Analysis of the Scopus dataset to discern unscrupulous publication practices”

Ilias Dimitriadis

SID: 3301130016

SCHOOL OF SCIENCE & TECHNOLOGY
A thesis submitted for the degree of

Master of Science (MSc) in Information and Communication Systems

DECEMBER 2014
THESSALONIKI – GREECE
“Analysis of the Scopus dataset to discern unscrupulous publication practices”

Ilias Dimitriadis

SID: 3301130016

Supervisor: Prof. Michail Sirivianos

Supervising Committee Members:

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Information and Communication Systems

DECEMBER 2014
THESSALONIKI – GREECE
1. Abstract

This dissertation was written as a part of the MSc in ICT Systems at the International Hellenic University.

Academic evaluation boards, universities, research centers, etc. across the globe have started incorporating publicly available bibliometric data for the formal assessment of academics and researchers. Scopus is one of the biggest web-accessible databases of abstracts and citations of peer-reviewed literature.

We describe some of the useful bibliometric information, the theory behind them, we analyze their advantages and drawbacks and we focus on some of the possible ways to manipulate these metrics.

We present our effort to validate the hypothesis that standard graph analysis techniques can unveil citation manipulation through author collusion, superficial referencing, fake paper generation and other academically unacceptable practices. What’s more, we will investigate the relationship, if there is any, between the imposition of bibliometric based assessment and the emergence of the above bad publication practices by the researchers.

No part of this research would have been completed without the help of my supervisor Professor Mihail Sirivianos. I am also really grateful to Konstantinos Ntonas, the inventor of DEiXTo who has been more than helpful and kind to help me retrieve all the data needed from Scopus using the DEiXTo tool. As stated below all the source code related to this part is completely programmed by Mr. Ntonas. Last but not least, I would like to thank Professor Christos Berberidis, who was more than eager to help me with any problem I have faced during this dissertation, providing me with advice and solutions.
# Table of Contents

1. Abstract .................................................................................................................. 4
2. Introduction ............................................................................................................. 7
3. Literature Review ................................................................................................... 11
   3.1. The Impact Factor .......................................................................................... 11
   3.2. The H-index ..................................................................................................... 14
   3.3. The Eigenfactor Metrics .................................................................................. 17
      3.3.1. PageRank ................................................................................................ 17
      3.3.2. Eigenfactor Score & Article Influence .................................................... 20
   3.4. Variations & the Alternative Metrics ............................................................... 24
      3.4.1. SNIP & SJR .............................................................................................. 24
      3.4.2. Altmetrics .............................................................................................. 26
   3.5 The Eigentrust Algorithm .................................................................................. 27
      3.5.1. The Basic Eigentrust .............................................................................. 28
      3.5.2. The distributed Eigentrust ..................................................................... 29
      3.5.3. Secure Eigentrust .................................................................................... 30
4. Problem Definition ..................................................................................................... 33
   4.1. Academic Hoaxes ............................................................................................ 33
   4.2. The Researchers’ side ...................................................................................... 35
5. Contribution ................................................................................................................ 39
   5.1. Introduction ...................................................................................................... 39
   5.2. Data Selection .................................................................................................. 40
   5.3. Deixto Tool Personalization .......................................................................... 43
   5.4. Declaring the data as objects ......................................................................... 45
   5.5. XML Parsing ................................................................................................... 46
   5.6. Network of Papers .......................................................................................... 49
   5.7. Graph Visualization ....................................................................................... 52
   5.8. Graph Analysis & Metrics ............................................................................. 53
   5.9. Eigen Trust Implementation & Future Work ................................................ 53
   5.10. The PagenTrust JAVA APP ......................................................................... 54
   5.11. Graph Analysis results .................................................................................. 56
   5.12. The peer review problem .............................................................................. 74
6. Conclusion .................................................................................................................... 77
7. References ........................................................................................................................................... 79
8. Appendix ........................................................................................................................................... 83
  8.1. Deixto Source Code – XML Patterns – XML results .................................................................. 83
  8.2. JAVA Source Code ..................................................................................................................... 98
  8.3. Graphs ........................................................................................................................................ 120
2. Introduction

"Today, I wouldn't get an academic job. It's as simple as that. I don't think I would be regarded as productive enough." These are the exact words that the famous physicist and Nobel 2013 prize winner Peter Higgs used, to describe his academic work in an interview for the Guardian newspaper last December (Higgs, 2013). The reason behind this rather pessimistic opinion is the facts that Peter Higgs has published fewer than ten papers after his groundbreaking work in 1964. Considering the fact that academics nowadays bring out several papers per year, someone can understand the basis of this assumption. However all this publication “madness” seems to have surpassed the real motivation behind the publication of a scientific paper, article, etc., which is no other than to promote, discover and share knowledge with other scientists all over the world.

Thousands of articles, papers, and theses are submitted daily making it hard for all the publishers and scientist to separate the important, reliable and groundbreaking ones to others less important, fraudulent or full of errors. Thus the mass volume of all the emerging papers, created the need of some instruments for the evaluation of the researcher performance and subject’s importance. Although bibliometrics as a field emerged 40 years ago, no one expected that it would turn out to play such an important role in research assessment. There is a wide range of available bibliometric factors acting as indicators for the performance of each paper, journal, etc. In chapter 3 we are going to present, analyze and criticize all the bibliometrics indicators available starting from the ones who are more widespread.

Eugene Garfield is without a doubt one of the fathers of the bibliometrics, scientometrics and infometrics Science. In a 1955 Science paper, he suggested a new quantitative value of the citation index that would help the historians evaluate the influence of a journal, the so-called “impact factor” (Garfield, 1955). Forty years after this publication none, not even the author of that article would expect that “a tool designed primarily to alleviate problems of information retrieval and dissemination would foster the growth of quantitative studies of scientific output on a rather large scale” (Garfield, 1995). In chapter 3.1 we further
analyze the simple algorithm behind the calculation of the impact factor (IF from now on), the impact and the proper use of IF, its advantages and its drawbacks.

In 2005 Jorge E. Hirsch, a physicist at UCSD, introduced another bibliometric indicator, the Hirsch-index widely known as the h-index (Hirsch, 2005). Although it is a rather recent addition to the bibliometrics indicators family it has gained support and is used by many bibliometric sources. H-index is supposed to provide a measure for both the productivity and impact of a published work or the scientist/researcher behind it. In chapter 3.2 we attempt a deeper dive in the algorithm and the purpose of the h-index, unfolding some of the main advantages and disadvantages of its use and providing some alternatives based on the work of other researchers.

Another value recently introduced to bibliometric community is the Eigenfactor. Developed by Carl T. Bergstrom and Jevin D. West, the eigenfactor is actually a numerical representation of the total importance of a journal. It relies on the Pagerank algorithm introduced by Sergey Brin and Larry Page back in 1998 (Brin, et al., 1998) and consists of the eigenfactor score and article influence score which is comparable to the IF. Eigenfactor is gaining ground over the other bibliometric factors because it takes account of not only the number of citations but also the reputation of the incoming citations. In chapter 3.3 we move on with the explanation of the methods of calculation of the eigenfactor, the algorithm behind it, we present a comparison between this and the other indicators and we suggest a simple collaboration for better results.

The dominant presence of social media in the current state of the Web is not a controverted subject. It is a fact that the social media and the social nature of the web offers a chance for the creation of another metric which can be used to measure the impact or importance of scholar publications. The birth of altmetrics (alternative metrics) was marked by the following online manifesto: “No one can read everything. We rely on filters to make sense of the scholarly literature, but the narrow, traditional filters are being swamped. However, the growth of new, online scholarly tools allows us to make new filters; these altmetrics reflect the broad, rapid impact of scholarship in this burgeoning ecosystem. We call for more tools and research based on altmetrics.” (Priem, et al., 2010). Although we believe that it is by far the most manipulable metric, in chapter 3.4 we analyze the
reasoning behind it and we offer some examples of factors that could be used as altmetrics along with some variations of the most common bibliometric indicators.

The emergence of all these bibliometric data led to the creation and appearance of major citation indexing online services whose purpose was to allow researchers extract useful bibliometric information regarding publication venues and authors. However due to the mass production of scholar publications and to the need of a measure to evaluate researchers’ performance faster, academic evaluation boards started incorporating all these available bibliometric data for the formal assessment of academics and researchers. It is not a secret that this approach lurks some dangers for the reliability and the rightness of publication practices. In chapter 4.1 we refer to all the possible problems that may emerge from this “numbers-only” approach. In addition another perspective is brought to surface. We examine the current meaning of “peer-reviewing”. It is a fact that peer-reviewing of scholar publications may have been a veritable concept 50 years ago, when the volume of publications was somehow confined. Nowadays though, having in mind that only during 2008, researchers from China and the US published 500,000 papers by themselves, the concept of peer-reviewing loses points over the concept of a numbers-only approach for the evaluation of papers.

The problem researchers, academic boards, reviewers and review committees are facing is multilateral. On the one hand there is the research community composed by independent researchers, research teams, universities, etc. that are frustrated by the fact that their work is evaluated by a combination of numbers. This number only approach is the result of the daily on growing production of academic work throughout the entire world. On the other hand there are journals, universities, businesses, reviewers, librarians, etc., which are in need of a metric to evaluate and categorize, according to their impact level, a research or a researcher. In chapter 4 we attempt a more detailed approach as far as the problem all these people and communities face is concerned and we try to give some extra information that might be of interest.

After all this being said, we come to the conclusion that we need to find a method that will be fair and helpful for all sides. A new metric would not solve the problems above. Most probably it would add more trouble and would make things even more complex. It is
wiser to suggest a collaboration of already established metrics and algorithms in order to
draw safer and righteous conclusions. In chapter 5 a new methodology, along with some
experimental results, is presented that will help with the evaluation of academic work and
research. Its main concern is to spot researches, researchers and journals that adopt
unacceptable practices in order to achieve high rankings. A JAVA application has been
developed to help us draw these conclusions, called PagenTrust. Furthermore a new
implementation of an already established algorithm is suggested. It stands on a theoretical
level but it is interesting to look out for.

The goal of this dissertation is to prove that simple Graph analysis can prove the
existence of inadmissible techniques in the publication industry. A whole publication
network was established and a java application was developed in our effort to do so. In
chapter 6 we present our report on how successful our attempt to solve this matter has been.
Taking into consideration the final results and having in mind any future work we move on
with a final conclusion of this topic, while presenting our thoughts and worries on what the
future of bibliometrics and publications reserves.
3. Literature Review

3.1. The Impact Factor

E. Garfield first introduced the idea of the impact factor in 1955 (Garfield, 1955). It was supposed to be a simple way to compare journals no matter their size. Along the way, the notion of the impact factor evolved and instead of just measuring the influence and the importance of each journal, it also became an indicator of the author’s impact. As we are going to see later, this is not an acceptable approach.

Impact factor stands for the numerical representation of a journal’s influence and importance. It is easily calculated by dividing two numbers. The numerator is the number of citations of this year to any item published in a specific journal for the last two years, while the denominator is the sum of all the source items published during these two years in the same journal (Garfield, 1955). The impact factors for a specific year are published every year in the Journal Citation Reports (JCR) but they usually have a delay of approximately three years because the calculation takes account of the citations of this year for the articles of the previous two. This short citation time window of the IF is definitely a disadvantage because for many fields it takes a long time to start gathering citations.

<table>
<thead>
<tr>
<th>JOURNAL</th>
<th>2013 IF</th>
<th>JOURNAL</th>
<th>2013 IF</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Cancer Journal for Clinicians</td>
<td>162.500</td>
<td>Nature Reviews Drug Discovery</td>
<td>37.231</td>
</tr>
<tr>
<td>Chemical Reviews</td>
<td>45.661</td>
<td>Nature Materials</td>
<td>36.425</td>
</tr>
<tr>
<td>Reviews of Modern Physics</td>
<td>42.860</td>
<td>Nature Reviews Immunology</td>
<td>33.836</td>
</tr>
<tr>
<td>Nature</td>
<td>42.351</td>
<td>Nature Nanotechnology</td>
<td>33.265</td>
</tr>
<tr>
<td>Annual Review of Immunology</td>
<td>41.392</td>
<td>Cell</td>
<td>33.116</td>
</tr>
<tr>
<td>Nature Reviews Genetics</td>
<td>39.794</td>
<td>Science</td>
<td>31.447</td>
</tr>
<tr>
<td>The Lancet</td>
<td>39.207</td>
<td>Nature Reviews Neuroscience</td>
<td>31.376</td>
</tr>
<tr>
<td>Nature Biotechnology</td>
<td>39.080</td>
<td>Chemical Society Reviews</td>
<td>30.425</td>
</tr>
<tr>
<td>Nature Reviews Cancer</td>
<td>37.912</td>
<td>JAMA</td>
<td>30.387</td>
</tr>
</tbody>
</table>

Figure 1:
The Top 20 Journals according to the IS IF in 2013
It is obvious that the impact factor has a high dependency on two factors. The first is the ‘source items’ and the second ‘the citation items’. Apparently it is critical to discriminate which source items count as such and citations to which items count as citations. In this part of our review we encounter the first and very significant drawback of the impact factor. Unfortunately, what is defined as source item usually varies from journal to journal, meaning that is not always clear whether a given item will count as a source item in the denominator. This can lead to an exaggerated impact factor of some journals compared to others (Amin & Mabe, 2007). In other words, there are journals that name some published items in such a way that they do not count as source items, leading to the calculation of a higher impact factor.

Another point that we have to stress is that the impact factor depends on the size of the journal, on the subject field and on the number of the authors. The impact factor depends on averages (calculation of citations, etc), which means that it will have some variations due to statistical effects. A journal with 1000 articles for example is expected to have more citations than one with 10. It is obvious that such a difference would have an immediate effect on the IF. Now, concerning the subject field, having in mind that for example Social Sciences have far less papers published each year than Neuroscience, it would be unacceptable to compare the IF of Neuroscience journal to the one of a Social Science journal. Last but not least, we should also take into account the number of authors. Garfield’s comment on the latter issue is the following: “Therefore, the key determinants in impact are not the number of authors or articles in the field but, rather, the mean number of citations per article (density)” (Garfield, 1999). However we have an opposite opinion. It is widely known that all the authors have the tendency to cite their previous works in a new paper. It is neither illegal nor unacceptable and usually it has some basis. This means that for a paper with three authors the possibility of having direct citations to previous works is three times higher, verifying our previous claims. Summarizing, it does not make any sense comparing the IF of journals of different subject fields. What’s more we should also point out that there are many Sciences, such as Social Sciences and Humanities, that use different publication channels and citation practices than those covered by the IF (Glänzel, 2010).
Impact factor is widely used as a measure of quality. Obviously it is not a perfect tool but it has a quite long background in terms of use and is a rather good technique to evaluate scientific research. However, despite the fact that is widespread, it is “easily” manipulated. Manipulating the IF has many aspects. In the following paragraph we attempt to reveal some of the possible ways to do so. First of all, as we have already mentioned, the easiest way to affect the IF is by decreasing the source items countable in the denominator of the IF. This can be done by the journals themselves and the policy they decide to follow. Second, every author can cite his/her own self, meaning that while citing his own work he increases the total citation count of the journal which publishes his/her research and therefore increasing the IF of the specific journal, his research and his own prestige. Furthermore there is a practice followed by some publishers, which is called coercive citation. Coercive citation is all about forcing the author of an article, paper, etc to add forged citations either to him or other articles of the same journal in order to increase the citation count of the journal. The editors of journals usually follow such practices and in fact a recent research (Wilhite & Fong, 2010) showed that although it is considered to be an inappropriate and unethical practice, 57% of the people asked still say they would add mock citations before submitting to a journal known to coerce. Apart from the coercive citation practices followed by some editors, other manipulation techniques followed. Such one is described thoroughly by Alper (Alper, 2004) where an author submitted a case report to take as an answer that he should add some irrelevant references and what’s more he should submit it again as a letter to the editor (in order not to be counted as a source item). Last but not least there is a term called “citation stacking”, introduced by Thomson Reuters. Citation stacking happens when citation bursts from one journal to another are spotted, usually showing an affiliation between these two and quite often revealing the presence of common editors in both journals. At this point we should however stress out that as long as editors seem to care about their impact factors, they are interested in fresh, valid and important papers. As Hachinski states “When impact factors rise, editors editorialize. When they fall, editors fall silent” (Hachinski, 2001).
The IF holds of great respect inside the scientific community, although it is quite manipulative. However, what we consider as its greatest drawback is not that it is susceptible to manipulation but the fact that it does not take into account the importance of the citing papers. Whether an article is cited by a groundbreaking, highly cited work in a prestigious journal or cited by an abstruse work in a journal of questionable reliability, it has the same weight. Obviously this cannot be avoided since there is no algorithm behind the calculation of the IF which could contribute to the importance of the citing source. Pagerank (Brin, et al., 1998) is such an algorithm, whose basic features are used in other bibliometric indicators, as we will see later on.

3.2. The H-index

The H-index is a term introduced by the physicist Jorge E. Hirsch back in 2005 (Hirsch, 2005). It is an indicator for the evaluation of scientists’ research output. By definition “A scientist has index h if h of his or her Np papers have at least h citations each and the other (Np-h) papers have less than h citations each.” (Hirsch, 2005) As we can see it combines both the productivity and the impact a researcher’s papers have. It is based on the lifetime achievement of the researcher, showing both the citations and his/hers publications. Scopus and the Web of Science have incorporated the h-index in their databases, confirming our impression that the h-index is here to stay. The h-index is simple, but yet ingenious, providing the ground for comparison between scientists. It can under some limitations replace the impact factor, mainly because of its simplicity.
As most of the bibliometric indicators, h-index has many disadvantages. The first, and the most popular one, is still the fact that h-index depends on the field of studies. A comparison of the impact of two scientists of different fields based on their h-index is not acceptable. In addition, h-index has a negative effect on newcomers. A Scientist that starts publishing items has a very low h-index due to the fact that he does not have a large quantity of publications, which means that his citation count is also low. Thus, even if for example he has published a groundbreaking work, which accumulated many citations, his h-index will remain very low and a possible numbers only approach would favor another scientist with more but less important publications. What applies for newcomers also applies for scientists with short careers and for those who are very selective with the work they publish. As Hirsch indicates in his original paper (Hirsch, 2005), the h-index is a tool to evaluate researchers in the same stage of their careers. It is not meant to be a tool for historical comparisons and that is the reason he introduced the m factor that is the result of the division of h with the scientific age of the researcher.

The arrival of h-index led to the shift of some scientists’ perspective. (Bornmann & Werner, 2011) This shift has to do with the motivation behind the publication methods each researcher adopts. One of H-index’s main characteristics is that it is not affected by the

![Figure 2](image)
current performance of each researcher. It lacks sensitivity to performance changes. This means that if a researcher has done a great work in the past few years and has reached a high h-index, his h-index will not change even if he publishes less important or valid or no papers at all. The concept behind the H-index could foster productivity instead of promoting quality (Costas & Bordons, 2007), increasing the amount of low-level publications. Although the h-index of an author does not increase as his publications increase, the possibility to obtain a higher h-index rises as the total number of documents rises.

While ignoring the other bibliometric indicators, such practices could mislead us as far as the importance of a certain researcher is concerned. Van Raan in his paper (Van Raan, 2006) proved the strong correlation between h-index and peer judgment, but he also showed that this wouldn’t stand for smaller groups with less citation traffic. What’s more, h-index is susceptible to manipulation through self-citation and it does not take into account the importance of the citing source, characteristics of the impact factor as well, but still not solved by the h-index. A characteristic example is described in the hilarious paper of Cyril Labbé (Labbe, 2010). Labbé created a fake researcher called Ike Antkare and boosted his h-index to 94. To do this Labbé used software, created by an MIT group of scientists, called SciGen (Stribling, et al., 2005), to produce 102 publications and a network of self-citations. More details can be found in his paper. Although this took place in the Google Scholar service, Google’s citation indexing service, due to technical and methodological problems this service faces, it is a clear indication that h-index is susceptible to manipulation via self-citation. The ever-growing pressure for publishing exercised by academic boards, editors, universities, etc may have as a consequence the unleashing of malpractices to optimize the h-index and other bibliometric indicators.

It’s been almost ten years since the first appearance of the h-index, which led to the creation of another interesting front in bibliometrics. Many variations that are supposed to solve some of the problems have been proposed (g-index (Egghe, 2006), $h^2$ lower, $h^2$ center, and $h^2$ upper (Bornmann, et al., 2010)) but none has gained significant recognition by the scientometrics community. Despite of all its drawbacks, which we mentioned above, h-index’s popularity is rising year after year (Bornmann, 2014) and its use by major citation
indexing services like Scopus and Web of Science. However, its main limitations should be taken into consideration. The fact that someone cannot compare scientists of different subject areas, different publication years and of different scientific age should not be ignored. Impact factor still remains a very important indicator, despite of all its disadvantages, mainly because it is the industry standard for many years and has records of publications starting from the 1970s. We think that the growing use of h-index will have the same effects, people will still use it because it is becoming an industry standard regardless its deficiencies. We believe that this is a problem that the Bibliometrics community has to deal with shortly.

3.3. The Eigenfactor Metrics

While the above bibliometric methods (and their variations) are very popular because of their simplicity (mainly), they do not take advantage of all the useful information included in the citation network that they build. Most importantly they do not take into consideration where citations come from. This way, citations from prestigious and high quality journals count the same as citations from obscure, unreliable sources. Derek J. de Solla Price back in 1965, showed that citations form a huge network, which connects scientific papers to each other (de Solla Price, 1965). If we move up to the journals’ level, we can recognize each journal as a node in this network and the citations from each journal to another as links. The Eigenfactor metrics, initially developed by Carl T. Bergstrom, Jevin D. West and Marc A. Wiseman, are based on this exact citation network (Bergstrom, et al., 2008). The Eigenfactor attempts to rank journals according to their importance in the network. The algorithm behind this attempt is based on the Pagerank algorithm (Brin, et al., 1998). We think that in order to fully comprehend the mechanism behind the Eigenfactor metrics it is proper to continue with a review of the Pagerank algorithm, presented in chapter 3.3.a.

3.3.1. PageRank

PageRank is an algorithm developed in The Stanford University (Brin, et al., 1998). It was the deliverable item of an attempt to find other ways to implement a search engine
apart from the already existing ones. Pagerank takes advantage of the link structure of the World Wide Web, lies on the basis of a network built by all the links and pages and is a measure of the importance of Web pages. The Pagerank of a web page is high when the total sum of the rank of its backlinks (In a citation network backlinks stand for citations) is high. PageRank not only counts the backlinks of a page, but also extends this idea by not counting links from all pages equally. In other words PageRank takes into account the importance of each page in its calculation. Pagerank in the original paper of Page and Brin (Brin & Page, 1998) is defined as follows:

We assume page A has pages T1...Tn, which point to it (i.e., are citations). The parameter d is a damping factor, which can be set between 0 and 1. We usually set d to 0.85. Also C(A) is defined as the number of links going out of page A. The PageRank of a page A is given as follows:

$$PR(A) = (1-d) + d \left( \frac{PR(T_1)}{C(T_1)} + ... + \frac{PR(T_n)}{C(T_n)} \right)$$

Note that the Pagerank's form a probability distribution over web pages, so the sum of all web pages' Pagerank will be one.

PR (T_i) is the PageRank of page T_i that has connection with page A. C (T_i) shows the number of outbound links on page T_i and d stands for the damping factor. The damping factor is a measure of the probability that a random surfer jumps from one page to another. It is a product of the so-called random surfer model, which plays a very important role in the Pagerank algorithm. Brin & Page ended up with an empirical value of 0.85 for d. It implies that five times out of six the random surfer will choose a link on the webpage and one out of six will choose to go to a new page. In the equation above we see that the PageRank of A is recursively defined by the PageRank of those pages that link to page A. Pagerank is calculated using an iterative algorithm. Although it seems quite complex, the convergence of all the data happens quite fast. We can calculate a page’s PageRank without knowing the final value of the PageRank of the other pages. We could proceed with a more in-depth analysis of the PageRank algorithm, showing the math behind it but it is not included in the scope of this work.
The citation network has many similarities to the link structure of the World Wide Web. Every citation can be considered as a backlink and every reference as an outbound link. Instead of web pages we may have papers, articles, etc and journals as nodes of this network. Pagerank is used widely as a very efficient information retrieval tool, taking advantage of the link structure of the Web. Obviously PageRank may have similar results to the citation network. There have been many studies on this matter, showing the perspectives of using Pagerank for bibliometric purposes.

Back in 2007 researchers proposed the use of Google’s PageRank to evaluate the relative performance of publications (Chen, et al., 2007). They proposed a different damping factor, $d=0.5$, which they thought fitted best in the random surfer model as far as the citation network is concerned. In another research, a comparison between the impact factor and PageRank was made leading to the introduction of two new variants. The first was the weighted PageRank and the Second the Y factor (Bollen, et al., 2006). The Y factor is the product of Weighted PageRank and the Impact factor. The Weighted PageRank, viable only in the Journal Citation network since hyperlinks cannot have weights, also took account of the prestige transferred from one journal to another by adding a propagation proportion, a weight, since some journals are more connected to each other. Concluding, the research showed that despite the fact that Weighted PageRank overlapped with the IF, 

![Figure 3](image.png)

An iterative ranking scheme
there were many differentiations detected. When PageRank emerged, it led to a revolution in web searching and we think it can offer solutions in the bibliometric science covering the holes that the use of IF or other metrics have left behind.

Of course PageRank is not the holy grail of Bibliometrics. There are still issues to be faced. Pagerank manipulation is one problem, but it is more common when PageRank is used to categorize web pages. Google Bombing is such an example. Another disadvantage of PageRank is the treatment of dangling nodes. Dangling nodes are, in the citation network, those journals that are cited but do not cite other journals. The most important drawback of using PageRank as a bibliometric indicator relies on the fact that citations, unlike hyperlinks, cannot be updated after publication. This means that since a paper can only cite earlier publications an aging effect takes place, which is propagated while chain citations lead continuously to older and older publications (Maslov & Redner, 2008). PageRank operates like a “lifetime achievement award”. The absence of a time variant in the calculation equation is the reason why. In CiteRank a random researcher chooses a recent paper with probability that exponentially decreases according to the age of the publication (Walker, et al., 2007). Introducing two new parameters, CiteRank adds the time variable in the equation and the research proves that there are some optimal values that work for different subject fields.

Although PageRank seems to overtake other bibliometric indicators in matters of performance, it has not been adopted by any major citation indexing service. Eigenfactor, on the other hand, which is actually an alteration of PageRank including some additional features and modifications, seems to gain ground in the bibliometrics race.

3.3.2. Eigenfactor Score & Article Influence

As we have already mentioned, the scientific literature forms a network of publications, connected by citations (de Solla Price, 1965). Bergstrom and his colleagues’ attempt is all
about ranking journals as Google ranks web pages, in order to reveal the more, or less, influential journals. Influential journals are considered those that are cited by other influential journals. There is an obvious iteration in this method. This iterative algorithm is called Eigenfactor. This algorithm is consistent with a random researcher model, similar to the one used by Pagerank (the random surfer model). It simply describes the steps a random researcher follows when he goes to the library, reads a journal article by chance, moving on with one of the citations of this article, then proceeds with one of the articles cited by the citation and does this ad infinitum. The researcher obviously will choose to read articles that are hosted by important journals and will spend more time on those rather on small ones. The density with which our researcher chooses each journal is a measure of the importance of each journal (Bergstrom, et al., 2008).

![Diagram](image)

**Figure 4**

Scheme of most ranking algorithms versus the Eigenfactor

Eigenfactor metrics consist of two values, the Eigenfactor Score and the Article Influence. The Eigenfactor score represents the possibility of choosing a specific journal over others. For example if a journal has an Eigenfactor Score of 2, it means that the random reader will spend a two percent of the time on this journal over the total time. Apparently as the Eigenfactor Score rises, so does the influence of the specific journal. Eigenfactor Score is used to estimate the total value of a journal (Bergstrom, et al., 2010). The Article Influence score on the other hand is used to estimate the influence per article.
for a specific journal. It is a metric directly comparable to the Impact Factor. However, since the Article Influence is equal to the Eigenfactor Score divided by the total number of articles of the journal (normalized), takes into consideration the importance of each journal making it a more appropriate measure over the Impact Factor.

The math behind the Eigenfactor Scores is similar to this the PageRank algorithm uses. The algorithm computed eigenvector centrality weights for the value of citations (which are received by Thomson-Reuters Journal Citation Reports (JCR) database) and then calculating weighted citation rates for each reference item. Seven steps are followed in order to calculate the Eigenfactor Metrics and are the following (Bergstrom, et al., n.d.):

1. Data Input
2. Creating an Adjacency Matrix
3. Modifying the Adjacency Matrix
4. Identifying the Dangling Nodes
5. Calculating the Stationary Vector
6. Calculating Eigenfactor Score and Article Influence Score
7. Outputting the results

In other words, from the data available by JCR we extract a five-year cross citation matrix, since Eigenfactor counts all citations for a five-year target window, called $Z_{ij}$ (the citations from a journal j in a specific year to journal I during the five-year window. In this matrix we change all the diagonal elements to 0. This way we omit all self-citations. Normalizing Z by the column sums, we end up with the matrix $H_{ij} = Z_{ij}/\sum_k Z_{kj}$. Afterwards we compute $a$, which is the article vector and is equal to the number of articles published by a journal during the five year target window, divided by the articles of all the journals during this time period. To deal with all the possible dangling nodes of the network (those for which a row in the H matrix has all 0 entries), we change all 0s with the article vector. We end up with a different version $H'$ of the H matrix (Bergstrom & West, n.d.). A new row-stochastic matrix $P$ is defined as follows: $P = \alpha H' + (1 - \alpha) A$ where $A$, known as the teleportation matrix (West & Vilhena, n.d.), is composed of identical rows each equal to the article vector $a$, and $\alpha$ is a parameter set to 0.85. Let $p$ be the left eigenvector of $P$ associated with the unity Eigen value, that is, the vector $p$ such that $p = pP$. It is possible to
prove that this vector exists and is unique. The vector \( p \), called the influence vector, contains the scores used to weight citations allocated in matrix \( H \). The Eigenfactor vector \( r \) is computed as \( r = pH \), that is, the Eigenfactor score of journal \( j \) is: \( r_j = \sum p_i H_{ij} \).

In order to compare the Eigenfactor Metrics with the IS IF, we will choose the Article Influence instead of the Eigenfactor Score because it is closer to the notion of IF. First, as we have already mentioned there is very important difference in the way each metric takes into account the prestige of each journal. In the IS IF all citations are equal, either they come from a prestigious journal or not. Article Influence depends on the prestige of journals. Also, another important change in the Article Influence is the five-year target window it uses, giving the opportunity to papers (and Sciences) to be cited and acknowledged. There are certain Sciences that have these characteristics, Sciences in which it can take longer for an article to begin to receive citations (like Mathematics) so a two-year window (IS IF) is considered quite narrow. What is more, Eigenfactor Metrics do not count self-citations. Doing so, they are less susceptible to manipulation and they reduce the incentives for an editor to adopt a questionable unethical tactic. On the other hand, in IF self-citations can boost (as we have already mentioned above) a journal’s ranking. Last but not least, Eigenfactor Metrics consider the reference intensity of the citing journals (Franceschet, 2010). Citations that come from journals who do not use long bibliographies are considered more important than those at journals with long reference lists. As a result the difference between fields or different parts of same fields (that adopt different citing styles) reduces. The difference between fields of course is not eliminated, however when using the Article Influence as a metric we can find Journals that haven’t even reached the top 400 in the IF rankings featuring in the top 40 of journal ranking according to the Article Influence value. This gives us the opportunity to make a raw overall estimation of journals in total.
3.4. Variations & the Alternative Metrics

3.4.1. SNIP & SJR

SNIP (Source Normalized Impact per Paper), developed by Henk Moed (Moed, 2010) at the University of Leiden is an alternative solution based on the notion of the Impact Factor. SNIP measures contextual citation impact by weighting citations based on the total number of citations in given subject field. In other words, SNIP is a ratio of two measures, The Raw Impact per Paper divided by the Database Citation Potential. RIP is the average number of citations received in a specific year by articles, papers, etc published in the journal during the three preceding years. DCP shows a journal’s citation potential in the subject field it covers, since as we have already mentioned, the citation tactic and frequency varies from field to field. Some strong points of SNIP are that it takes under consideration the citation frequency of a research field and it also takes into account the immediacy factor. In addition, because it is based only on citations form peer-reviewed to other peer-reviewed papers means that it counters any potential for coercive citation techniques. The fact that it is based only on paper citations leads to a wider approach as far as a journal’s subject field is concerned. Since it counts the number of cited references published in journals processed for the database and not the total number of cited references in a field’s paper means that it accounts for how well the database covers the field. Of course there are still issues to be faced. Amongst others SNIP values tend to be higher for journals publishing review and it does not take into account the growth of the literature in a field (Moed, 2010). The Scopus Citation Indexing Database has adopted the SNIP metric along with the SJR, which is analyzed below.
SJR stands for Scimago Journal Rank and it is obvious that it is a metric for evaluating the importance of a journal. It was developed by SCImago Research Group (González-Pereira, et al., 2010). While SNIP is based on the principles of the IF, SJR is quite similar to the Eigenfactor Metrics and Pagerank. It is based on the group of eigenvector centrality methods and it is actually a prestige metric. All citations should not count the same. In order to achieve that, an initial identical amount of prestige is assigned to each journal, then a publication prestige is given by the number of papers included in the database and lastly a citation prestige is given depending on the number and importance of each received citation. Then, following the steps of an iterative algorithm similar to the one PageRank uses, the computation is carried out until we reach a point where convergence exists. Self-citations count up to a maximum of 33% in order to avoid excessive self-citation issues while a three-year time window is used so as to cover the peaks in the citation process and remain short enough to reflect the dynamics of the whole communication process (González-Pereira, et al., 2010). A process followed afterwards leads to normalization for differences in citation behavior between subject Fields. What is more, SJR is not easily
manipulated. The only way to raise your SJR score is by being published in more reputable and important journals. As shown above, SJR and SNIP are used by the Scopus database providing a quite good evaluation of a journal’s importance when both of them are taken under consideration.

3.4.2. Altmetrics

Bibliometrics as a field emerged more than fifty years ago but since then it has never stopped evolving. Undoubtedly the publications distribution channel has changed. The Web is the biggest source for scholar publications, researches and information in general. The volume of emerging papers keeps rising, impelling researchers to find new, simpler and better tools to evaluate the importance and impact of each scientific work. It was the early 00s when we reached to a key turning point in the Science Publishing History. What started as a tool to make the life of academics and the evaluation boards’ easier has turned into some’s nightmare. There is a growing frustration among the researchers’ community that their work is evaluated by a number only approach, taking into consideration just the values of out-of-date metrics that are just indicators of citation measurements (Nature Materials Editorial, 2012). At the same time, the peer-reviewing community is under heavy criticism for its deliverables.

There is a constant need for new metrics which have to adjust to the evolution of Science Publishing. Since Science Publications strengthen their Web presence day by day, why not use metrics that derive from the Web; a new term called altmetrics was introduced back in 2010, signaled by the manifesto mentioned in the introduction. In the beginning altmetrics was based on a Twitter hash tag (#altmetrics) that regarded the article level metric. However the term “altmetrics” refers to a measurement of the online presence, the web footprint, of either articles, people, journals, papers, authors, etc. It is the structure of the Web, the Social Networks outbreak and the Web 2.0 Framework that makes such a measurement viable. Instead of using impact factors, h-index and other bibliometric indicators one could assess a scientific work by its online presence, the number of times it has been viewed, the number of times it has been downloaded or someone made a “tweet” about this work, the times it was “re-tweeted”, the “likes” it gathered on a Facebook Post,
how many people bookmarked the web page that hosted this work, etc. Mendeley, property of Elsevier publishing company, is a web tool/program that uses altmetrics to evaluate the importance of an article, counting the times people have read or intend to read an article.

However, one should distinguish all these altmetrics to short and long term. Short-term metrics, such as the tweets and likes, are just indicators of the popularity of a specific work and not of the importance. Long-term metrics, as the numbers of downloads of a paper or comments, which are collected in a slower rate are more significant. It is obvious, that these metrics are susceptible to all kind of manipulation techniques. It is quite easy for someone to boost his posts, likes or tweets. It is also possible for someone to “buy” likes and gain popularity over the Social Media. It is a simple SEO (search Engine Optimization) procedure. Another issue that we need to keep in mind is that there are two aspects of popularity, good popularity and bad popularity. What we mean is that a paper may abruptly gain too much attention not because of being innovative or controversial but rather because it has been retracted or been identified as fake, causing an “alarm” to set off in the social media and over the web. Of course there is always a positive aspect in all conditions and in the specific case altmetrics could act as an indicator of “hot” scientific subjects and fields that attract more attention in the academic community.

3.5 The Eigentrust Algorithm

EigenTrust is a peer-to-peer reputation trust algorithm that was introduced and developed by S. Kamvar, M. Schlosser and H. Garcia-Molina in 2003. They described this algorithm as a way to decrease the number of inauthentic files in peer-to-peer networks (P2P). It is based on the notion of transitive trust. For a P2P network this means that if a certain peer i trusts another peer j, it also trusts those that j peer considers to be trustworthy. In order to do so they assign a global trust value to each peer, which is calculated by the aggregation of local trust values of all peers that had an experience with this certain peer. Roughly they define “the global reputation of each peer i is given by the local trust values assigned to peer i by other peers, weighted by the global reputations of the assigning peers." (Kamvar, et al., 2003).
As we have already said, the EigenTrust algorithm responds to the need of a P2P network where peer i downloads a file from peer j, peer j from a peer k and so on. Every time a peer i downloads something from peer j it rates with +1 a satisfactory transaction and with -1 an unsatisfactory one.

Hence, the local trust value that peer i has for j is:

\[ S_{ij} = \sum_{tr_{ij}} = \text{sat}(i, j) - \text{unsat}(i, j), \text{ where } t_{ij}=1 \text{ or } -1 \]

We have to normalize these results so:

\[ C_{ij} = \frac{\text{max}(s_{ij},0)}{\sum_{\text{max}(s_{ij},0)}}. \]

Normalizing these results gives a trust value of i for j at each peer between 0 and 1 for each peer. To aggregate trust values for a peer k, a peer m will have to ask all well known peers about their opinions of k. Using these values, a peer i can calculate the trust \( t_{ik} \) it places in peer k, by adding all the opinions of its acquaintances about peer k, weighted by the trust peer i places in them: \( t_{ik} = \sum_{ij} c_{ij} c_{jk} \). By using matrix notation, where \( C \) is the matrix \([c_{ij}]\) and \( t_i \) the vector containing the values \( t_{ik} \) we have: \( t_i = C^T c_i \). This trust values only reflect the experience of peer i and his acquaintances. To get a wider view, peer i will ask its friends’ friends \( (t_i = (C^T)^2 c_i) \). If this continues \( (t_i = (C^T)^n c_i) \), it will have a complete view of the network after a large number \( (n) \) of iterations. If \( n \) is large, the trust vector \( t_i \) will converge to the same vector for every peer i, namely to the left principal eigenvector of \( C \). Hence, \( t \) is a global trust vector, the elements of which quantify how much trust the system as a whole places in peer j (Kamvar, et al., 2003). There are three different ways to calculate \( t \), the basic EigenTrust, the distributed EigenTrust and the secure EigenTrust.

3.5.1 The Basic EigenTrust

For this basic variant the P2P aspect of the network is completely ignored and the computation takes place in a central system. This leads to:
Several issues are not considered by this algorithm:

1. Pre-trusted peers: In a P2P network some peers should have higher reputation ratings from the beginning than all the others. This is done by adding a start vector $p$, so that $t = (c^T)p$ converges faster than $t = (c^Te)$.

2. Inactive peers: They redefine $C_{ij}$ to incorporate this:

$$C_{ij} = \begin{cases} \frac{\max(s_{ij}, 0)}{\sum_j \max(s_{ij})} & \text{if } \sum_j \max(s_{ij}) \neq 0; \\ p_j & \text{otherwise} \end{cases}$$

While $p_{ij}$ is the $i$-th component of start vector $p$.

3. Malicious Collectives: A group of peers (G) can easily subvert the P2P system by giving all members of G high and all other peers low local trust values. For omitting that a factor $a < 1$ is added to the equation of $t^{(k+1)}$ and the final basic EigenTrust algorithm is:

$$f^{(0)} = \bar{e};$$

repeat
$$t^{(k+1)} = C^T t^{(k)};$$
$$\delta = ||t^{(k+1)} - t^k||;$$
until $\delta < e$;

**Algorithm 1**: Simple non-distributed EigenTrust algorithm

3.5.2. The distributed EigenTrust

In this implementation of the EigenTrust algorithm each peer stores its local trust vector $C_i$ but it also store its own global trust value $t_i$. 

$$f^{(0)} = \bar{p};$$

repeat
$$t^{(k+1)} = C^T t^{(k)};$$
$$t^{(k+1)} = (1 - a)t^{(k+1)} + a\bar{p};$$
$$\delta = ||t^{(k+1)} - t^k||;$$
until $\delta < e$;

**Algorithm 2**: Basic EigenTrust algorithm
Many of the parts of the equation above will be zero because peer i hasn’t had much interaction with other peers. So in other words, each peer i asks all the peers that have downloaded files from it, for $t_{ij}^{(0)} = p_{ij}$ and then inserts a loop where it calculates $t_{i}^{(k+1)} = (1-a)(c_{1i}t_{1i}^{(k)} + \ldots + c_{ni}t_{ni}^{(k)}) + a p_{i}$, then sends $c_{ji}t_{j}^{(k+1)}$ to all peers j that have downloaded files from it and receive back $c_{ji}t_{j}^{(k+1)}$ from all peers that have downloaded files from it. The loop breaks when $t_{i}^{(k+1)}$ and $t_{i}^{(k)}$ is less than a value e.

3.5.3. Secure EigenTrust

There are some issues connected with the distributed EigenTrust. The most important one is that malicious peers can easily report false trust values, capsizing the system. Moreover these malicious peers can assign incorrect values to other peers during the calculation of other peers’ trust. In order to deal with these problems more than one peers are used to calculate a trust value. The fore mentioned peers are called score managers. Every peer has a number M of score managers whose position is covered with the use of DHTs (Distributed Hash Tables). Having these in mind, the secure EigenTrust algorithm can be defined. If M is the number of score managers for each peer, $h_{0} \ldots h_{M-1}$ a number one way hash functions and $pos_{i}$ a position of peer I in the hash space the final algorithm will be:

```
foreach peer i do
    Submit local trust values $c_{i}^{'}$ to all score managers at positions $h_{m(pos_{i})}$, $m = 1 \ldots M - 1$;
    Collect local trust values $c_{d}^{'}$ and sets of acquaintances $B_{d}^{'}$ of daughter peers $d \in D_{i}$;
    Submit daughter d’s local trust values $c_{d}$ to score managers $h_{m(pos_{d})}$, $m = 1 \ldots M - 1$, $\forall j \in B_{d}^{'}$;
    Collect acquaintances $A_{d}^{'}$ of daughter peers;
    foreach daughter peer $d \in D_{i}$ do
        Query all peers $j \in A_{d}^{'}$ for $c_{jd}p_{j}$;
        repeat
            Compute $t_{d}^{(k+1)} = (1 - a)(c_{1d}t_{1d}^{(k)} + c_{2d}t_{2d}^{(k)} + \ldots + c_{nd}t_{nd}^{(k)}) + a p_{d}$;
            Send $c_{d}t_{d}^{(k+1)}$ to all peers $j \in B_{d}^{'}$;
            Wait for all peers $j \in A_{d}^{'}$ to return $c_{d}t_{j}^{(k+1)}$;
        until $|t_{d}^{(k+1)} - t_{d}^{(k)}| < \epsilon$;
    end
end
```

At this point we should refer to the deficiencies of the EigenTrust algorithm. Some of them have already been mentioned but in order to use it properly we have to keep them in mind.
Thus, the EigenTrust algorithm can be manipulated by malicious peers which provide inauthentic files and assign incorrect high trust values to peers that are not reliable. Moreover these malicious peers can assign high trust values to other colluding malicious peers. Even trickier is the case where malicious peers act as normal peers by providing some authentic files too along with inauthentic ones.
4. Problem Definition

With regards to the previous chapter, where all the bibliometric factors were presented along with their pros and cons, we can more or less identify the nature of the problem we are to face. We need to find a way, a method, an algorithm that will give us fair, accurate results as far as the importance, the quality and the overall achievement and the influence of a researcher or a research itself is concerned. This “method” needs to have a rather simple implementation, be easy to use and have the advantage of being easily adapted and be quick. The problem we are facing is not simple. The fact that it exists for many years and still has not been solved makes it an even greater challenge. In our opinion, the reason this is happening is that throughout all these years an incorrect approach has been followed. The ongoing research concerning bibliometrics, the constantly emerging trend of using many different factors in order to be as accurate as possible shows that no one is actually satisfied with all the existing methods. But we need to focus more on the meaning of the word “no one” which was previously used. Who is no one? From our point of view the problem we are facing has two sides. We need to approach this challenge not only under the aspect of the academic boards, journals, universities but under the aspect of the researchers themselves as well.

4.1. Academic Hoaxes

Hoaxes in academia are not a new problem. Today’s publish-or-perish research culture is what leads in the generation of more and more fake papers and all the other academically unacceptable practices, which we are to investigate. Back in 2005 a program called Sci-Gen (as we already mentioned) was developed by a group of MIT students, who wanted to show how easy it is to publish a gibberish paper. However many applications have been developed in order to detect fake generated papers, such as “scigendetection” developed by Cyril Labbé, who previously had proved with his own work how a fake author with fake papers can reach the top in the academic community (Labbe, 2010). Scigendetection spots some patterns usually used by Sci-Gen to detect Sci-Gen generated papers (Anon., n.d.).
The problem is much harder to deal with when other malicious practices are involved. The main issue with these practices is that they exploit some of the already known disadvantages of the available bibliometric indicators. As we have already stated in the third chapter there are many bibliometric indicators and all of them can be somehow manipulated. In a few words, impact factor can be modified by each journal by choosing less source items to count as such, it is size dependent, self-citation still counts in favor of the paper, it cannot apply to different fields, it doesn’t take under consideration the importance of each paper and we can spot cases of citation stacking and forged citations. On the other hand h-index operates in such a way that newcomers are not treated equally, it lacks of sensitivity to performance changes, while self-citation, the importance of citing papers and the adjustment of different fields are still an issue. Eigenfactor metrics, an alternate and improved version of Pagerank, although is close to solving the existing problems is still unable to spot every unacceptable method used to achieve high rankings in the academic community.

There is anecdotal evidence that practices like superficial referencing, extreme self-citation, coercive citation, author collusion, fake author and paper generation and birth of fake journals take place throughout the academic community. What is more, these phenomena seem to become more and more popular every day. The purpose of all these practices is to trick all the available bibliometric factors by assigning a higher impact factor, h-index, etc to the paper, the journal that publishes the paper, the author of the paper and so on. We can provide numerous examples of papers and authors that held of great respect but after many years proved to be a hoax. We have analyzed all the citation issues in the previous chapter but what about author collusion; Author collusion is a rather recent phenomena, where authors from usually different universities, institutes or journals decide to cooperate in order to achieve a higher ranking. Their cooperation though does not rely on scientific work, rather than mutual citation support, superficial referencing and some times positive reviewing.

However we believe that the corruption in this field goes much deeper. For example the output of Chinese science is something to look out for. Although in the beginning of the 21st Century China held a minor portion of all the published scientific work, it has now
turned out to hold the second biggest share in the scientific community (economist, 2013). A report from Nature back in 2010 though, pointed out that one third of 6,000 scientific researchers in China admitted the use of fraudulent practices (Qiu, 2010). Such an example shows that we have to deal with thousands of hoaxes and not just some specific cases. What we actually need is the ability to spot these cases using what we already have. In the following chapter (chapter 5) we provide a possible solution to this problem.

4.2. The Researchers' side

The researchers’ community has been affected by far by the “publish & perish” culture currently dominating the academic community and all the industry related to it, and it is not a small industry. Scientific journals are of great importance. It is the default medium through which scientific researches can be assorted, the best place to store all the scientific results and a way to provide knowledge; innovative ideas and groundbreaking work to every scientist, researcher, etc throughout the world. They are one of the key factors that affect our well-being and our evolution. It is the publication pool however that is used as a primary measure of scientific assessment by universities, academic boards, funding agencies and are of great importance when hiring, promoting, investing on some researcher are concerned. Their influence can go even further, to nation decisions on an effort to achieve a higher ranking on science achievements. Since big rewards are following distinct publications, it is not odd that some people decide to adopt unethical practices to achieve their goals (Arnold, 2009). But what about all the researchers out there that do not follow unethical tactics and wish to publish their own work and get acknowledgement for it (if it is worth it of course).

It is clear that we, as scientists, are responsible for ourselves. It is our decision to follow the easy path or take the difficult road to the reconnaissance of our work. However in order to do so we have to make sure that all researchers, journals, publishers, academics are treated equally and have a rather common strict policy. On the contrary, what the currently established situation seems to be, nowadays researchers are forced to engage in wide-scale and systematic self-citation, pressure is put on them in order to use citations that favor a journal or another author, pay money so that their work gets published, rather than operate
on scholarly accepted grounds. It is obvious that there is a high need of a method that is reliable and fair when publication issues arise. We need to make sure that journals adhere to high standards and to discriminate those that follow these general basic, ethical rules from those that completely avoid complying with them in favor of fake acknowledgement.

Figure 6

Peer Review Reviewed

Most, if not all, of the well-respected journals provide peer reviewing as a service. Peer reviewing though, is tied up with an arising problem. Many cases have been spotted where although peer reviewing was available as a service it never took place and it is not just that. According to researches conducted (Bohannon, 2013), (Baxt, et al., 1998) even when peer review did take place the results were quite disappointing. In both cases the researchers used a fake paper to test whether it would pass the peer review and get published or not. The statistics for each case were overall common. As we can see in Figure 6, for the first case 157 journals accepted the paper and only 98 rejected it. In this point we should make
clear that the submitted papers were simple alterations of a specific credible paper but with such grave mistakes that any reviewer could identify and not accept it as publishable. Thus it is obvious, that peer reviewing also needs to be reviewed and in the next chapter we suggest a rather simple solution to this problem.
5. Contribution

As has been noted repeatedly, our goal is not to provide another metric, which will be added in the long list of bibliometric factors. We intend to show that with the help of the already established ones in combination with the Eigen Trust algorithm and using Graph analysis we will be able to spot papers, authors and consequently journals and institutes that adopt unacceptable publication practices.

5.1. Introduction

In order to achieve our goal, we will use bibliometric information, which will be retrieved by the Scopus massive online dataset using the “deixto” information retrieval tool.

1. First of all we have to choose the kind of information we need to retrieve from each paper in order to be able to use them properly (chapter 5.2).
2. Secondly we will use and transform the source code of “deixto” to come up with a pattern that best fits in what information we need to retrieve (chapter 5.3).
3. The next step is to declare all the information and data we have as “objects”. The whole analysis phase will be implemented on JAVA, which is an object-oriented language (chapter 5.4).
4. Afterwards we will proceed with the parsing of the XML files so as to be able to use them appropriately (chapter 5.5).
5. Our next goals are to show the interconnection between these objects and try to present them as a part of a network (chapter 5.6).
6. By the time we manage to build the network we will be able to produce the Graph that connects all the nodes of this network (chapter 5.7).
7. Since the Graph is ready we can proceed with calculating metrics on each node.
8. We will calculate the Pagerank and the eigenvector centrality of each node (chapter 5.8).
9. We will proceed with the findings of the calculation above (chapter 5.9).
10. Finally we will present a Java implementation of the Eigen Trust algorithm and we will present the next steps of this research (chapter 5.10).
5.2. Data Selection

Scopus delivers a comprehensive review of the world’s research output in the fields of science, technology, medicine, social sciences, and arts and humanities. It features smart tools to track, analyze and visualize research. It actually works like a highly customizable search engine for scientific research outputs. A search can be defined by terms of context, authors, affiliations, subject fields and can be as detailed as someone can imagine. We decided to retrieve information on papers that belong in the field of computer science, for the years 2010 and 2012. Using the Scopus search engine we came up with 25,512 results. There are 13,905 results for the year 2012 and 11,307 for the year 2010 which are sorted on the number of citations each publication has (see figure 7).

![Scopus Search Engine Results](image)

**Figure 7**

Scopus Search Engine Results

After clicking on each publication we can see more detailed information about it. So our first step is to gather all the links that refer to each publication separately. We will use the
“deixto” extraction tool to do so, as we will see in the next chapter (5.3). The page that provides all the data we need to know for each publication is shown in Figure 8 below. All the colored rectangles pinpoint the information we will need to extract for each publication.

Figure 8

Webpage with detailed information for each publication

In order to extract all the necessary data we will use a rather complex method (analytical description in chapter 5.3) based on the DOM (Document Object Model) (Hégaret, 2002). Below, in Table 1 we can see what data exactly we need to extract.

<table>
<thead>
<tr>
<th>Cyan</th>
<th>Black</th>
<th>Blue</th>
<th>Red</th>
<th>Green</th>
<th>Orange</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eid</td>
<td>Title</td>
<td>Authors</td>
<td>Affiliation</td>
<td>References</td>
<td>Citations</td>
</tr>
</tbody>
</table>

Table 1
1. **Eid:** A unique parameter for every publication in Scopus. Similar to DOI. It is included in the link of each publication.

2. **Title:** The title of each publication.

3. **Authors:** The names of the authors for every publication. Clicking on the name of each author a new unique parameter is passed called author_id. We are collecting both names and ids.

4. **Affiliation:** Information for the affiliation of each publication

5. **References:** A list of all the publications that this specific one refers to. We collect the eid for each publication but not the name of their authors.

6. **Citations:** A list with the publications that have cited this certain document. In order to collect this information we need to “dig” deeper and get directed to another page from which we can collect the eid, the title and the name of each author.

At this point we should add that we came across some technical problems. Although we were aiming to collect as many papers as possible we did not manage to do so because Scopus returns only the first 2,000 publications. This means that instead of 25,512 publications we were about to gather, we managed to collect only 4,000 (2,000 for each year). Also, during the process of scraping we came across several disconnections because Scopus can detect the scraping of multiple documents and stops every ~150 publications. Using the DeixtoBot (see chapter 5.3) we managed to gather as much information as possible.

All these data are enough to provide us the basis on which our network of papers is to be built. There are of course improvements to be made, for example we could not retrieve the author ids for the references but it is in our future plans. This lack of authors for the references though does not mean that we cannot build our network, just that we will compromise with the unique ids, the eid, for every publication – reference.
5.3. Deixto Tool Personalization

As we have already mentioned, most of the scraping process will be done using an information retrieval tool (freeware) called “Deixto” (Kokkoras, et al., September 19-21, 2013). As stated in “deixto” official page: “DEiXTo (or ΔEiXTo) is a powerful web data extraction tool that is based on the W3C Document Object Model (DOM). It allows users to create highly accurate “extraction rules” (wrappers) that describe what pieces of data to scrape from a website”. It is very important to say that all this work has been done completely by Konstantinos Ntonas, who is the founder of deixto and to whom I am really grateful, after deciding together what we wanted to do exactly. None of the following coding and programming tasks (chapter 5.3) has been delivered by me.

![DEiXTo GUI](image)

**Figure 9**

The GUI interface of DEiXto

First of all as we have already mentioned we had to collect all the links to the individual pages of each publication. To do so we used the GUI of deixto (Figure 9). After gathering the data we used Microsoft Excel to manually remove some redundant parameters
(scopus_detail_urls.xls – See Appendix), such as the session id of each process which caused trouble when we were to use the links that we gathered. Since we had gathered all the detailed links for the 4,000 most cited papers for years 2010 and 2012, we moved on with the implementation of a Perl script (scrape_details.pl – See Appendix) which visited all the available links one by one and built an XML file for each of the publications, containing all the necessary data. Because of the fact that there was too much information in each page, it was not possible to scrape all the data needed with just one pattern. Thus, multiple XML deixto patterns were developed (XML Patterns – See Appendix). Those were the following:

1. Affiliation
2. Author A (for the authors of the paper)
3. Author B (for the authors of the citations)
4. Reflist (for the list of references)
5. Scopus_citations_details (for the title, the eid and the authors of citations)
6. Scopus_urls (for the URLs of each publication)
7. Scopus_urls_2 (for the URLs of each reference)
8. View_All_Citations (to get directed to the page that contains all the citations)

We also have to mention that in order to view and collect all the information of the citations we had to visit another page containing all of them. From the citations available we could only save 200 of them because of the Scopus policy. However only one publication had more than 200 citations, so that is not considered to be a problem.

The detail page indexing source code is based on Selenium (Anon., n.d.) a very popular and reliable web browser automation tool and on the DEiXToBot Perl module. The latter is a Perl object for programmatic web browsing with the potential of processing DEiXTo DOM-based tree patterns (DEiXToBot.pm – See Appendix). We have used Selenium to open a Firefox dialog window so as to be able to visit all the target URLs, whose code is then used by the DEiXToBot to retrieve all the pieces of information we are interested in. In other words, Selenium stores locally all the web pages, which are then passed on to DEiXToBot. DEiXToBot via its own “get method” and the file://scheme opens these pages. Finally we should add that the module that is used in order to build the DOM of each page
inside the DEiXToBot’s source code is the XML::LibXML. The deliverables of this whole complex process are 4,000 unique xml files, each one containing all the needed information for each one of the 4,000 publications (Scopus XML files – See appendix).

5.4. Declaring the data as objects

For the purpose of moving on with the construction of our network and any further analysis we have to transform all our data in objects. Before doing so though, we should create our objects so as to be ready for use when needed. To do so we will use JAVA, a very wide spread object oriented programming language. To make our life easier we will use Eclipse (Anon., n.d.). Eclipse is the most widely-used open source integrated development environment (IDE).

It is easy to understand that the main object will be a “publication”. Each one of our records describes a certain publication which consists of other publications, in our case citations and references. So we have the main publication, the citation and the reference. The term publication consists of many unique publications which are the objects for the class “publication”. Of course our class has some attributes. For each publication we will definitely have an eid, a title, a URL and a collection of authors, citations and references. Of course authors is another object (class) which has as attributes a unique id, a name and their affiliation and consists of many separate authors (objects). In figure 10 (see also in Appendix) we will show the whole structure of our source code. Most of the classes that appear in that figure are to be implemented in the next steps but the diagram gives us a quite clear idea of the whole structure.
5.5. XML Parsing

One of the most demanding tasks throughout the whole process has been the parsing of all the XML data that we had already collected in order to be used in our JAVA code. There are many implementations of simple XML Parsers available in the web and they work great if what you are supposed to parse is a simple XML file. On the contrary, the XML files that we gathered are rather complex. The tags that we used while scraping the Scopus dataset were not chosen wisely. For example, instead of giving another name for the authors of the citations we used the same one as for the original publication. This means that if we would...
like to parse the document by simply identifying the tag names we could not be able to do so. However, our XML file is based on the DOM model and there are still solutions. Some of the most common XML Parsers for JAVA are the following:

- The Sax Parser
- The Dom Parser
- The StaX Parser

In the beginning we used the Sax Parser but we faced multiple problems. Afterwards we used the Dom Parser which happened to be the right one for our occasion. The reason is that the Dom Parser loads the complete XML content into a Tree structure. After multiple iterations through the Nodes we get to retrieve the XML content. In Figure 11 we can take a look at a sample of our xml files.

![XML File Sample](image)

**Figure 11**

Scopus XML file sample
As we can see the parent node is a publication. All the other nodes are considered as child nodes. However there are many child nodes that also have children nodes. Furthermore some of the nodes contain tag attributes which we want to retrieve (authors_id for example). To achieve a successful parsing this is the methodology we used:

1. First of all we have to build a file loader in JAVA. It scans all the documents with an .xml extension in the folder we specified.
2. The second step is to develop a Document Builder and Parser. These are default steps adjusted to our case.
3. Now we need to identify the beginning of our xml file, so we our looking for a <Publication> tag.
4. Since we find this we start building our tree by getting all the Child Nodes using the method .getChildNodes. In Figure 12 we can see a code snippet for this task.
5. We identify all the nodes by their tag names using the .getNodeName method and we do this repeatedly for all the child nodes. When we encounter an “authors” tag we use the same method as before to get its own children and the .getAttribute method to retrieve the corresponding attribute.
6. In the case of the citation authors we have to follow the preceding method thrice in order to get as deep as possible and retrieve all the information we need.

This is an iterative process as we can see in the XMLReader.class (See Appendix) which stops only when all the child nodes have been parsed.
What is more important for the whole analysis phase is the construction of the network of papers and authors we are asked to implement. It is this hypothesis that sets the ground for further analysis. Up to now we have collected all the information we needed, we moved on with the parsing of these data and their transformation into a form which can be used to create the publication network. With this intention we are asked to search for a tool or build our own that will help us depict the connection between the publications we have already

**Figure 12**

XML Parser sample source code

### 5.6. Network of Papers

What is more important for the whole analysis phase is the construction of the network of papers and authors we are asked to implement. It is this hypothesis that sets the ground for further analysis. Up to now we have collected all the information we needed, we moved on with the parsing of these data and their transformation into a form which can be used to create the publication network. With this intention we are asked to search for a tool or build our own that will help us depict the connection between the publications we have already
scraped. Since we use JAVA, after a detailed search we ended up using a JAVA library called JUNG (Java Universal Network/Graph Framework (Anon., n.d.)).

JUNG is a software library that provides a common and extendible language for the modeling, analysis, and visualization of data that can be represented as a graph or network. It is designed to support a variety of representations of entities and their relations, such as directed and undirected graphs, multi-modal graphs, graphs with parallel edges, and hypergraphs. In our case we will use a directed multisparse Graph.

First of all we need to build a new class that we call MyEdge. MyEdge has as arguments a pair of publications, the source publication and the target publication. We have to keep in mind that our goal is to build a network graph whose vertices are connected to each other (or not) with edges, so since we have created our edges (MyEdge) we can move on with the creation of a Graph. Hence, we proceed with the creation of a new class called MyGraph which holds as arguments the pair publication-edge. Publications will be the vertices of the graph and the edges what connects them (Figure 13).

![Figure 13](image)

A simple graph formed by vertices and edges

At this point we should add that we have created another class under the name PublicationManager that handles all the methods that have to do with the tasks where the class publication is involved. In this class we implement a method called createGraph which constructs the whole network of publications (see Figure 14 – See Appendix for the whole source code).
Figure 14

Java code for the creation of the network

The scope of this method is obvious. To be more specific it turns every publication to a vertex. So first of all we have the creation of vertices from all the available main publications and then we turn the references and the citations to vertices as well. The connection that takes place between them is directed. That means that a publication points to a reference and a citation points to a publication.

Furthermore, although it is not necessary, we have also implemented a method that counts the number of clusters that are created through this network. The cluster size actually shows the density of the connections between all the publications throughout the whole network of publications (Number of components). For our dataset (4,000 unique publications) we come up with a total of 24 components.
5.7. Graph Visualization

It would be very interesting to see how this whole network can be illustrated in a single Graph. To do so we create another class called GraphVisualizer (see Appendix). This class implements methods that are related to the depiction of the Graph in our screen. Apart from the methods that are related to this specific task we also add some methods that give us the ability to adjust the Graph as we want using our mouse. However, in order for the Graph to be viewed properly we need to have a very large screen and quite high processing power, both of which we did not have the chance to use. In figure 15 a simple Graph created by almost 1,000 publications is shown. The time it took for the Graph to reach its final form was almost two hours with the means available. The Graph for the whole network can be found in the Appendix.

Figure 15

A network of 1,000 publications – Graph
Nevertheless, it is important to say that the visualization of the Graph is not necessary. It does not provide us with further information than those that we can retrieve from the simple Graph analysis we are to execute.

5.8. Graph Analysis & Metrics

The Graph Analysis we are about to execute consists of two main parts. The calculation of PageRank and the calculation of the Eigenvector Centrality for each Publication. We have already explained what PageRank is in chapter 3.3.1. Eigenvector Centrality is actually an implementation quite similar to the Eigenfactor (chapter 3.3.2). The 'eigenvector centrality' for a vertex is defined as the fraction of time that a random walker will spend at that vertex over an infinite time horizon.

To calculate the above metrics we will create two more classes, the PageRankOfNode and the EigenVectorOfNode. Both of them have as attributes the node id, the node title and the PageRank or eigenvector accordingly. The calculation of both metrics takes place in the PublicationManager class and there is an already implemented algorithm in the JUNG library that executes the calculations. We simply adjust the source code accordingly to our case.

Apart from the calculation we include some other methods that are necessary. They are the methods that provide us with the wanted results. So, in the PublicationManager class we also include methods that print the PageRank for each node, the title of the node (where available since references don not include the title), the eid of the node and the authors. All of them are enough to draw out conclusions as far as the importance and impact of each publication is concerned.

5.9. Eigen Trust Implementation & Future Work

The whole point of this dissertation is to come up with an application that can deliver us all the suspicious publication and/or authors. The logic behind it, is that first we would calculate the PageRank of each publication, the eigenvector centrality of each publication and come up with some results. The results should be more or less similar. In our research
we found that there are some remarkable differences in the ranking of the papers with these methods, especially for those in the middle of the whole ranking. That is not enough though. Our main goal is to calculate the trust of all nodes, peers, publications (it is the same thing either way we call it) using the Eigentrust algorithm and then compare these results with the previous ones provided by the PageRank and Eigenvector calculation.

In order for the EigenTrust algorithm to work we would assign a positive trust value to some peers, publications. The ones that we would choose would be the ones that take the first ten places in the PageRank and the Eigenvector rankings. It is quite obvious that such results cannot be manipulated, so we are sure that these papers are indeed reliable and trustworthy. Through an iterative process that is further described in chapter 3.5 we get to calculate all the local trust values for each peer, ending up with a simple ranking of the most trustworthy peers to the less trustworthy ones. We believe that these results will be quite different than the ones we found during the previous calculations. The ones that appear to have an important difference in the rankings will be the suspicious ones. Since we would be able to spot the suspicious ones we could then further analyze these papers, authors, journals, etc. to come up with the final judgment.

The problem is that there is no implementation of the Eigen Trust algorithm available. Due to the limited time we had for the completion of this dissertation, a deliverable implementation of the Eigen Trust algorithm for our case was not developed. It is however an ongoing research and we plan to deliver a ready to use JAVA Application that would spot suspicious behavior in the near future. We call this application PagenTrust. In the next chapter 5.10 we present our Java Application. What is more in chapter 5.12 we also introduce a new idea as far as peer reviewing is concerned. It aims to face the problems that rise due to the existence of defective and unreliable peer reviewing.

5.10. The PagenTrust JAVA APP

The PagenTrust JAVA APP is a stand-alone Java application which runs locally on any machine with JAVA installed (Figure 16).
The PagenTrust App

Its main tasks are the following:

- It loads the XML files using a File Chooser, where someone can choose the folder he wants.
- It analyzes the data. The analysis of data consists of the XML parsing and the calculation of components. During the process, information on which file is analyzed is shown.
- After the completion of the previous task, the app gives the user the option to calculate the PageRank and choose the value of the damping factor or the Eigen vector centralitiy. Either of the two options provides the results in the console.
- Someone can also choose the number of publications he/she wishes to print.
- Last, it provides the option for Graph Visualization. It is strongly advised that this should be selected only if there is high processing power available.
Of course all the above actions were already included in the source code we have presented so far. To reach the final form of this application, in other words the Graphical User Interface (GUI), we used an Eclipse Plugin called WindowBuilder. The process we follow is actually quite simple. Firstly we design the application window, we add the buttons and the parameter windows. What we need to do afterwards is to connect these buttons and brackets with the code we have already written and make sure that each time a button is clicked an event starts. The latter is done by creating a button listener for all the actions.

5.1. Graph Analysis results

In this section we will present the results that came up after the whole process we followed. As it has already been mentioned this section concerns 4,000 publications along with their references and citations. The 4,000 publications are from the years 2009-2010 and 2011-2012. The field of study of all these publications is “Computer Science”. With the help of our PagenTrust app we will calculate the PageRank and the eigenvector centrality for each publication. The PageRank results we are to present are produced when the value of the damping factor is equal to 0.85. Furthermore we have to make clear that we cannot end up with complete conclusions, due to the lack of the results coming from the implementation of the eigentrust algorithm, but we can understand the methodology behind this whole process.

We will present two tables, each one contains the results for each of the two methods of calculating the importance of each publication. Table 2 contains the results for the implementation of PageRank with damping factor equal to 0.85. In table 4 we present the results of the Eigen vector centrality calculation. We also have to add that we present only the top100 results, although it is important to keep in mind that all 4,000 results are needed. Finally there are some results where the title is null. This is because the references did not include the title in the original XML files.

<table>
<thead>
<tr>
<th>Eid</th>
<th>Title</th>
<th>Pagerank Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2-s2.0-65649117685</td>
<td>7.18096557157772E-4</td>
</tr>
<tr>
<td></td>
<td>Title</td>
<td>DOI</td>
</tr>
<tr>
<td>---</td>
<td>----------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>2</td>
<td>2-s2.0-79955702502 LIBSVM: A Library for support vector machines</td>
<td>5.785637287622513E-4</td>
</tr>
<tr>
<td>3</td>
<td>2-s2.0-67649884743 null</td>
<td>4.580702460506854E-4</td>
</tr>
<tr>
<td>4</td>
<td>2-s2.0-61549128441 Robust face recognition via sparse representation</td>
<td>4.5756346893206475E-4</td>
</tr>
<tr>
<td>5</td>
<td>2-s2.0-65449188232 Jalview Version 2-A multiple sequence alignment editor and analysis workbench</td>
<td>4.5157950881283217E-4</td>
</tr>
<tr>
<td>6</td>
<td>2-s2.0-65449136284 null</td>
<td>4.330909828939651E-4</td>
</tr>
<tr>
<td>7</td>
<td>2-s2.0-68549104404 null</td>
<td>4.2087703775730813E-4</td>
</tr>
<tr>
<td>8</td>
<td>2-s2.0-77950347409 null</td>
<td>3.7974762720471036E-4</td>
</tr>
<tr>
<td>9</td>
<td>2-s2.0-77952649961 Linked data - The story so far</td>
<td>3.5006978985058576E-4</td>
</tr>
<tr>
<td>10</td>
<td>2-s2.0-62049084378 Research electronic data capture (REDCap)-A metadata-driven methodology and workflow process for providing translational research informatics support</td>
<td>3.2870311470548194E-4</td>
</tr>
<tr>
<td>11</td>
<td>2-s2.0-63649117166 null</td>
<td>3.16489834504555E-4</td>
</tr>
<tr>
<td>12</td>
<td>2-s2.0-68049121093 Anomaly detection: A survey</td>
<td>2.929940323522335E-4</td>
</tr>
<tr>
<td>13</td>
<td>2-s2.0-73449109459 Meep: A flexible free-</td>
<td>2.8262952190233093E-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>14</td>
<td>2-s2.0-84882385335</td>
<td>Pattern Recognition</td>
</tr>
<tr>
<td>15</td>
<td>2-s2.0-80052078099</td>
<td>null</td>
</tr>
<tr>
<td>16</td>
<td>2-s2.0-84881078972</td>
<td>Data Mining: Concepts and Techniques</td>
</tr>
<tr>
<td>17</td>
<td>2-s2.0-79551587720</td>
<td>Cytoscape 2.8: New features for data integration and network visualization</td>
</tr>
<tr>
<td>18</td>
<td>2-s2.0-77956877124</td>
<td>The Internet of Things: A survey</td>
</tr>
<tr>
<td>19</td>
<td>2-s2.0-77955422240</td>
<td>Object detection with discriminatively trained part-based models</td>
</tr>
<tr>
<td>20</td>
<td>2-s2.0-70449533121</td>
<td>ABINIT: First-principles approach to material and nanosystem properties</td>
</tr>
<tr>
<td>21</td>
<td>2-s2.0-77955309392</td>
<td>null</td>
</tr>
<tr>
<td>22</td>
<td>2-s2.0-77950537175</td>
<td>null</td>
</tr>
<tr>
<td>23</td>
<td>2-s2.0-56449087483</td>
<td>null</td>
</tr>
<tr>
<td>24</td>
<td>2-s2.0-67650711615</td>
<td>SOAP2: An improved ultrafast tool for short read alignment</td>
</tr>
<tr>
<td>25</td>
<td>2-s2.0-61349147303</td>
<td>null</td>
</tr>
<tr>
<td>26</td>
<td>2-s2.0-77957244650</td>
<td>null</td>
</tr>
<tr>
<td>27</td>
<td>2-s2.0-77950369345</td>
<td>Data clustering: 50 years beyond K-means</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>28</td>
<td>2-s2.0-73249139477</td>
<td>Object based image analysis for remote sensing</td>
</tr>
<tr>
<td>29</td>
<td>2-s2.0-77957243284</td>
<td>RDP3: A flexible and fast computer program for analyzing recombination</td>
</tr>
<tr>
<td>30</td>
<td>2-s2.0-57149144228</td>
<td>A survey of affect recognition methods: Audio, visual, and spontaneous expressions</td>
</tr>
<tr>
<td>31</td>
<td>2-s2.0-80054078476</td>
<td>Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega</td>
</tr>
<tr>
<td>32</td>
<td>2-s2.0-77951770756</td>
<td>BEDTools: A flexible suite of utilities for comparing genomic features</td>
</tr>
<tr>
<td>33</td>
<td>2-s2.0-77749249761</td>
<td>null</td>
</tr>
<tr>
<td>34</td>
<td>2-s2.0-77949587649</td>
<td>null</td>
</tr>
<tr>
<td>35</td>
<td>2-s2.0-77956031473</td>
<td>A survey on transfer learning</td>
</tr>
<tr>
<td>36</td>
<td>2-s2.0-77958110812</td>
<td>Conducting meta-analyses in R with the metafor</td>
</tr>
<tr>
<td>37</td>
<td>2-s2.0-67649598885</td>
<td>null</td>
</tr>
<tr>
<td>38</td>
<td>2-s2.0-71349088033</td>
<td>DBpedia - A crystallization point for</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>39</td>
<td>2-s2.0-74049108922</td>
<td>the Web of Data</td>
</tr>
<tr>
<td>40</td>
<td>2-s2.0-68549133155</td>
<td>BLAST+: Architecture and applications</td>
</tr>
<tr>
<td>41</td>
<td>2-s2.0-77953882250</td>
<td>Learning from imbalanced data</td>
</tr>
<tr>
<td>42</td>
<td>2-s2.0-77954583479</td>
<td>Picante: R tools for integrating phylogenies and ecology</td>
</tr>
<tr>
<td>43</td>
<td>2-s2.0-79961181125</td>
<td>Recent advances and industrial applications of multilevel converters</td>
</tr>
<tr>
<td>44</td>
<td>2-s2.0-84859061927</td>
<td>The ORCA program system</td>
</tr>
<tr>
<td>45</td>
<td>2-s2.0-63149185343</td>
<td>null</td>
</tr>
<tr>
<td>46</td>
<td>2-s2.0-61849169018</td>
<td>null</td>
</tr>
<tr>
<td>47</td>
<td>2-s2.0-67349183350</td>
<td>FactSage thermochemical software and databases - recent developments</td>
</tr>
<tr>
<td>48</td>
<td>2-s2.0-77954619566</td>
<td>null</td>
</tr>
<tr>
<td>49</td>
<td>2-s2.0-77951298115</td>
<td>null</td>
</tr>
<tr>
<td>50</td>
<td>2-s2.0-77955426203</td>
<td>Evaluating color descriptors for object and scene recognition</td>
</tr>
<tr>
<td>51</td>
<td>2-s2.0-79551668253</td>
<td>FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>52</td>
<td>2-s2.0-77955894071</td>
<td>METAL: Fast and efficient meta-analysis of genomewide association scans</td>
</tr>
<tr>
<td>53</td>
<td>2-s2.0-63149159130</td>
<td>null</td>
</tr>
<tr>
<td>54</td>
<td>2-s2.0-77952299957</td>
<td>Prodigal: Prokaryotic gene recognition and translation initiation site identification</td>
</tr>
<tr>
<td>55</td>
<td>2-s2.0-70449516548</td>
<td>Spectrum-efficient and scalable elastic optical path network: Architecture, benefits, and enabling technologies</td>
</tr>
<tr>
<td>56</td>
<td>2-s2.0-84883905026</td>
<td>Google's PageRank and beyond: The science of search engine rankings</td>
</tr>
<tr>
<td>57</td>
<td>2-s2.0-61449215161</td>
<td>Mean squared error: Lot it or leave it? A new look at signal fidelity measures</td>
</tr>
<tr>
<td>58</td>
<td>2-s2.0-80051762104</td>
<td>Distributed optimization and statistical learning via the alternating direction method of multipliers</td>
</tr>
<tr>
<td>59</td>
<td>2-s2.0-64549119687</td>
<td>null</td>
</tr>
<tr>
<td>60</td>
<td>2-s2.0-73849116467</td>
<td>null</td>
</tr>
<tr>
<td>61</td>
<td>2-s2.0-79952709519</td>
<td>pROC: An open-source package for R and S+</td>
</tr>
<tr>
<td>Code</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>62</td>
<td>2-s2.0-79952003251</td>
<td>to analyze and compare ROC curves</td>
</tr>
<tr>
<td>63</td>
<td>2-s2.0-67650099990</td>
<td>Differential evolution: A survey of the state-of-the-art</td>
</tr>
<tr>
<td>64</td>
<td>2-s2.0-84867293252</td>
<td>Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels</td>
</tr>
<tr>
<td>65</td>
<td>2-s2.0-78049312324</td>
<td>GenALEX 6.5: Genetic analysis in Excel. Population genetic software for teaching and research-an update</td>
</tr>
<tr>
<td>66</td>
<td>2-s2.0-76749083002</td>
<td>Image super-resolution via sparse representation</td>
</tr>
<tr>
<td>67</td>
<td>2-s2.0-61749090884</td>
<td>Networking named content</td>
</tr>
<tr>
<td>68</td>
<td>2-s2.0-77951965920</td>
<td>Functional brain networks develop from a &quot;local to distributed&quot; organization</td>
</tr>
<tr>
<td>69</td>
<td>2-s2.0-67049119331</td>
<td>Skyline: An open source document editor for creating and analyzing targeted proteomics experiments</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>70</td>
<td>2-s2.0-58349111875</td>
<td>AmiGO: Online access to ontology and annotation data</td>
</tr>
<tr>
<td>71</td>
<td>2-s2.0-71049116435</td>
<td>null</td>
</tr>
<tr>
<td>72</td>
<td>2-s2.0-62249142056</td>
<td>null</td>
</tr>
<tr>
<td>73</td>
<td>2-s2.0-65549111754</td>
<td>null</td>
</tr>
<tr>
<td>74</td>
<td>2-s2.0-70350642087</td>
<td>null</td>
</tr>
<tr>
<td>75</td>
<td>2-s2.0-59649083826</td>
<td>Differential evolution algorithm with strategy adaptation for global numerical optimization</td>
</tr>
<tr>
<td>76</td>
<td>2-s2.0-69349090197</td>
<td>null</td>
</tr>
<tr>
<td>77</td>
<td>2-s2.0-61949087310</td>
<td>Word sense disambiguation: A survey</td>
</tr>
<tr>
<td>78</td>
<td>2-s2.0-70349626031</td>
<td>Adaptive particle swarm optimization</td>
</tr>
<tr>
<td>79</td>
<td>2-s2.0-60249084757</td>
<td>null</td>
</tr>
<tr>
<td>80</td>
<td>2-s2.0-66549114708</td>
<td>Outstanding research problems in NoC design: System, microarchitecture, and circuit perspectives</td>
</tr>
<tr>
<td>81</td>
<td>2-s2.0-65749110333</td>
<td>null</td>
</tr>
<tr>
<td>82</td>
<td>2-s2.0-79953048649</td>
<td>null</td>
</tr>
<tr>
<td>83</td>
<td>2-s2.0-60849139395</td>
<td>GOrilla: A tool for discovery and visualization of enriched GO terms in ranked gene lists</td>
</tr>
<tr>
<td>84</td>
<td>2-s2.0-64849107578</td>
<td>Architectures for</td>
</tr>
<tr>
<td>Page</td>
<td>DOI</td>
<td>Title</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>85</td>
<td>2-s2.0-78649738324</td>
<td>Multi-cell MIMO cooperative networks: A new look at interference</td>
</tr>
<tr>
<td>86</td>
<td>2-s2.0-67349161331</td>
<td>Controlled microwave heating in modern organic synthesis: Highlights from the 2004-2008 literature</td>
</tr>
<tr>
<td>87</td>
<td>2-s2.0-77953810904</td>
<td>null</td>
</tr>
<tr>
<td>88</td>
<td>2-s2.0-70349675664</td>
<td>Bayesian phylogeography finds its roots</td>
</tr>
<tr>
<td>89</td>
<td>2-s2.0-70350769233</td>
<td>null</td>
</tr>
<tr>
<td>90</td>
<td>2-s2.0-66449084442</td>
<td>null</td>
</tr>
<tr>
<td>91</td>
<td>2-s2.0-84872057266</td>
<td>null</td>
</tr>
<tr>
<td>92</td>
<td>2-s2.0-77954199597</td>
<td>PSORTb 3.0: Improved protein subcellular localization prediction with refined localization subcategories and predictive capabilities for all prokaryotes</td>
</tr>
<tr>
<td>93</td>
<td>2-s2.0-70449350937</td>
<td>null</td>
</tr>
<tr>
<td>94</td>
<td>2-s2.0-80053512597</td>
<td>Open Babel: An Open chemical toolbox</td>
</tr>
<tr>
<td>Eid</td>
<td>Title</td>
<td>Eigenvector Centrality</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>96</td>
<td>null</td>
<td>9.428115542142515E-5</td>
</tr>
<tr>
<td>97</td>
<td>null</td>
<td>9.321047213117137E-5</td>
</tr>
<tr>
<td>98</td>
<td>A survey of context modelling and reasoning techniques</td>
<td>9.284973433915352E-5</td>
</tr>
<tr>
<td>99</td>
<td>A survey of mobile phone sensing</td>
<td>9.25671242217836E-5</td>
</tr>
<tr>
<td>100</td>
<td>null</td>
<td>9.23207976514269E-5</td>
</tr>
</tbody>
</table>

Table 2. The PageRank rankings

<table>
<thead>
<tr>
<th>Eid</th>
<th>Title</th>
<th>Eigenvector Centrality</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DnaSP v5: A software for comprehensive analysis of DNA polymorphism data</td>
<td>0.0017806930532711636</td>
</tr>
<tr>
<td>2</td>
<td>LIBSVM: A Library for support vector machines</td>
<td>0.0014337733293561697</td>
</tr>
<tr>
<td>3</td>
<td>null</td>
<td>0.0012284509927789297</td>
</tr>
<tr>
<td>4</td>
<td>Robust face recognition via sparse representation</td>
<td>0.0011928549296818236</td>
</tr>
<tr>
<td>5</td>
<td>null</td>
<td>0.0011771998550223867</td>
</tr>
<tr>
<td>6</td>
<td>Jalview Version 2-A multiple sequence alignment editor and analysis workbench</td>
<td>0.0011438075887215823</td>
</tr>
<tr>
<td>7</td>
<td>null</td>
<td>0.001136022728048169</td>
</tr>
<tr>
<td></td>
<td>Title</td>
<td>URL</td>
</tr>
<tr>
<td>---</td>
<td>----------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>8</td>
<td>Linked data - The story so far</td>
<td>null</td>
</tr>
<tr>
<td>9</td>
<td>Research electronic data capture (REDCap)-A metadata-driven methodology and workflow process for providing translational research informatics support</td>
<td>null</td>
</tr>
<tr>
<td>10</td>
<td>Anomaly detection: A survey</td>
<td>null</td>
</tr>
<tr>
<td>11</td>
<td>Meep: A flexible free-software package for electromagnetic simulations by the FDTD method</td>
<td>null</td>
</tr>
<tr>
<td>12</td>
<td>null</td>
<td>null</td>
</tr>
<tr>
<td>13</td>
<td>Data Mining: Concepts and Techniques</td>
<td>null</td>
</tr>
<tr>
<td>14</td>
<td>Cytoscape 2.8: New features for data integration and</td>
<td>null</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>----</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td>21</td>
<td>2-s2.0-77955422240</td>
<td>Object detection with discriminatively trained part-based models</td>
</tr>
<tr>
<td>22</td>
<td>2-s2.0-77956877124</td>
<td>The Internet of Things: A survey</td>
</tr>
<tr>
<td>23</td>
<td>2-s2.0-77950537175</td>
<td>null</td>
</tr>
<tr>
<td>24</td>
<td>2-s2.0-77955309392</td>
<td>null</td>
</tr>
<tr>
<td>25</td>
<td>2-s2.0-70449533121</td>
<td>ABINIT: First-principles approach to material and nanosystem properties</td>
</tr>
<tr>
<td>26</td>
<td>2-s2.0-57149144228</td>
<td>A survey of affect recognition methods: Audio, visual, and spontaneous expressions</td>
</tr>
<tr>
<td>27</td>
<td>2-s2.0-77957244650</td>
<td>null</td>
</tr>
<tr>
<td>28</td>
<td>2-s2.0-67650711615</td>
<td>SOAP2: An improved ultrafast tool for short read alignment</td>
</tr>
<tr>
<td>29</td>
<td>2-s2.0-56449087483</td>
<td>null</td>
</tr>
<tr>
<td>30</td>
<td>2-s2.0-0036079158</td>
<td>null</td>
</tr>
<tr>
<td>31</td>
<td>2-s2.0-61349147303</td>
<td>null</td>
</tr>
<tr>
<td>32</td>
<td>2-s2.0-77950369345</td>
<td>Data clustering: 50 years beyond K-means</td>
</tr>
<tr>
<td>33</td>
<td>2-s2.0-73249139477</td>
<td>Object based image analysis for remote sensing</td>
</tr>
<tr>
<td>34</td>
<td>2-s2.0-77957243284</td>
<td>RDP3: A flexible and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fast computer program for analyzing recombination</td>
</tr>
<tr>
<td>----</td>
<td>----</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>35</td>
<td>2-s2.0-77749249761</td>
<td>null</td>
</tr>
<tr>
<td>36</td>
<td>2-s2.0-70449516548</td>
<td>Spectrum-efficient and scalable elastic optical path network: Architecture, benefits, and enabling technologies</td>
</tr>
<tr>
<td>37</td>
<td>2-s2.0-80054078476</td>
<td>Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega</td>
</tr>
<tr>
<td>38</td>
<td>2-s2.0-77951770756</td>
<td>BEDTools: A flexible suite of utilities for comparing genomic features</td>
</tr>
<tr>
<td>39</td>
<td>2-s2.0-37549003336</td>
<td>null</td>
</tr>
<tr>
<td>40</td>
<td>2-s2.0-77949587649</td>
<td>null</td>
</tr>
<tr>
<td>41</td>
<td>2-s2.0-3042535216</td>
<td>null</td>
</tr>
<tr>
<td>42</td>
<td>2-s2.0-77956031473</td>
<td>A survey on transfer learning</td>
</tr>
<tr>
<td>43</td>
<td>2-s2.0-77951298115</td>
<td>null</td>
</tr>
<tr>
<td>44</td>
<td>2-s2.0-74049108922</td>
<td>BLAST+: Architecture and applications</td>
</tr>
<tr>
<td>45</td>
<td>2-s2.0-77958110812</td>
<td>Conducting meta-analyses in R with the</td>
</tr>
<tr>
<td></td>
<td>Title</td>
<td>Author(s)</td>
</tr>
<tr>
<td>---</td>
<td>-----------------------------------------------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>46</td>
<td>metafor</td>
<td>null</td>
</tr>
<tr>
<td>47</td>
<td>Learning from imbalanced data</td>
<td>null</td>
</tr>
<tr>
<td>48</td>
<td>Networking named content</td>
<td>null</td>
</tr>
<tr>
<td>49</td>
<td>DBpedia - A crystallization point for the Web of Data</td>
<td>null</td>
</tr>
<tr>
<td>50</td>
<td>null</td>
<td>null</td>
</tr>
<tr>
<td>51</td>
<td>Picante: R tools for integrating phylogenies and ecology</td>
<td>null</td>
</tr>
<tr>
<td>52</td>
<td>Mean squared error: Lot it or leave it? A new look at signal fidelity measures</td>
<td>null</td>
</tr>
<tr>
<td>53</td>
<td>null</td>
<td>null</td>
</tr>
<tr>
<td>54</td>
<td>Recent advances and industrial applications of multilevel converters</td>
<td>null</td>
</tr>
<tr>
<td>55</td>
<td>Google's PageRank and beyond: The science of search engine rankings</td>
<td>null</td>
</tr>
<tr>
<td>56</td>
<td>null</td>
<td>null</td>
</tr>
<tr>
<td>57</td>
<td>null</td>
<td>null</td>
</tr>
<tr>
<td>58</td>
<td>The ORCA program system</td>
<td>null</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>59</td>
<td>2-s2.0-61849169018</td>
<td>null</td>
</tr>
<tr>
<td>60</td>
<td>2-s2.0-67349183350</td>
<td>FactSage thermochemical software and databases - recent developments</td>
</tr>
<tr>
<td>61</td>
<td>2-s2.0-63149159130</td>
<td>null</td>
</tr>
<tr>
<td>62</td>
<td>2-s2.0-77955426203</td>
<td>Evaluating color descriptors for object and scene recognition</td>
</tr>
<tr>
<td>63</td>
<td>2-s2.0-71049116435</td>
<td>null</td>
</tr>
<tr>
<td>64</td>
<td>2-s2.0-77955894071</td>
<td>METAL: Fast and efficient meta-analysis of genomewide association scans</td>
</tr>
<tr>
<td>65</td>
<td>2-s2.0-79551668253</td>
<td>FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data</td>
</tr>
<tr>
<td>66</td>
<td>2-s2.0-64549119687</td>
<td>null</td>
</tr>
<tr>
<td>67</td>
<td>2-s2.0-77952299957</td>
<td>Prodigal: Prokaryotic gene recognition and translation initiation site identification</td>
</tr>
<tr>
<td>68</td>
<td>2-s2.0-65749110333</td>
<td>null</td>
</tr>
<tr>
<td>69</td>
<td>2-s2.0-80051762104</td>
<td>Distributed optimization and statistical learning via</td>
</tr>
<tr>
<td>70</td>
<td>2-s2.0-67349139858</td>
<td>the alternating direction method of multipliers</td>
</tr>
<tr>
<td>71</td>
<td>2-s2.0-5294906084</td>
<td>LTE: The evolution of mobile broadband</td>
</tr>
<tr>
<td>72</td>
<td>2-s2.0-73849116467</td>
<td>null</td>
</tr>
<tr>
<td>73</td>
<td>2-s2.0-59649083826</td>
<td>Differential evolution algorithm with strategy adaptation for global numerical optimization</td>
</tr>
<tr>
<td>74</td>
<td>2-s2.0-0025183708</td>
<td>null</td>
</tr>
<tr>
<td>75</td>
<td>2-s2.0-33645712892</td>
<td>null</td>
</tr>
<tr>
<td>76</td>
<td>2-s2.0-79952709519</td>
<td>pROC: An open-source package for R and S+ to analyze and compare ROC curves</td>
</tr>
<tr>
<td>77</td>
<td>2-s2.0-79952003251</td>
<td>Differential evolution: A survey of the state-of-the-art</td>
</tr>
<tr>
<td>78</td>
<td>2-s2.0-65649120715</td>
<td>null</td>
</tr>
<tr>
<td>79</td>
<td>2-s2.0-67650099990</td>
<td>Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels</td>
</tr>
<tr>
<td>80</td>
<td>2-s2.0-70350642087</td>
<td>null</td>
</tr>
<tr>
<td>81</td>
<td>2-s2.0-84867293252</td>
<td>GenALEx 6.5</td>
</tr>
<tr>
<td>Page</td>
<td>DOI</td>
<td>Title</td>
</tr>
<tr>
<td>------</td>
<td>-----------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>82</td>
<td>2-s2.0-61749090884</td>
<td>Genetic analysis in Excel. Population genetic software for teaching and research-an update</td>
</tr>
<tr>
<td>83</td>
<td>2-s2.0-78049312324</td>
<td>Image super-resolution via sparse representation</td>
</tr>
<tr>
<td>84</td>
<td>2-s2.0-69549111057</td>
<td>Cutting-plane training of structural SVMs</td>
</tr>
<tr>
<td>85</td>
<td>2-s2.0-77951965920</td>
<td>Skyline: An open source document editor for creating and analyzing targeted proteomics experiments</td>
</tr>
<tr>
<td>86</td>
<td>2-s2.0-67049119331</td>
<td>Functional brain networks develop from a &quot;local to distributed&quot; organization</td>
</tr>
<tr>
<td>87</td>
<td>2-s2.0-69349090197</td>
<td>null</td>
</tr>
<tr>
<td>88</td>
<td>2-s2.0-65549111754</td>
<td>null</td>
</tr>
<tr>
<td>89</td>
<td>2-s2.0-58349111875</td>
<td>AmiGO: Online access to ontology and annotation data</td>
</tr>
<tr>
<td>90</td>
<td>2-s2.0-65649092976</td>
<td>null</td>
</tr>
<tr>
<td>91</td>
<td>2-s2.0-70349750047</td>
<td>null</td>
</tr>
<tr>
<td>92</td>
<td>2-s2.0-58449115775</td>
<td>null</td>
</tr>
<tr>
<td>93</td>
<td>2-s2.0-62249142056</td>
<td>null</td>
</tr>
<tr>
<td>94</td>
<td>2-s2.0-60249084757</td>
<td>null</td>
</tr>
<tr>
<td>95</td>
<td>2-s2.0-78649738324</td>
<td>Multi-cell MIMO cooperative networks: A new look at interference</td>
</tr>
<tr>
<td>96</td>
<td>2-s2.0-61949087310</td>
<td>Word sense disambiguation: A survey</td>
</tr>
<tr>
<td>97</td>
<td>2-s2.0-70349626031</td>
<td>Adaptive particle swarm optimization</td>
</tr>
<tr>
<td>98</td>
<td>2-s2.0-70350769233</td>
<td>null</td>
</tr>
<tr>
<td>99</td>
<td>2-s2.0-66449084442</td>
<td>null</td>
</tr>
<tr>
<td>100</td>
<td>2-s2.0-70350345210</td>
<td>null</td>
</tr>
</tbody>
</table>

*Table 3 The Eigenvector rankings*

After analyzing the above results we can observe that the rankings are quite similar for the high positions (1-40). However the ranking deviation starts growing while we move towards the lower positions. In Chart 1 we present the relation between the ranking deviation and the according position in the list of results. Sometimes we observe a difference of more than 15 positions and although we have not implemented the eigentrust algorithm, these papers are probably candidates for further research. It is almost certain that the deviation will keep growing even more for the lowest positions.
5.12. The peer review problem

The results that arise from a survey performed by NATURE (Abbott, et al., 2010) speak for themselves. No researcher’s career and future prosperity should be judged by a number. The time he/she spends or the effort to catch up with all their duties cannot be quantified. However, scientists believe that when their work needs to be evaluated by an external factor, this number is what represents their work. In the prementioned survey a 75% of all participants say this number approach is used by their institutions for hiring decisions. However, since we cannot change this approach we should make sure that this approach is as fair as possible. In the previous chapters we have talked about what we can do to avoid colluders and inadmissible practices. But when it comes to peer reviewing (as we have already see in chapter 4.2) things are different.

In this section we present a theoretical model that may help and contribute to a fairer judgment by peer reviewers. A new term is introduced, PeerNtrust. PeerNtrust is an algorithm based on the eigentrust algorithm. It is actually the eigentrust algorithm, where instead of peers for a p2p network we have peer reviewers. The peer reviewing community can also be considered as a large network of peers. Following the methodology we
presented in chapters 5.2 - 5.11 we can spot all the publications that are considered suspicious. After making sure that these papers are manipulated we can spot the peer reviewer – reviewers behind it. Consequently, that would mean that the peer reviewer is also responsible for this publication. Of course the same methodology can be followed in order to spot the peer reviewers that are responsible and deliver good results.

Since a peer reviewer has positive results, he can be assigned with a high trust value. Peer reviewers that have a history of good publications can be considered our pre-trusted peers. After each review, depending on the quality of the fore coming publication, their local trust value can either increase or decrease. We can now move on through an iterative process that would finally assign some reliable trust values to all peer reviewers and the journals that they work for.

This would have a two sided effect. Peer reviewers will start being more responsible and will not adjust their judgment to the policy their journal wants them to adopt. The only obvious problem is for the pre trusted peers. As we have already mentioned, a malicious pre trusted peer can lead to highly incorrect and unreliable results. However, since the pre-trusted peers will not be many they can be controlled by an external subjective source that would evaluate their work every now and then. This methodology may look good on paper, but it is not certain that it will actually succeed. It is another part of this dissertation on which further work is needed and it is be done in the recent future.
6. Conclusion

In this dissertation we have presented all the bibliometrics factors that are used widely along with their characteristics, advantages and disadvantages. We have proceeded with a deeper research on the problem that the academic community and the researchers are facing, suggesting that no number can replace the original work of each researcher. However, since these numbers are widely used we aim to make these metrics more reliable, robust and trustworthy.

We have presented a method to spot colluders and unacceptable publication practices in the academic community. Our application is able to scrape open data available by the Scopus, one of the biggest online databases of abstracts and citations of peer-reviewed literature online, turn them into XML files, and compute the PageRank and the eigenvector centrality of each publication and in the near future the local trust value of each author and journal subsequently. We also suggest a solution to malicious peer reviews by implementing the EigenTrust algorithm in the network of peer reviewers. The latter stands on a theoretical base but we believe that it can be developed in a functional prototype.


Higgs, P., 2013. [Interview] (December 2013).

Hirsch, J., 2005. An index to quantify an individual’s scientific research output. 15 November, 102(46), p. 16571.


Labbe, C., 2010. Ike Antkare one of the great stars in the scientific firmament, s.l.: s.n.


8. Appendix

8.1. Deixto Source Code – XML Patterns – XML results

Scrape_details.pl

#!/usr/bin/perl

use strict;
use warnings;
use utf8;
use Encode;
use DEiXToBot;
use Spreadsheet::Write;
use WWW::Selenium;
#use Selenium::Remote::Driver;
use File::Temp qw/ tempfile /;

# INITIALIZE - CREATE/CONFIGURE BOT

my $agent = DEiXToBot->new( os_charset => 'utf8',
                          nice => 0,
                          sleep => "2..6" );

$agent->parse_head(0); # don't initialize response headers

# set the depth of the page stack to 1, to avoid eating up memory
$agent->stack_depth(1);

$agent->quiet(1); # turn off warnings
$agent->agent_alias('Linux Mozilla');

my $writer;
my @urls = ();
my %hash;

opendir(DIR, '/home/user/scopus') or die "can't open scopus directory: $!";

my %down;
my @files = readdir(DIR);
for my $name (@files)
{
    $name=~m#(.+?)\.xml#;
    print $name, "\n";
    $down{$1}=1;
}

use Spreadsheet::Read;

my $ref = ReadData("scopus_detail_urls.xls");

if ( not defined $ref ) {
    die "Could not read file.\n";
}

my @rows = Spreadsheet::Read::rows ($ref->[1]);
my $last_row = $ref->[1]{maxrow};
my $last_col = $ref->[1]{maxcol};

my $i=0;
while ($i < $last_row) {
    #if ($i++ < 10) { next }
    $rows[$i][0]=~m#eid=(.+?)&origin#;
    if (exists $down{$1}) { $i++;next }
    push @urls, $rows[$i][0];
    $rows[$i][1]=~s#®##;
    $hash{$rows[$i][0]} = $rows[$i][1];
    $i++;
}

#open my $th, '<', 'scopus_detail_urls.txt';
#while(my $line = <$th>) {
    #if ($i++ < 20) { next }
    # chomp $line;
    # my @parts = split("\t",$line);
    # push @urls, $parts[0];
    # $hash{$line}=$parts[1];
    # print "$i) $line\n";
    # $i++;
    #}
    #close $th;
    #print scalar(@urls),"\n";
    #exit;
    my $sel = WWW::Selenium->new( host => "localhost",
        'port' => 4444,
        'browser' => "*firefox",
        browser_url => "http://www.scopus.com",
    );
    $sel->start;
$sel->set_timeout(120000);

my @array;
my $cnt = 1;
for my $url (@urls) {
    print "$cnt) $url\n"
    $url =~ m#eid=(.+?)&origin#;
    my $eid = $1;

    my $output = IO::File->new("/home/user/scopus/$eid.xml");
    $writer = XML::Writer->new(
        OUTPUT => $output,
        DATA_MODE => 1,
        DATA_INDENT => 4,
        ENCODING => 'utf-8'
    );
    $writer->xmlDecl('UTF-8');
    $writer->startTag("publication");

    $writer->dataElement("url", $url);
    $writer->dataElement("eid", $eid);
    $hash{$url} = "";
    eval {$writer->dataElement("title", $hash{$url})};
    if ($@) {
        die $hash{$url}, "\n";
        next;
    }
}

eval {
    $sel->open($url);
```perl
#$sel->wait_for_page_to_load(120000);

};
if ($@) {
    sleep(120);
    $sel->open($url);
    #$sel->wait_for_page_to_load(120000);
}

$sel->pause(8000);

my $content = $sel->get_html_source(); #get_html_source();

my ($fh,$name); # create a temporary file containing the page's source code
($fh,$name) = tempfile();
print $fh $content;
close $fh;

$agent->get("file://$name"); # load the temporary file/page with the DEiXToBot agent using the file:// scheme

unlink $name; # delete the temporary file, it is not needed any more

if (! $agent->response->is_success) {
    die "Failed to get the page."
}

$agent->build_dom();

$agent->load_pattern('affiliations.xml');

$agent->extract_content();
```
my @row;
if (! $agent->hits) {
    print "Could not find affiliations";
}
else {
    my @records = @{$agent->records};
    $writer->startTag("affiliations");
    for my $record (@records) {
        my @rec = @$record;
        $writer->dataElement("affiliation", $rec[1], 'letter'=> $rec[0]);
    }
    $writer->endTag("affiliations");
}

$agent->load_pattern('author_a.xml');

$agent->extract_content();

if (! $agent->hits) {
    die "Could not find any authors!\n";
}
else {
    my @records = @{$agent->records};
    $writer->startTag("authors");
    for my $record (@records) {
        my @rec = @$record;
        $writer->dataElement("author", $rec[1], 'author_id' => $rec[0], 'letter'=> $rec[2]);
    }
}
$writer->endTag("authors");
}

$agent->load_pattern('reflist.xml');

$agent->extract_content();

if (! $agent->hits) {
    print "Could not find any references\n";
}

else {
    my @records = @{$agent->records};
    $writer->startTag("references");
    for my $record (@records) {
        my @rec = @$record;
        $rec[0]=~m#eid=(.+?)&origin#;
        my $refid=$1;
        $writer->dataElement("ref", $rec[0], 'eid'=>$refid);
    }
    $writer->endTag("references");
}

$agent->load_pattern('view_all_citations.xml');

$agent->extract_content();

if (! $agent->hits) {
    print "Could not find the link to view all citations\n";
}
else {
    my @records = @{$agent->records};
    my @record = @{$records[0]};
    $writer->dataElement("all_citations_url", $record[0]);
    scrape_citations($record[0]);
}

my $pause = int(rand(30));
print "Gonna sleep for $pause seconds\n";
sleep( $pause );

$writer->endTag('publication');
$writer->end();
$output->close();

$cnt++;print "$cnt\n";
#last;
#if ($cnt>100) { print "$cnt: have to break"; last; }
}

$sel->stop;

sub scrape_citations {
    my $curl = shift;
    my $original=$curl;
    my $offset=1;
    while(1){
        #$curl=$original."&offset=$offset";
        if ($offset == 1) {
            print $curl, "\n";
            eval {
                
            }
        } else { 
            print $curl, "\n";
            eval {
                
            }
        }
        #print $curl, "\n";
        $offset++;
    }#while (1)
}
$sel->open($curl);
#$sel->wait_for_page_to_load(120000);

};
if ($@) {
    print $@
    sleep(120);
    $sel->open($curl);
    #$sel->wait_for_page_to_load(120000);
}
}
else {
    # <a class="jsEnabled nextBtn cursorPointer" href="javascript:setSelectedLink('NextPageButton');" title="Next page" alt="Next page"></a>
    $sel->click('xpath=//a[@title="Next page"]');
}

$sel->pause(8000);
if ($offset == 1) {
    $sel->select('xpath=//select[@name="resultsPerPage"]',200);
    $sel->pause(8000);
}

my $content = $sel->get_html_source();
open my $sh,">","source.html";
print $sh $content;
close $sh;
my $resultsCount=200;
if ($content=~m#resultsCount">\s*(.+?)\s*</span)#mi) {
    $resultsCount = $1;
    $resultsCount=~s#,##g;
    print $resultsCount,"\n";
my ($fh,$name); # create a temporary file containing the page's source code
($fh,$name) = tempfile();
print $fh $content;
close $fh;

$agent->get("file://$name"); # load the temporary file/page with the DEiXTToBot agent using the file:// scheme
unlink $name; # delete the temporary file, it is not needed any more

if (! $agent->response->is_success) {
    die "Failed to get the page."
}

$agent->build_dom();

$agent->load_pattern('scopus_citations_details.xml');

$agent->extract_content();

if (! $agent->hits) {
    print "Could not find citations";
} else {
    my @records = @{$agent->records};
    $writer->startTag("citations");
for my $record (@records) {
    my @rec = @$record;
    $writer->startTag("citation");
    $writer->dataElement("eid", $rec[0]);
    $writer->dataElement("title", $rec[1]);
    $writer->startTag("authors");
    for my $i (1..7) {
        if ($rec[$i*2]) {
            $writer->dataElement("author","$rec[1+$i*2],'author_id' => $rec[$i*2]);
        }
    }
    $writer->endTag("authors");
    $writer->endTag("citation");
}
$writer->endTag("citations");

$offset+=200;
if ($resultsCount and ($offset > $resultsCount or $offset>=2000)) {
    last;
}
print "Offset: $offset\n";
sleep(5);
}

Scopus_urls.xml

<?xml version="1.0" encoding="UTF-8"?>

<Pattern><Node tag="DIV" stateIndex="grayed" IsRoot="true"><Node tag="LABEL" stateIndex="grayed"><Node tag="TEXT" stateIndex="grayed" regexpr="Document"/></Node><Node tag="SPAN" stateIndex="grayed"><Node tag="A" stateIndex="checked" regexpr=".+record/display.url.+">$<Node tag="TEXT" stateIndex="checked"/></Node></Node></Pattern>
Scopus_citations_details.xml

<?xml version="1.0" encoding="UTF-8"?>

publication

<url>http://www.scopus.com/record/display.url?eid=2-s2.0-77952247119&amp;origin=resultslist</url>

<title>Converter systems for fuel cells in the medium power range-a comparative study</title>

<affiliations>
<affiliation letter="a">REpower Systems AG, 22297 Hamburg, Germany</affiliation>  

<affiliation letter="b">Institute for Power Electronics and Electrical Drives, Christian-Albrechts-University of Kiel, 24143 Kiel, Germany</affiliation>  

</affiliations>  

<authors>  

<author author_id="16068987900" letter="a">Mohr, M.</author>  

<author author_id="24921424700" letter="b">Franke, W.T.</author>  

<author author_id="35249429200" letter="b">Wittig, B.</author>  

<author author_id="7101777659" letter="b">Fuchs, F.W.</author>  

</authors>  

<references>  

<ref eid="2-s2.0-4544324564">http://www.scopus.com/record/display.url?eid=2-s2.0-4544324564&amp;origin=reflist</ref>  

<ref eid="2-s2.0-33645567088">http://www.scopus.com/record/display.url?eid=2-s2.0-33645567088&amp;origin=reflist</ref>  

<ref eid="2-s2.0-0038417535">http://www.scopus.com/record/display.url?eid=2-s2.0-0038417535&amp;origin=reflist</ref>  

<ref eid="2-s2.0-8744242098">http://www.scopus.com/record/display.url?eid=2-s2.0-8744242098&amp;origin=reflist</ref>  

</references>  

<all_citations_url>http://www.scopus.com/search/submit/citedby.url?eid=2-s2.0-77952247119&amp;src=s&amp;origin=recordpage</all_citations_url>  

<citations>  

<citation>  

<ref id="2-s2.0-84897741547">http://www.scopus.com/record/display.url?eid=2-s2.0-84897741547&amp;origin=reflist</ref>  

<title>Operation modes analysis and limitation for diode-assisted buck-boost voltage source inverter with small voltage vector</title>  

<authors>  

<author author_id="56060550600">Zhang, Y.</author>  

<author author_id="23100565700">Liu, J.</author>  

</citation>  

</citations>
<author author_id="55223031200">Ma, X.</author>
<author author_id="56059937900">Feng, J.</author>
</authors>
</citation>

<author author_id="55437871800">Kollimalla, S.K.</author>
<author author_id="35607256300">Mishra, M.K.</author>
<author author_id="25637675100">Lakshmi, N.N.</author>
</authors>
</citation>

<author author_id="26642998300">Li, X.</author>
<author author_id="56261327100">Bhat, A.K.S.</author>
</authors>
</citation>

<author author_id="35231394300">Tang, Y.</author>
<author author_id="55762622200">Xie, S.</author>
</authors>
</citation>
8.2. JAVA Source Code

Main.java

```java
import java.io.IOException;
import java.text.ParseException;
import java.util.List;
import javax.xml.parsers.ParserConfigurationException;
import org.xml.sax.SAXException;

public class Main {
    public static void main(String... args) throws SAXException, IOException, ParserConfigurationException, ParseException{
        new ApplicationFrame();
    }
}
```
public class Publication implements Comparable<Publication>, Serializable{
    private String eid;
    private String url;
    private String title;
    private Map<String,Author> authors;
    private Map<String,Publication> references;
    private Map<String,Publication> citations;

    public Publication(){
        authors = new HashMap<String,Author>();
        references = new HashMap<String, Publication>();
        citations = new HashMap<String, Publication>();
    }

    public Publication(String id, String urlText){
        eid = id;
        url = urlText;
        authors = new HashMap<String,Author>();
        references = new HashMap<String, Publication>();
        citations = new HashMap<String, Publication>();
    }

    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public String getEid() {
        return eid;
    }

    public void setEid(String eid) {
        this.eid = eid;
    }

    public String getUrl() {
        return url;
    }

    public void setUrl(String url) {
        this.url = url;
    }
}
public Author getAuthor(String key){
    Author a = null;
    a = authors.get(key);
    return a;
}

public void addAuthor(Author a){
    if(!authors.containsKey(a.getId())){
        authors.put(a.getId(), a);
    }
}

public void addReference(Publication pub){
    if(!references.containsKey(pub.getEid())){
        references.put(pub.getEid(), pub);
    }
}

public void addCitation(Publication pub){
    if(!citations.containsKey(pub.getEid())){
        citations.put(pub.getEid(), pub);
    }
}

public Set<String> getReferencedAuthorsIds(){
    Set<String> referencedAuthorIds = new HashSet<String>();
    for(String key : references.keySet()){
        Publication pub = references.get(key);
        referencedAuthorIds.addAll(pub.authors.keySet());
    }
    return referencedAuthorIds;
}

public Set<String> getCitedAuthorsIds(){
    Set<String> citedAuthorIds = new HashSet<String>();
    for(String key : citations.keySet()){
        Publication pub = citations.get(key);
        citedAuthorIds.addAll(pub.authors.keySet());
    }
    return citedAuthorIds;
}

public Map<String, Publication> getCitations() {
    return citations;
}

public Collection<Author> getAuthors(){
    return authors.values();
}

public void printAuthors(){
for (Author anAuthor : getAuthors()) {
    System.out.print(anAuthor.getName() + " ", " );
}
System.out.println();

@Override
public String toString() {
    return eid;
}

public String toStringFull() {
    StringBuilder sb = new StringBuilder();
    sb.append(title).append("\n");
    sb.append("Authors: ");
    for (Author a : getAuthors()) {
        sb.append(a.getName()).append(", ");
    }
    sb.append("\n");
    sb.append("References: ").append("\n");
    int ref = 1;
    for (String refid : references.keySet()){
        sb.append(" [+ref+] : "+refid).append("\n");
        ref++;
    }
    sb.append("Cited By: ").append("\n");
    int cit = 1;
    for (String citId : citations.keySet()){
        Publication citation = citations.get(citId);
        sb.append(" <+cit+> : "+citation.getTitle()).append("\n");
        cit++;
    }
    return sb.toString();
}

@Override
public int compareTo(Publication other) {
    if (this.eid.equals(other.eid)) return 0;
    else return 1;
}

public boolean equals(Object other) {
    if (other instanceof Publication) {
        Publication otherPub = (Publication)other;
        if (this.eid.equals(otherPub.eid)) return true;
    }
    return false;
}

public Map<String, Publication> getReferences() {
    return references;
}

public int hashCode(){}

```java
return eid.hashCode();
}
}

Author.java

import java.io.Serializable;

public class Author implements Serializable{

private String id;
private String name;
private String affiliation;

public Author(String id, String name, String affiliation) {
    this.id = id;
    this.name = name;
    this.affiliation = affiliation;
}

public Author(String id) {
    this.id = id;
    this.name = "";
    this.affiliation = "";
}

public Author() {
    // TODO Auto-generated constructor stub
}

public String getId() {
    return id;
}

public void setId(String id) {
    this.id = id;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getAffiliation() {
    return affiliation;
}

public void setAffiliation(String affiliation) {
    this.affiliation = affiliation;
}
```
@Override
    public String toString() {
        return "Author [id=", id, ", name=", name, ", affiliation=
            + affiliation + "]";
    }
}

XMLReader.java

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NamedNodeMap;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

public class XMLReader {

    public Map<String,Publication> getPublications(File folderOfXMLs){
        Map<String,Publication> publicationList = new HashMap<String,Publication>();
        //Here is the file loader
        try{
            File[] listOffiles = folderOfXMLs.listFiles();
            for (int i = 0; i < listOffiles.length; i++) {
                File f = listOffiles[i];
                if (f.isFile() && f.get AbsolutePath().endsWith(".xml")) {
                    System.out.println("Parsing xml file "+(i+1)+" from "+listOffiles.length);
                    Publication publication = readXML(f.getAbsolutePath());
                    publicationList.put(publication.getEid(),publication);
                }
            }
        }
    }
}
catch(Exception e){
    e.printStackTrace();
}
return publicationList;

//Here is the Document Builder for our parser
public Publication readXML(String fileName) throws SAXException, IOException, ParserConfigurationException{
    DocumentBuilderFactory factory =
        DocumentBuilderFactory.newInstance();
    DocumentBuilder builder = factory.newDocumentBuilder();
    Document document = builder.parse(new FileInputStream(fileName));
    NodeList publicationNodes =
        document.getElementsByTagName("publication");

    Publication pub = new Publication();
    //Now we start with the multiple iteration scheme
    for (int i = 0; i < publicationNodes.getLength(); i++) {
        Node publicationNode = publicationNodes.item(i);
        if (publicationNode instanceof Element) {
            Element pubElement = (Element)publicationNode;
            NodeList childNodes = publicationNode.getChildNodes();
            for (int j = 0; j < childNodes.getLength(); j++) {
                Node cNode = childNodes.item(j);
                String nodeName = cNode.getNodeName();
                if(nodeName.equals("url")){
                    String content =
                        cNode.getLastChild().getTextContent().trim();
                    pub.setUrl(content);
                }
                else if(nodeName.equals("eid")){
                    String content =
                        cNode.getLastChild().getTextContent().trim();
                    pub.setEid(content);
                } else if(nodeName.equals("title")){
                    Node text = cNode.getLastChild();
                    if(text!= null){
                        String content =
                            text.getTextContent().trim();
                        pub.setTitle(content);
                    } else
                        pub.setTitle(" ");
                } else if(nodeName.equals("authors")){

cNode.getChildNodes();
k++;
authorsNodes.item(k);
authorNode.getAttributes();

atr.getNamedItem("author_id").getNodeValue();
authorNode.getLastChild().getTextContent().trim();
Author(authorId, authorName,"");

pub.addAuthor(a);

} else if(nodeName.equals("references")){
NodeList referenceNodes = cNode.getChildNodes();
r<referenceNodes.getLength(); r++;
referenceNodes.item(r);
referenceNode.getAttributes();

atr.getNamedItem("eid").getNodeValue();
Publication();

pub.addReference(ref);

}
}
else if(nodeName.equals("citations")){
NodeList citationsNodes = cNode.getChildNodes();
k<citationsNodes.getLength(); k++;
citationsNodes.item(k);
Publication();

for(int k=0; k<authorsNodes.getLength(); k++){

Node authorNode = authorsNodes.item(k);
if(authorNode != null) {
NamedNodeMap atr = authorNode.getAttributes();
if(atr != null){
String authorId = atr.getNamedItem("author_id").getNodeValue();
String authorName = authorNode.getLastChild().getTextContent().trim();
Author a = new Author(authorId, authorName,"");

pub.addAuthor(a);
}
}
else if(nodeName.equals("references")){
NodeList referenceNodes = cNode.getChildNodes();
r<referenceNodes.getLength(); r++;
referenceNodes.item(r);
referenceNode.getAttributes();

atr.getNamedItem("eid").getNodeValue();
Publication();

pub.addReference(ref);

}
}
else if(nodeName.equals("citations")){
NodeList citationsNodes = cNode.getChildNodes();
k<citationsNodes.getLength(); k++;
citationsNodes.item(k);
Publication();

for(int k=0; k<authorsNodes.getLength(); k++){

Node authorNode = authorsNodes.item(k);
if(authorNode != null) {
NamedNodeMap atr = authorNode.getAttributes();
if(atr != null){
String authorId = atr.getNamedItem("author_id").getNodeValue();
String authorName = authorNode.getLastChild().getTextContent().trim();
Author a = new Author(authorId, authorName,"");

pub.addAuthor(a);
}
}
}

else if(nodeName.equals("references")){
NodeList referenceNodes = cNode.getChildNodes();
r<referenceNodes.getLength(); r++;
referenceNodes.item(r);
referenceNode.getAttributes();

atr.getNamedItem("eid").getNodeValue();
Publication();

pub.addReference(ref);

}
}
else if(nodeName.equals("citations")){
NodeList citationsNodes = cNode.getChildNodes();
k<citationsNodes.getLength(); k++;
citationsNodes.item(k);
Publication();

for(int k=0; k<authorsNodes.getLength(); k++){

Node authorNode = authorsNodes.item(k);
if(authorNode != null) {
NamedNodeMap atr = authorNode.getAttributes();
if(atr != null){
String authorId = atr.getNamedItem("author_id").getNodeValue();
String authorName = authorNode.getLastChild().getTextContent().trim();
Author a = new Author(authorId, authorName,"");

pub.addAuthor(a);
}
}

else if(nodeName.equals("references")){
NodeList referenceNodes = cNode.getChildNodes();
r<referenceNodes.getLength(); r++;
referenceNodes.item(r);
referenceNode.getAttributes();

atr.getNamedItem("eid").getNodeValue();
Publication();

pub.addReference(ref);

}
}
else if(nodeName.equals("citations")){
NodeList citationsNodes = cNode.getChildNodes();
k<citationsNodes.getLength(); k++;
citationsNodes.item(k);
Publication();

for(int k=0; k<authorsNodes.getLength(); k++){

Node authorNode = authorsNodes.item(k);
if(authorNode != null) {
NamedNodeMap atr = authorNode.getAttributes();
if(atr != null){
String authorId = atr.getNamedItem("author_id").getNodeValue();
String authorName = authorNode.getLastChild().getTextContent().trim();
Author a = new Author(authorId, authorName,"");

pub.addAuthor(a);
}
}

else if(nodeName.equals("references")){
NodeList referenceNodes = cNode.getChildNodes();
r<referenceNodes.getLength(); r++;
referenceNodes.item(r);
referenceNode.getAttributes();

atr.getNamedItem("eid").getNodeValue();
Publication();

pub.addReference(ref);

}
}

else if(nodeName.equals("citations")){
NodeList citationsNodes = cNode.getChildNodes();
k<citationsNodes.getLength(); k++;
citationsNodes.item(k);
Publication();

for(int k=0; k<authorsNodes.getLength(); k++){

Node authorNode = authorsNodes.item(k);
if(authorNode != null) {
NamedNodeMap atr = authorNode.getAttributes();
if(atr != null){
String authorId = atr.getNamedItem("author_id").getNodeValue();
String authorName = authorNode.getLastChild().getTextContent().trim();
Author a = new Author(authorId, authorName,"");

pub.addAuthor(a);
}
}
}

else if(nodeName.equals("references")){
NodeList referenceNodes = cNode.getChildNodes();
r<referenceNodes.getLength(); r++;
referenceNodes.item(r);
referenceNode.getAttributes();

atr.getNamedItem("eid").getNodeValue();
Publication();

pub.addReference(ref);

}
}

else if(nodeName.equals("citations")){
NodeList citationsNodes = cNode.getChildNodes();
k<citationsNodes.getLength(); k++;
citationsNodes.item(k);
Publication();

for(int k=0; k<authorsNodes.getLength(); k++){

Node authorNode = authorsNodes.item(k);
if(authorNode != null) {
NamedNodeMap atr = authorNode.getAttributes();
if(atr != null){
String authorId = atr.getNamedItem("author_id").getNodeValue();
String authorName = authorNode.getLastChild().getTextContent().trim();
Author a = new Author(authorId, authorName,"");

pub.addAuthor(a);
}
}
}

else if(nodeName.equals("references")){
NodeList referenceNodes = cNode.getChildNodes();
r<referenceNodes.getLength(); r++;
referenceNodes.item(r);
referenceNode.getAttributes();

atr.getNamedItem("eid").getNodeValue();
Publication();

pub.addReference(ref);

}
}

else if(nodeName.equals("citations")){
NodeList citationsNodes = cNode.getChildNodes();
k<citationsNodes.getLength(); k++;
citationsNodes.item(k);
Publication();

for(int k=0; k<authorsNodes.getLength(); k++){

Node authorNode = authorsNodes.item(k);
if(authorNode != null) {
NamedNodeMap atr = authorNode.getAttributes();
if(atr != null){
String authorId = atr.getNamedItem("author_id").getNodeValue();
String authorName = authorNode.getLastChild().getTextContent().trim();
Author a = new Author(authorId, authorName,"");

pub.addAuthor(a);
}
}

else if(nodeName.equals("references")){
NodeList referenceNodes = cNode.getChildNodes();
r<referenceNodes.getLength(); r++;
referenceNodes.item(r);
referenceNode.getAttributes();

atr.getNamedItem("eid").getNodeValue();
Publication();

pub.addReference(ref);

}
}

else if(nodeName.equals("citations")){
NodeList citationsNodes = cNode.getChildNodes();
k<citationsNodes.getLength(); k++;
citationsNodes.item(k);
Publication();

for(int k=0; k<authorsNodes.getLength(); k++){

Node authorNode = authorsNodes.item(k);
if(authorNode != null) {
NamedNodeMap atr = authorNode.getAttributes();
if(atr != null){
String authorId = atr.getNamedItem("author_id").getNodeValue();
String authorName = authorNode.getLastChild().getTextContent().trim();
Author a = new Author(authorId, authorName,"");

pub.addAuthor(a);
}
}

else if(nodeName.equals("references")){
NodeList referenceNodes = cNode.getChildNodes();
r<referenceNodes.getLength(); r++;
referenceNodes.item(r);
referenceNode.getAttributes();

atr.getNamedItem("eid").getNodeValue();
Publication();

pub.addReference(ref);

}
}

else if(nodeName.equals("citations")){
NodeList citationsNodes = cNode.getChildNodes();
k<citationsNodes.getLength(); k++;
citationsNodes.item(k);
Publication();

for(int k=0; k<authorsNodes.getLength(); k++){

Node authorNode = authorsNodes.item(k);
if(authorNode != null) {
NamedNodeMap atr = authorNode.getAttributes();
if(atr != null){
String authorId = atr.getNamedItem("author_id").getNodeValue();
String authorName = authorNode.getLastChild().getTextContent().trim();
Author a = new Author(authorId, authorName,"");

pub.addAuthor(a);
}
}

else if(nodeName.equals("references")){
NodeList referenceNodes = cNode.getChildNodes();
r<referenceNodes.getLength(); r++;
referenceNodes.item(r);
referenceNode.getAttributes();

atr.getNamedItem("eid").getNodeValue();
Publication();

pub.addReference(ref);

}
}

else if(nodeName.equals("citations")){
NodeList citationsNodes = cNode.getChildNodes();
k<citationsNodes.getLength(); k++;
citationsNodes.item(k);
Publication();

for(int k=0; k<authorsNodes.getLength(); k++){

Node authorNode = authorsNodes.item(k);
if(authorNode != null) {
NamedNodeMap atr = authorNode.getAttributes();
if(atr != null){
String authorId = atr.getNamedItem("author_id").getNodeValue();
String authorName = authorNode.getLastChild().getTextContent().trim();
Author a = new Author(authorId, authorName,"");

pub.addAuthor(a);
}
}

else if(nodeName.equals("references")){
NodeList referenceNodes = cNode.getChildNodes();
r<referenceNodes.getLength(); r++;
referenceNodes.item(r);
referenceNode.getAttributes();

atr.getNamedItem("eid").getNodeValue();
Publication();

pub.addReference(ref);

}
}

if (citationNode instanceof Element) {
    Element citationElement = (Element) citationNode;
    NodeList citationChildNodes = citationElement.getChildNodes();
    for (int c = 0; c < citationChildNodes.getLength(); c++) {
        Node citationChildNode = citationChildNodes.item(c);
        String citationChildNodeName = citationChildNode.getNodeName();

        if (citationChildNodeName.equals("eid")) {
            String content = citationChildNode.getLastChild().getTextContent().trim();
            citation.setEid(content);
        } else if (citationChildNodeName.equals("title")) {
            String content = citationChildNode.getLastChild().getTextContent().trim();
            citation.setTitle(content);
        } else if (citationChildNodeName.equals("authors")) {
            NodeList citationAuthorsNodes = citationChildNode.getChildNodes();
            for (int a = 0; a < citationAuthorsNodes.getLength(); a++) {
                Node citationAuthorNode = citationAuthorsNodes.item(a);
                if (citationAuthorNode != null) {
                    NamedNodeMap atr = citationAuthorNode.getAttributes();
                    if (atr != null) {
                        String authorId = atr.getNamedItem("author_id").getNodeValue();
                        String authorName = citationAuthorNode.getLastChild().getTextContent().trim();
                        Author citAuthor = new Author(authorId, authorName, "");
                        citation.addAuthor(citAuthor);
                        pub.addCitation(citation);}
            }
        }
    }
}
PublicationManager.java

import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.Serializable;
import java.util.ArrayList;
import java.util.Collection;
import java.util.Collections;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Set;
import edu.uci.ics.jung.algorithms.cluster.WeakComponentClusterer;
import edu.uci.ics.jung.algorithms.filters.FilterUtils;
import edu.uci.ics.jung.algorithms.scoring.EigenvectorCentrality;
import edu.uci.ics.jung.algorithms.scoring.PageRank;
import edu.uci.ics.jung.graph.util.EdgeType;

public class PublicationManager implements Serializable {
    private MyGraph publicationGraph;
    private Map<String,Publication> publications;

    public PublicationManager(File folderOfXMLs){
        XMLReader reader = new XMLReader();
        publications = reader.getPublications(folderOfXMLs);
        createGraph();
        //publicationGraph.printNodeAndDegree();
    }
}
//new GraphVisualizer(publicationGraph);

WeakComponentClusterer<Publication,MyEdge> wcc = new
WeakComponentClusterer<Publication,MyEdge>();
Collection<MyGraph> ccs = FilterUtils.createAllInducedSubgraphs(wcc.transform(publicationGraph),publicationGraph);
System.out.println("Number of Components = "+ccs.size());
}

public void createGraph(){
    publicationGraph = new MyGraph();
    for(String pubId : publications.keySet()){ 
        Publication pub = publications.get(pubId);
        publicationGraph.addVertex(pub);
        Map<String,Publication> references = pub.getReferences();
        for(String refid : references.keySet()) {
            Publication reference = references.get(refid);
            publicationGraph.addVertex(reference);
            MyEdge edge = new MyEdge(pub, reference);
            publicationGraph.addEdge(edge, pub, reference,
            EdgeType.DIRECTED);
        }
        Map<String, Publication> citations = pub.getCitations();
        for(String citId : citations.keySet()){
            Publication citation = citations.get(citId);
            publicationGraph.addVertex(citation);
            MyEdge edge = new MyEdge(citation, pub);
            publicationGraph.addEdge(edge, citation, pub,
            EdgeType.DIRECTED);
        }
    }
}

public void printAllPublications(){
    for(Publication pub : publications.values()){ 
        System.out.println(pub);
    }
}

public void savePublicationManagerLocally(){
    try {
        File outDir = new File("serializedPublications");
        if(outDir.exists() == false) {
            outDir.mkdir();
        }
        String fileName = "serializedPublications" + File.separator + "publications.ser";
        FileOutputStream fos = new FileOutputStream(fileName);
        ObjectOutputStream oos = new ObjectOutputStream(fos);
        oos.writeObject(this);
        oos.flush();
        oos.close();
        fos.close();
    } catch (IOException e) { 
        
    } 
}
```java
public static PublicationManager loadPublicationManager()
{
    PublicationManager manager = null;
    try {
        FileInputStream fis = new FileInputStream("serializedPublications" + File.separator + "publications.ser");
        ObjectInputStream ois = new ObjectInputStream(fis);
        manager = (PublicationManager) ois.readObject();
        fis.close();
        ois.close();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    }
    return manager;
}

public List<PageRankOfNode> calculatePageRank(double dampingFactor) {
    List<PageRankOfNode> pageRanksPerNode = new ArrayList<PageRankOfNode> ();
    PageRank<Publication, MyEdge> pageRank = new PageRank<Publication, MyEdge> (publicationGraph, dampingFactor);
    pageRank.evaluate();
    for (Publication node : publicationGraph.getVertices()) {
        PageRankOfNode pageRankOfThisNode = new PageRankOfNode (node.getEid(), node.getTitle(), pageRank.getVertexScore(node));
        pageRanksPerNode.add(pageRankOfThisNode);
    }
    return pageRanksPerNode;
}

public void printTopPageRanks(List<PageRankOfNode> pageRanksAndNodes, int howMany)
{
    Collections.sort(pageRanksAndNodes);
    for (int i = 0; i < howMany; i++) {
        PageRankOfNode prof = pageRanksAndNodes.get(i);
        String nodeId = prof.getNodeId();
        String nodeTitle = prof.getNodeTitle();
        Publication pub = publications.get(nodeId);
        double pageRankForThisPub = prof.getPageRank();
        System.out.format("%150s%30s%30s
| Title: %s| " + nodeTitle+"|"," -Eid: |"+ nodeId+"|", " -Pagerank: "+ pageRankForThisPub+"|"+ "
| Pagerank: "+ pageRankForThisPub+"|"+ "
"+ "

    //System.out.println(nodeTitle+"|"+nodeId+"|"+pageRankForThisPub);
```
//System.out.format("%1$30s\t%2$6.5f\t", nodeId,pageRankForThisPub);
//System.out.println(nodeId);
//System.out.println(pageRankForThisPub);
//System.out.println(nodeTitle);
for(Author a: pub.getAuthors())
    System.out.print(a.getName()+" ");
System.out.println();
}

System.out.println("****************************************************
********************************************************************************
");
System.out.println("****************************************************
********************************************

");
System.out.println("****************************************************
********************************************************************************
");
    System.out.println("");
}

public String getPageRanksInCSV(List<PageRankOfNode> pageRanksAndNodes, int howMany) {
    StringBuilder sb = new StringBuilder();
    Collections.sort(pageRanksAndNodes);
    for(int i=0; i<howMany; i++) {
        PageRankOfNode prof = pageRanksAndNodes.get(i);
        String nodeId = prof.getNodeId();
        String nodeTitle = prof.getNodeTitle();
        double pageRankForThisPub = prof.getPageRank();

        sb.append(nodeTitle).append(",").append(nodeId).append(",").append(pageRankForThisPub).append("\n");
    }
    return sb.toString();
}

public List<EigenVectorOfNode> calculateEigenVector(MyGraph graph) {
    List<EigenVectorOfNode> eigenVectorsPerNode = new ArrayList<EigenVectorOfNode>();
    EigenvectorCentrality<Publication, MyEdge> eigenVector = new EigenvectorCentrality<Publication, MyEdge>(publicationGraph);
    eigenVector.acceptDisconnectedGraph(true);
    eigenVector.evaluate();

    for(Publication node : publicationGraph.getVertices()) {
        EigenVectorOfNode eigenVectorOfThisNode = new EigenVectorOfNode(node.getEid(), node.getTitle(), eigenVector.getVertexScore(node));
        eigenVectorsPerNode.add(eigenVectorOfThisNode);
    }
    return eigenVectorsPerNode;
public void printTopEigenRanks(List<EigenVectorOfNode> eigenAndNodes, int howMany) {
    Collections.sort(eigenAndNodes);
    for (int i = 0; i < howMany; i++) {
        EigenVectorOfNode eof = eigenAndNodes.get(i);
        String nodeId = eof.getNodeId();
        String nodeTitle = eof.getNodeTitle();
        Publication pub = publications.get(nodeId);
        double eigenForThisPub = eof.getEigenVector();
        System.out.format("%150s%30s%30s
        " + nodeTitle+"", -Eid: |"+ nodeId+"|", " -eigenVector: "+ eigenForThisPub+"|"+
        ");
        //System.out.format("%1$30s
        \t%2$6.5f
        \t",
        nodeId,eigenForThisPub);
        //System.out.println(nodeId);
        //System.out.println(eigenForThisPub);
        //System.out.println(nodeTitle);
        for (Author a: pub.getAuthors())
            System.out.print(a.getName()+""+"n");
    }
}

MyGraph getPublicationGraph() {
    return publicationGraph;
}

MyEdge.java

import java.io.Serializable;

public class MyEdge implements Serializable {
    private Publication sourceNode;
    private Publication targetNode;

    public MyEdge(Publication source, Publication target){
        sourceNode = source;
        targetNode = target;
    }

    public Publication getSourceNode() {
        return sourceNode;
    }

    public void setSourceNode(Publication sourceNode) {
        this.sourceNode = sourceNode;
    }
}
public Publication getTargetNode() {
    return targetNode;
}

public void setTargetNode(Publication targetNode) {
    this.targetNode = targetNode;
}

public boolean equals(Object other) {
    if (other instanceof MyEdge) {
        MyEdge otherEdge = (MyEdge) other;
        if (this.sourceNode.equals(otherEdge.sourceNode) &&
            this.targetNode.equals(otherEdge.targetNode))
            return true;
    }
    return false;
}

MyGraph.java

import java.util.HashMap;
import java.util.Map;
import edu.uci.ics.jung.graph.DirectedSparseMultigraph;

public class MyGraph extends DirectedSparseMultigraph<Publication, MyEdge> {
    private Map<String, Double> pageRanks;
    private Map<String, Double> eigenVectors;

    public MyGraph() {
        super();
        pageRanks = new HashMap<String, Double>();
        eigenVectors = new HashMap<String, Double>();
    }

    public void printEdges() {
        for (MyEdge edge : this.getEdges()) {
            System.out.println(edge.getSourceNode().getEid() + " --> " + edge.getTargetNode().getEid());
        }
    }

    public void printNodeAndDegree() {
        for (Publication node : this.getVertices()) {
            System.out.println(node.getEid() + " in degree " + this.inDegree(node) + " out degree " + this.outDegree(node));
        }
    }

    public void calculatePageRanks() {
        // PageRank algorithm implementation
    }
}
GraphVisualizer.java

import java.awt.Dimension;
import java.awt.Toolkit;
import javax.swing.JFrame;
import edu.uci.ics.jung.algorithms.layout.FRLayout;
import edu.uci.ics.jung.algorithms.layout.Layout;
import edu.uci.ics.jung.graph.ObservableGraph;
import edu.uci.ics.jung.visualization.VisualizationViewer;
import edu.uci.ics.jung.visualization.control.DefaultModalGraphMouse;
import edu.uci.ics.jung.visualization.control.ModalGraphMouse.Mode;
import edu.uci.ics.jung.visualization.decorators.ToStringLabeller;
import edu.uci.ics.jung.visualization.renderers.Ren
renderers.VertexLabel.Position;

public class GraphVisualizer extends JFrame {

    public GraphVisualizer(MyGraph graph) {
        super("My Graph");
        Layout<Publication, MyEdge> layout = new FRLayout<Publication,
MyEdge>(graph);
        Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
        Dimension graphDimension = new Dimension(screenSize.width - 50,
        screenSize.height - 50);
        VisualizationViewer<Publication, MyEdge> viewer = new
VisualizationViewer<Publication, MyEdge>(layout,graphDimension);

        viewer.getRenderer().getVertexLabelRenderer().setPosition(Position.CNTR);
        //viewer.getRenderContext().setVertexLabelTransformer(new
ToStringLabeller());
        DefaultModalGraphMouse<Publication, MyEdge> graphMouse = new
DefaultModalGraphMouse<Publication, MyEdge>();

        viewer.setGraphMouse(graphMouse);
        graphMouse.setMode(Mode.PICKING);
        ObservableGraph<Publication, MyEdge> observableGraph = new
ObservableGraph<Publication, MyEdge>(graph);

        setSize(Toolkit.getDefaultToolkit().getScreenSize());
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }

    public void calculateEigenVectors()
    {
    }

}
setContentPane(viewer);
setVisible(true);
}
}

**PagerankOfNode.java**

```java
public class PageRankOfNode implements Comparable<PageRankOfNode> {

    private String nodeId;
    private Double pageRank;
    private String nodeTitle;

    public PageRankOfNode(String nodeId, String nodeTitle, Double pageRank) {
        this.setNodeId(nodeId);
        this.setNodeTitle(nodeTitle);
        this.setPageRank(pageRank);
    }

    @Override
    public int compareTo(PageRankOfNode other) {
        if (this.getPageRank() < other.getPageRank())
            return 1;
        else if (this.pageRank > other.pageRank)
            return -1;
        else
            return 0;
    }

    public String getNodeTitle() {
        return nodeTitle;
    }

    public void setNodeTitle(String nodeTitle) {
        this.nodeTitle = nodeTitle;
    }

    String getNodeId() {
        return nodeId;
    }

    void setNodeId(String nodeId) {
        this nodeId = nodeId;
    }

    Double getPageRank() {
        return pageRank;
    }
}
```
void setPageRank(Double pageRank) {
    this.pageRank = pageRank;
}

EigenVectorOfNode.java

public class EigenVectorOfNode implements Comparable<EigenVectorOfNode> {
    private String nodeId;
    private Double eigenVector;
    private String nodeTitle;

    public EigenVectorOfNode(String nodeId, String nodeTitle, Double eigenVector) {
        this.setNodeId(nodeId);
        this.setNodeTitle(nodeTitle);
        this.setEigenVector(eigenVector);
    }

    @Override
    public int compareTo(EigenVectorOfNode other) {
        if(this.getEigenVector() < other.getEigenVector())
            return 1;
        else if(this.eigenVector > other.eigenVector)
            return -1;
        else
            return 0;
    }

    public String getNodeTitle() {
        return nodeTitle;
    }

    public void setNodeTitle(String nodeTitle) {
        this.nodeTitle = nodeTitle;
    }

    String getNodeId() {
        return nodeId;
    }

    void setNodeId(String nodeId) {
        this.nodeId = nodeId;
    }

    Double getEigenVector() {
        return eigenVector;
    }

    void setEigenVector(Double eigenVector) {
ApplicationFrame.java

```java
public class ApplicationFrame extends JFrame {

    private JPanel contentPane;
    private JButton btnLoadXmlFile;
    private JLabel lblFileInfo;
    private File xmlFolder;
    private JButton btnAnalyzeXmlFiles;
    private JButton btnCreatePageRank;
    private JPanel panel;
    private PublicationManager manager;
    private JPanel panel_1;
    private JPanel panel_2;
    private JPanel panel_3;
```
private JFormattedTextField dampingFactorTxt;
private JLabel lblHowManyPublications;
private JFormattedTextField txtHowManyPubsToPrint;
private JButton btnCreateEigenVector;
private JButton btnCreateGraph;

public ApplicationFrame() throws ParseException {
    setTitle("-----------The PagenTrust App-----------");
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 575, 412);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
    contentPane.setLayout(new BorderLayout(0, 0));
    setContentPane(contentPane);
    ButtonListener listener = new ButtonListener();

    panel = new JPanel();
    panel.setBorder(new TitledBorder(null, "XML Loader", TitledBorder.LEADING, TitledBorder.TOP, null, null));
    contentPane.add(panel, BorderLayout.NORTH);
    panel.setLayout(new MigLayout("", "[97px][117px][]", "[23px][][14px]")
        .add(new JButton("Load XML Folder"), "cell 0 0,alignx left,aligny center")
        .addActionListener(listener));

    btnLoadXmlFile = new JButton("Load XML Folder");
    btnLoadXmlFile.addActionListener(listener);

    btnAnalyzeXmlFiles = new JButton("Analyze XML Files");
    btnAnalyzeXmlFiles.setEnabled(false);
    btnAnalyzeXmlFiles.addActionListener(listener);

    panel_1 = new JPanel();
    panel_1.setBorder(new TitledBorder(null, "Graph Actions", TitledBorder.LEADING, TitledBorder.TOP, null, null));
    contentPane.add(panel_1, BorderLayout.CENTER);
    panel_1.setLayout(new GridLayout(2, 1, 1, 1));

    lblFileInfo = new JLabel("Number of files:");
    panel_1.add(lblFileInfo, "cell 2 0,alignx left,aligny center");

    panel_2 = new JPanel();
    panel_2.setBorder(new TitledBorder(null, "Page Rank Options", TitledBorder.LEADING, TitledBorder.TOP, null, null));
    panel_2.add(panel_3);
    panel_3.setLayout(null);

    lblDumpingFactor = new JLabel("Damping Factor (alpha)"Reducers")->
    lblDumpingFactor.setBounds(10, 24, 120, 14);
    panel_3.add(lblDumpingFactor);

    btnCreatePageRank = new JButton("Create Page Rank Metric");
    btnCreatePageRank.setEnabled(false);
    btnCreatePageRank.addActionListener(listener);
btnCreatePageRank.setBounds(10, 76, 219, 23);
panel_3.add(btnCreatePageRank);

dampingFactorTxt = new JFormattedTextField(new MaskFormatter("#.##"));
dampingFactorTxt.setText("0.15");
dampingFactorTxt.setColumns(5);
dampingFactorTxt.setBounds(140, 21, 35, 20);
panel_3.add(dampingFactorTxt);

lblHowManyPublications = new JLabel("How many publications to print?");
lblHowManyPublications.setBounds(10, 50, 194, 14);
panel_3.add(lblHowManyPublications);

txtHowManyPubsToPrint = new JFormattedTextField(new MaskFormatter("##"));
txtHowManyPubsToPrint.setText("99");
txtHowManyPubsToPrint.setColumns(5);
txtHowManyPubsToPrint.setBounds(215, 47, 35, 20);
panel_3.add(txtHowManyPubsToPrint);

btnCreateEigenVector = new JButton("Create Eigen Vector Centrality Metric");
btnCreateEigenVector.setBounds(296, 76, 211, 23);
panel_3.add(btnCreateEigenVector);
btnCreateEigenVector.setEnabled(false);
btnCreateEigenVector.addActionListener(listener);

panel_2 = new JPanel();
panel_2.setBorder(new TitledBorder(new EtchedBorder(EtchedBorder.LOWERED, null, null), "Visualize", TitledBorder.LEADING, TitledBorder.TOP, null, new Color(0, 0, 0)));
panel_1.add(panel_2);
panel_2.setLayout(null);

btnCreateGraph = new JButton("Create Graph");
btnCreateGraph.setBounds(22, 49, 117, 29);
panel_2.add(btnCreateGraph);
btnCreateGraph.setEnabled(false);
btnCreateGraph.addActionListener(listener);

setVisible(true);
}

public class ButtonListener implements ActionListener {

@Override
public void actionPerformed(ActionEvent arg0) {
    if(arg0.getSource() == btnLoadXmlFile){
        JFileChooser jfc = new JFileChooser();
        jfc.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
    }
}
int userDecision = jfc.showOpenDialog(null);
jfc.setCurrentDirectory(new File("D:\"));
if(userDecision == JFileChooser.APPROVE_OPTION) {
    xmlFolder = jfc.getSelectedFile();
    File[] listOffiles = xmlFolder.listFiles();
    int numFiles = listOffiles.length;
    lblFileInfo.setText("Folder Loaded, Total "+numFiles+" files");
    btnAnalyzeXmlFiles.setEnabled(true);
}
else if(arg0.getSource() == btnAnalyzeXmlFiles){
    manager = new PublicationManager(xmlFolder);
    btnCreatePageRank.setEnabled(true);
    btnCreateEigenVector.setEnabled(true);
    btnCreateGraph.setEnabled(true);
}
else if(arg0.getSource() == btnCreatePageRank) {
    String dampingFactorString = dampingFactorTxt.getText();
    if(dampingFactorString.length() > 1){
        double dampingFactor = Double.parseDouble(dampingFactorString);
        List<PageRankOfNode> pageRanksAndNodes = manager.calculatePageRank(dampingFactor);
        String howManyPubs = txtHowManyPubsToPrint.getText();
        int howMany = Integer.parseInt(howManyPubs);
        System.out.println("Page Ranks of Publication Graph. Damping Factor = "+dampingFactorString);
        manager.printTopPageRanks(pageRanksAndNodes,howMany);
    }
}
else if(arg0.getSource() == btnCreateEigenVector){
    List<EigenVectorOfNode> eigenAndNodes = manager.calculateEigenVector(null);
    String howManyPubs = txtHowManyPubsToPrint.getText();
    int howMany = Integer.parseInt(howManyPubs);
    manager.printTopEigenRanks(eigenAndNodes,howMany);
}
else if (arg0.getSource() == btnCreateGraph) {
    SwingUtilities.invokeLater(new Runnable() {
        public void run() {
            GraphVisualizer visualizer = new GraphVisualizer(manager.getPublicationGraph());
            visualizer.setVisible(true);
        }
    });
}
8.3. Graphs

The Graph for the whole network (not readable)
The class diagram