aspect of the trend towards increasing empiricism in software engineering, as has been asserted by Shaw [3]. Additionally, we found that collaborators in software engineering do not maintain co-authorship relations with same individuals over long periods of time as a means of achieving research impact.

11. REFERENCES

[1] M. Newman, "Scientific collaboration networks. i. network construction and fundamental results," *Physical Review E*, vol. 64, no. 1, p. 016131, 2001. [Online]. Available: http://dx.doi.org/10.1103/PhysRevE.64.016131

[2] F. L. Bauer, L. Bolliet, and H. J. Helms, "Nato software engineering conference," 1968.

[3] M. Shaw, "Continuing prospects for an engineering discipline of software," *IEEE Software*, vol. 26, pp. 64–67, 2009.

[4] ——, "What makes good research in software engineering?" in *Proceedings of the European Joint Conference on Theory and Practice of Software*, 2002, pp. 1–7.

[5] D. L. Parnas, "Stop the numbers game," *Commun. ACM*, vol. 50, no. 11, pp. 19–21, Nov. 2007. [Online]. Available: http://doi.acm.org/10.1145/1297797.1297815

[6] D. Blei, A. Ng, and M. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[7] S. M. Ross, "Introductory statistics, 2nd ed." New York: Elsevier Academic Press, 1968.

[8] D. C. Howell, *Statistical methods for psychology*. Cengage Learning, 2011.

[9] B. Tabachnick and L. Fidell, *Using Multivariate Statistics*. Boston: Pearson Education, 2007.

[10] T. L. Griffiths, "Finding scientific topics," *Proceedings of the National Academy of Sciences*, vol. 101, no. suppl_1, pp. 5228–5235, 2004.

[11] Y. Jo, J. E. Hopcroft, and C. Lagoze, "The web of topics: discovering the topology of topic evolution in a corpus," in *Proceedings of the 20th international conference on World wide web*, 2011, pp. 257–266.

[12] H. M. Wallach, I. Murray, R. Salakhutdinov, and D. Mimno, "Evaluation methods for topic models," in *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009, pp. 1105–1112.

[13] R. Guimerà, B. Uzzi, J. Spiro, and L. A. N. Amaral, "Team assembly mechanisms determine collaboration network structure and team performance," *Science (New York, N.Y.)*, vol. 308, no. 5722, pp. 697–702, Apr. 2005, PMID: 15860629. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/15860629

[14] M. J. Wheatley, *Leadership and the New Science: Discovering Order in a Chaotic World*, 2nd ed. Berrett-Koehler Publishers, 2001.

[15] S. R. Covey, *The 7 Habits of Highly Effective People*, 1st ed. Free Press, Sep. 1990.

[16] J. D. Herbsleb and A. Mockus, "An empirical study of speed and communication in globally distributed software development," *IEEE Trans. Softw. Eng.*, vol. 29, pp. 481–494, June 2003. [Online]. Available: http://portal.acm.org/citation.cfm?id=1435631.859041

[17] K. Ehrlich, G. Valetto, and M. Helander, "Seeing inside: Using social network analysis to understand patterns of collaboration and coordination in global software teams," in *Proceedings of the International Conference on Global Software Engineering*, ser. ICGSE '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 297–298. [Online]. Available: http://dx.doi.org/10.1109/ICGSE.2007.39

[18] M. Ley and P. Reuther, ": The problem of data quality," *EGC*, vol. RNTI-E-6, pp. 5–10, 2006.