Network aware team formation

COCOON CORE - A Scientific Co-operation Recommender System

Bart Jan van Engelen
Faculty of Computer Science, Section Artificial Intelligence, Free University Amsterdam
De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands

August, 2012

supervisors: Lora Aroyo, Rory Sie

Abstract

Utilizing publicly available social collaboration network data of one university we propose to build a recommender system that assists researchers in selecting their research partners. We hypothesize that the influence an author has on the scientific collaboration network as well as the similarities of interest between two authors are predictors for the influence of the scientific output created by these authors. Furthermore we expect that making the network in which researchers co-operate more transparent by means of the recommender is of itself of value to the researchers. Although we did not find definitive proof that the recommender selects valuable co-operations above less valuable co-operations in our dataset we did find evidence that at least the influence of an author on the co-operation network is a good predictor for the influence of the scientific output of this author. What is more we find that users are moderately positive towards the recommendations of the system and find the system easy to use. Lastly we see that the network of our university is comparable to that of other universities, giving some external validity to these results.

Acknowledgements

I'd like to thank Lora Aroyo for her conceptual contributions and expert feedback. Furthermore the great amount of effort made by Rory Sie can not go unacknowledged. Your good advice and support were vital to the completion of this thesis. Finally this thesis would not have been without the support of family and friends who encouraged and enabled me during the course of this project. Your contributions far exceed any enumeration I could include here. Therefore I'm forced to suffice with enumerating you names. Jan, Debby, Mieke, Rudi, Alexander, Matijs and Victoria; thank you so, so very much.
## Contents

1 Introduction .......................................................................................................................... 4  
  1.1 Context .......................................................................................................................... 4  
  1.2 Relevance ....................................................................................................................... 4  
  1.3 Document Overview ..................................................................................................... 5  
2 Theoretical Background .................................................................................................... 6  
  2.1 Introduction ................................................................................................................... 6  
  2.2 Scientific quality ............................................................................................................ 6  
  2.3 Social network analysis ............................................................................................... 11  
  2.4 Recommender systems ............................................................................................... 25  
  2.5 Keyword similarity ..................................................................................................... 28  
3 Objective, research questions ............................................................................................ 32  
  3.1 Motivation for building a network based recommender system ................................. 32  
  3.2 Objective ...................................................................................................................... 32  
  3.3 Achieving objective ...................................................................................................... 32  
  3.4 Research Questions ................................................................................................... 33  
4 Methodology ....................................................................................................................... 35  
  4.1 Introduction to the methodology ................................................................................ 35  
  4.2 COCOON CORE workflow ........................................................................................ 35  
  4.3 Data Collection ........................................................................................................... 37  
  4.4 DSpace data description in graph terms .................................................................... 43  
  4.5 Research question 1 ................................................................................................... 45  
  4.6 Research question 2 ................................................................................................... 50  
5 Implementation .................................................................................................................. 54  
  5.1 Introduction .................................................................................................................. 54  
  5.2 System architecture ..................................................................................................... 54  
  5.3 Databases .................................................................................................................... 55  
  5.4 Calculations ................................................................................................................ 59  
  5.5 Offline Processes ....................................................................................................... 62  
  5.6 Online Processes ....................................................................................................... 69  
  5.7 Front End .................................................................................................................... 71  
  5.8 Caching ....................................................................................................................... 82  
  5.9 Libraries and datatypes ............................................................................................. 82  
  5.10 Summary of design choices ..................................................................................... 83  
6 Results .................................................................................................................................. 84  
  6.1 Experiment 1 ............................................................................................................... 84  
  6.2 Experiment 2 ............................................................................................................... 91  
  6.3 Summary of all results ............................................................................................... 97  
7 Discussion ............................................................................................................................ 99  
  7.1 Introduction to discussion ............................................................................................ 99  
  7.2 Experiment 1 ............................................................................................................... 99  
  7.3 Experiment 2 ............................................................................................................... 103  
  7.4 External Validity ....................................................................................................... 106  
8 Conclusion .......................................................................................................................... 112  
  8.1 Summary of what we have done ................................................................................. 112  
  8.2 Conclusions ............................................................................................................... 112  
  8.3 Similarities with existing research ............................................................................. 113  
  8.4 Implications ............................................................................................................... 113  
9 Future research .................................................................................................................... 115  
  9.1 H-Index as recommendation variable ......................................................................... 115  
  9.2 Combine data from multiple universities .................................................................. 115
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.3 External source for keywords</td>
<td>116</td>
</tr>
<tr>
<td>9.4 Use closeness or degree as centrality measure</td>
<td>116</td>
</tr>
<tr>
<td>9.5 Tweak hybrid system</td>
<td>116</td>
</tr>
<tr>
<td>9.6 Network centric recommender</td>
<td>116</td>
</tr>
<tr>
<td>10 References</td>
<td>117</td>
</tr>
<tr>
<td>Appendix A – Graph theoretical concepts</td>
<td>121</td>
</tr>
<tr>
<td>Appendix B – Implementation workflows</td>
<td>123</td>
</tr>
<tr>
<td>Offline Processes</td>
<td>123</td>
</tr>
<tr>
<td>Online Processes</td>
<td>128</td>
</tr>
<tr>
<td>Appendix C – Recommendation valuation questionnaire</td>
<td>138</td>
</tr>
<tr>
<td>Appendix D – SUS Usability questionnaire</td>
<td>139</td>
</tr>
<tr>
<td>Appendix E – Responses Experiment 2 (recommendations)</td>
<td>140</td>
</tr>
<tr>
<td>Appendix F – Responses Experiment 2 (SUS)</td>
<td>141</td>
</tr>
<tr>
<td>Appendix G – Original database tables</td>
<td>142</td>
</tr>
</tbody>
</table>
1 Introduction

1.1 Context
Online social networking has been a big success in the personal sphere for the last six to seven years. Nowadays, Internet users spend 18% of their online time on social networking, compared to 8% in 2007 (http://socialmediatoday.com/amzini/306252/social-networking-growth-stats-and-patterns). Apparently the tools provided by these online social networks add significant value to the personal lives of its end users. At first, this development sparked off the attention of businesses that were trying to monetize the large attention market that the use of these personal online social networks created. In the last few years, however, businesses are also trying to use the tools online social networks provide in order to leverage the implicit knowledge owned by their employees and explicit knowledge in their documents. Salesforce chatter is an example of both leveraging types of knowledge (http://www.salesforce.com/chatter/whatischatter/).

Using web 2.0 tools (wiki's, blogs etc.) in the process of forming scientific output is commonly referred to as Science 2.0 (http://en.wikipedia.org/wiki/Science_2.0). Although Science 2.0 is focussed primarily on online communication and co-creation tools, we now also see online social network tools being used in the scientific community as well; e.g. Mendeley (http://www.mendeley.com). The use of these online social network tools together with efforts to standardize the way in which meta data on scientific publications is made available through the internet enables use to use social network analysis to improve scientific co-operation.

1.2 Relevance
During their research career, researchers encounter many people with many different backgrounds and research topics. It is difficult to distinguish between them, to remember them, and if interesting, to maintain a relationship with them. This often results in researchers not being aware of what other researchers are doing. Especially for young researchers, such as junior researchers and PhD candidates, it may be difficult to identify those peers in their network that may be relevant or useful with respect to their own topic. With the adoption of social media in research, the abundance of information has become even more overwhelming due to the broadcasting character of this kind of communication.

There are a number of problems we face with respect to team selection in scientific co-operation. Firstly, researchers typically cannot see further than a few step away in their network. They are unable to oversee the value of a candidate alliance outside of that range. For instance, two scientists working on adjacent research topics, who are not acquainted with each other because they do not
share a common research partner. Secondly, researchers have to cope with an information overload. As a researcher’s network grows, it becomes more difficult to estimate the value of all peers in the network. When starting out a PhD student might be well aware of the research done by his entire co-operation network. Simply because this network consists of about 20 people. After a few years however he or she might have a co-operation network of 200 and having complete knowledge of all the research done by the scientists in his or her network has become impossible. Even if researchers were informed about all papers (content) and social variables (utilities) in their network, they simply could not compute the optimal choice of co-author. This phenomenon is known as bounded rationality (Colman, A. M., 2003) (Selten, R., 1998) (Simon, H. A., 1991).

We propose a solution to aforementioned problems by building a scientific cooperation recommender system. The system will combine author network influence and author interest similarity to recommend future co-authors to its users. The advice is based on the predicted value of future research co-operation. This way the system presents researchers a subset of their peers, for instance the most valuable peers with respect to a certain research topic. This may reduce the information overload perceived by the researchers. Additionally, providing such insight into the value of their peers may overcome the problem of bounded rationality. A scientific co-operation recommendation system can increase their awareness of the value of possible scientific co-operations. Therefore it may increase the scientific output and success of an individual researcher and/or the overall scientific output and success of a research community.

1.3 Document Overview

In the next chapter we will discuss the theoretical background of this thesis. Here we will discuss the theory that is relevant to this thesis and how we used this theory in our implementation. In the next chapter we formulate the objective and the research questions of this thesis. In the following chapter we will discuss the methodology. This chapter describes the methods we used to answer our research questions. In the next chapter we describe the implementation of the recommender. This chapter discusses all technical models and choices we made during the implementation of the system. The succeeding chapter gives an overview of the results of all experiments we discussed in the methodology. We also include interpretation of these results. In the discussion we will outline some aspects of the methodology that might reduce the certainty with which conclusions can be drawn from the results. We also include here a further discussion of the results themselves. In the penultimate chapter, conclusion, we summarize what we have done in this thesis and what the results are. We also discuss how the conclusions compare to other research and what the implications of this thesis are. We will finish with some thoughts on future research that could be done in extension of this thesis.
2 Theoretical Background

2.1 Introduction

The three theoretic fields that are relevant to this thesis are scientific quality measures, social network analysis and recommender systems. We will see what kind of scientific quality measures are available and why the H-index is a very popular measure.

Furthermore we discuss the graph concepts that are relevant to the networks used in this thesis and the network models to which we try to fit our network. On top of that we discuss instances of networks based on scientific output data and the measures we use to analyse these networks. We continue by explaining why betweenness centrality is best suited as measurement for network influence in our network. Next we outline the different recommendation techniques used in recommender systems. Later on in the objective we explain what techniques we use in our recommender system.

We conclude this chapter with an explanation of two techniques often used in recommender systems; Term Frequency – Inverse Document Frequency calculation and vector similarity, followed by the objective, research questions and hypothesis.

2.2 Scientific quality

2.2.1 Introduction to scientific quality

Every year Times magazine publishes the Times Higher Education Ranking. A ranking of the top 400 universities around the world (http://www.timeshighereducation.co.uk/world-university-rankings/2011-2012/top-400.html). This ranking is used to benchmark scientific output of universities. All dutch universities are in the top 200 with an overall score between 40 – 60 % of the top universities in the world. Rating scientific output is necessary for faculty recruitment and advancement as well as for awards or grants. But how do we measure the quality of scientific output? - Both for a university and for an individual researcher. For this thesis we need scientific quality measures to be able to rate authors and author co-operations.

2.2.1.2 Focus for this thesis

We will focus on the scientific output of a single scientist because this is a target variable in our evaluation. Also we work on a dataset for one university so there are no comparisons to be made of scientific output between universities.

2.2.2 Tools to calculate scientific output

In response to the demand for valuation of scientific output a few tools have been created. Publish or Perish is one example (http://www.harzing.com/pop.htm). Based on google scholar citations this
desktop tool calculates a number of scientific output measures. Publish or perish advertises itself as a tool to help individual researchers present their research impact.

Tools like publish or perish generate scientific quality values, but do not help scientists choose strategic co-operations based on these values. So while tools like publish or perish are designed to create data to help valuate the impact of the work of scientists, we propose to use this data and help scientists choose with whom they should cooperate to make a larger scientific impact. We can also use the data to evaluate the successfulness of co-operations.

2.2.3 Problems with scientific quality
When measuring the quality of scientific output we have to deal with some general problems. Publishing a lot of papers and putting effort in getting attention to them might give a lot of exposure and this might lead to a sense of quality, but the number of times a paper is read does not necessarily say anything about the quality of the paper. If a paper is read a lot, but generates little citations the author might be a good networker, but his work is not so influential. On the other hand; a paper with a lot of citations does not necessarily introduce a very influential idea. It might for instance be a very well structured review paper. The real influence however might be in the papers it reviews. Furthermore, the average number of citations is dependant on the research field in which a paper is published. The number of citations of a paper in one research field reflects a greater influence than the same number of citations of a paper in a different research field. Comparing researchers between research fields is therefore difficult.

Some scientific output measures rely on peer review performed by publication comities. Peer review of course is very useful when it is conducted by true experts in the field of the paper. In practice however peer reviewers often have a more general competence and therefore resort in part to other measures themselves like the ones mentioned above. In that sense these methods measure the reputation of an author and not the present scientific output of the author. Journal impact factors published in SCI Journal Citation Reports are also often used. They present the quality of the journal and therefore the quality of the papers published in the journal. The impact factors however correlate poorly with the citations of papers (Seglen, P. O., 1997).

2.2.4 Scientific quality measures
Despite the general problems with scientific output valuation mentioned above there are a number of quality measures in use today. We will list a few and explain why we believe the H-index is best choice as target variable for our recommender.

2.2.4.1 number of papers
One measure for the scientific output of a scientist is the number of papers a scientist has published.
This is a very basic measure. Moreover this measure addresses the quantity of the scientific output, but not the quality. Any viable measure for scientific output is based on citations or in a lesser degree peer review.

2.2.4.2 number of citations
This measure does focus on the quality of the output, but might be influenced by one or two big hits. Also the risk of missing citations of a paper in our data is potentially very big. If we cannot automatically find a paper of a person that in reality has a lot of citations, our target variable is off by a big degree.

2.2.4.3 average citations per paper
The average citations per paper does also focus on the quality of the output, but does not take productivity of a scientist into account. A scientist that published 1 paper that gathered 10 citations gets the same valuation as a scientist that published 10 papers that all gathered 10 citations.

2.2.4.4 The number of papers with more than y citations
This does help to measure the importance of papers and does only count significant papers. On the down side y has an arbitrary value that might create a bias towards certain authors with papers that have a citation number just over the y value. This measure is particularly vulnerable to biases between different research fields.

2.2.4.5 H-index
The H-index quantifies both the actual scientific productivity and the apparent scientific impact of a scientist. A scientist has index h if h of his or her total number of papers – Np,tot - have at least h citations each and the other (Np,tot - h) papers have h or less citations each (Hirsch, J E., 2005). For example, an H-index of 20 means the researcher has 20 papers, each of which has been cited 20+ times.

![Figure 1: H-index in Paper Citation Distribution](image)

Figure 1: H-index in Paper Citation Distribution
The H-index is an alternative to total citations which can be disproportionately affected by a few very highly cited papers. The total number of citations to papers of a scientist will be larger than $h^2$. Hirsch found that a scientist total number of citations ($N_{c,tot}$) is typically 3 to 5 times higher than his or her squared H-index.

$$N_{c,tot} = ah^2$$

$3 \leq a \leq 5$

All papers that have more citations than $h$, but do not have a number of equally cited papers by the same scientist, will not be counted in the $h$-value. This helps to normalize the index and not give a “one hit scientist” a big H-index number. Citations of papers with a lower citation number than $h$ will also not count in the H-index.

Hirsch recognizes the influence of time on the H-index of a scientist. An important paper will get new citations every year, so without adding new papers the H-index of a scientist can increase over time. Although this is appropriate because the influence of a paper is best measured over time, it does however put scientists with a low scientific age at a disadvantage. Google citations, for instance, compensates for this by additionally showing the H-index over the past five years (http://scholar.google.com/citations). In a broader research the effects of age differences in a research team can be considered, but for us it falls outside of our scope. For our purposes it is therefore not necessary to compensate for the disadvantage of younger scientist. Time can give researchers a higher H-index, but only if old papers get new citations which increases the impact of a scientist. This impact is what we are interested in, not the age of the scientist.

We sum up the advantages of the H-index over other single number measures:

- The advantage of the H-index over the total number of papers $N_p$ is that it measures impact and not just productivity, which is less likely to be associated with power and knowledge than impact is.

- The advantage of the H-index over the total number of citations $N_c$ is that it corrects for a few papers that get a lot of citations. These papers may not represent the author well particularly if these papers are co-written by other scientists.

- The advantage of the H-index over the average citations per paper is that it reflects productivity as well as citations. The average citations per paper does not take the number of papers into account.

- The number of papers with more than $y$ citations does look at only papers with a significant
impact, but since y is arbitrary it will disadvantage scientist that may have a solid number of papers with a number of citations just below y.

2.2.4.5.1 Our use of the H-index

Our dataset of papers does not include the citations of these papers. We could however use the H-index of authors as calculated by other researchers. Two of these other sources are Publish or Perish and Reader Meter (http://readermeter.org/).

Publish or perish uses google scholar as resource for papers and citations to papers. In order to get the citations per year per paper we have to go back to the original source; Google Scholar. Reader Meter does not calculate the H-index using the citations of the papers of the author, but uses the number of times a paper is read on Mendeley (http://www.mendeley.com). In theory these two H-indexes should correlate as papers that are cited more often will most likely be read more often. We will have to see whether these two indices compare.

Because the H-index in practice can differ greatly from one resource to another we use the Mendeley data to validate the google scholar data. For our recommendation we are not interested in absolute values, but rather relative values.

We consider it to be valid to use H-indices calculated from other sources (Google Scholar and Mendeley) because all authors in our database have the same (dis)advantages to their visibility in these publicly available repositories.

2.2.4.6 G-index

The g-index is another single natural number index for scientific output. It is defined by: Given a set of articles ranked in decreasing order of the number of citations that they received, the g-index is the (unique) largest number such that the top g articles received (together) at least g^2 citations. An illustrated example for an g-index can be found in figure 2.
The g-index can be restated as: g papers of the author have on average g citations. This is a less strict index than the H-index which states that h papers should all have at least h citations. The g-index ignores less of the total citations of the author and therefore the g-index will be higher than the H-index. As the H-index the g-index ignores the long tail of scarcely cited papers, but it allows for highly cited papers to 'move' citations to lower cited papers to not ignore the potentially high number of citations of the most highly cited papers (Egghe, L., 2006).

2.2.4.7 H-b index
The H-b index is developed for indexing the popularity of topics instead of authors. It can therefore help new researchers to identify interesting topics to do research in. For this thesis the H-b index is not applicable because our prime goal is to compare authors and not topics. The topics are selected by the users themselves and not recommended by the system. A topic recommender could in a later stage be included in the system.

2.2.5 Recap scientific quality
The scientific quality is the quality of the scientific output of a researcher or an organization. In this thesis we will focus on the scientific output of researchers. Any good measure for scientific output is based on the number of citations of that output. There are many simple and complex metrics that are based on the number of citations of papers by a specific author. For our thesis we opted to focus on the H-index which adjusts for one hit wonders and it rewards impact over productivity.

2.3 Social network analysis
Social network analysis is the methodical analysis of a social relationship structure after it has been translated to a network structure. The field emerged in the 1930's out of three research fields: Psychology, Anthropology and Mathematics. For this thesis we need social network analysis in order to measure the network influence an author has in our dataset.
2.3.1 Development of social network analysis

2.3.1.1 Psychology
In the field of psychology Gestalt theory proposes perception as a pattern matching process. This concept can be applied to the cognitive scope, but also in the social scope by researching social perception and group structure. There are three main psychologists of the gestalt movement that made the first movement into the research field of social network analysis.

Jacob Moreno asked people who their friends were and mapped these relationships in respect to their psychological behaviour (Moreno, J. L., Jennings, H. H., 1938). He stated that sociological phenomenons could be explained to these small scale social relationships. It was therefore logical to map these small scale social relationships in a network Moreno called a sociogram. A sociogram is a network of people (points) an lines between these points (relationships). Moreno investigated 'stars' (people with a lot of relationships) and social asymmetry (difference between in and out degree).

Kurt Lewin arrived at a same representation of a social network from a slightly different starting point. He perceived group behaviour as a function of conflicting social forces. Lewin introduces two new concepts that play a role in modern social network analysis. The first is the idea of regions as (relatively) isolated groups within a network. The second is the idea of the mathematical investigation of the social network.

Fritz Heider was interested in the balance between people. He proposed that the mind always seeks balance. People therefore prefer to form relationships with people they like. A natural question is: what happens is A likes B and A likes C but B does not like C? To be in balance A has to conform to B or C and dislike the other. This of course is a social structure easily expressed in a graph.

Cartwright and Harary proved mathematically that groups that have these imbalanced are bound to transform into multiple cliques that minimize the imbalance.

2.3.1.2 Anthropology
To study a society one can study the web of kinship of friendship relationships between members of a society. On a theoretical level that is exactly what Anthropologist Radcliffe-Brown and Nadel did.

Empirical work comes from the famous worker efficiency studies in the Western Electric Company in Chicago in the 1920's. Warner, Mayo, Roethlisberger and Dickson studied the optimal conditions for worker productivity in the plant, and found that any change in conditions resulted in an increase of productivity. Shifting to a more anthropological approach they just observed workers all day and payed attention to the social relationships between workers. What they found was that the 'informal structure' within the plant was of great influence on the productivity.

On the Manchester University in the 1950's Barnes and Bott and Mitchell researched the influence
of individual relationships on the society as a whole. They looked not only at the content of these relationships, but also at the structure of social network. Based on this work Harrison White of Harvard University worked on the mathematical aspect of social network analysis. In the 1960's and 1970's White translated many concepts like the 'network role' of a person in the network to mathematical formulas which enabled researchers to quantify their social network research. Among the first research based on this quantifiable approach was a book by Mark Granovetter – Getting a job – in which he describes that most people find a job through relationships that can be characterized as acquaintances (Granovetter, M.S., 1974). Another quantitative study was done by Lee – The Search for an Abortionist (Lee, N.H., 1969). This study showed that typically a woman looking for an doctor who will perform an abortion has three friends or acquaintances between herself and the doctor.

2.3.1.3 Mathematics
The main research field in mathematics in relation to social network analysis is obviously graph theory. Graph theory started in 1736 with Euler's paper on the 'seven bridges of Königsberg' (Euler, 1956). By reformulating the problem to a series of bridges crossed between landmasses Euler introduced graph theory. The problem could be formulated in a much more abstract form. Landmasses became vertices (or nodes) and bridges became edges.

2.3.2 Graph theory
Euler made the abstraction from landmasses and bridges to nodes and edges. This abstraction can be made for many more problems including social network problems. The researchers discussed in the fields of psychology and anthropology basically made this abstraction. Below we will in an abstract manner discuss the graph theoretical concepts that are relevant to our social network analysis. In 2.3.4.1.1 Co-author networks on page 17 we will make the translation from the general graph theory to our specific problem domain.

2.3.2.1 Graph definitions
A graph is a set of nodes and edges. Nodes can be seen as objects and edges as links between two objects. We specify a set of nodes, or vertices as the set V. We specify the set of edges as the set E. Now we can specify a graph G as a set of vertices and edges: G = (V,E) where E is a two-dimensional subset of V. For each element of E, edge \{v,u\} v ∈ V and u ∈ V. Both nodes and edges can have labels. The size of a graph is the number of its edges, i.e. |E(G)|. Any two vertices v an u in V that have a relation in E are said to be adjacent to each other, v ~ u.

A path is an ordered set of neighbouring edges that do not visit any node more than once. If two nodes are connected by a path of one or more edges we call the two nodes connected. A trail is an ordered set of neighbouring edges that do not visit any edge more than once. The distance dG(u, v)
between two vertices \( v \) and \( u \) is the length of a shortest path between them. If two vertices are not connected the distance between them is infinite. The **eccentricity** \( \varepsilon_G(v) \) of a vertex \( v \) is the maximum distance from \( v \) to any other vertex. The **diameter** \( \text{diam}(G) \) of a graph \( G \) is the maximum eccentricity over all vertices in a graph. The **radius** \( \text{rad}(G) \) is the minimum eccentricity of all vertices in a graph.

Edges can be **directed** or **undirected**. A directed edge has an **origin** node and a **target** node. An undirected edge in symmetrical in the sense that it can be traveled in both directions so that both endpoints connected to the edge can function as origin and as target. The number of edges that a node has is called its **degree**. In a directed graph we distinguish in-degree and out-degree by the number of edges that end in the node and the number of edges that originate from the node. If all edges that are connected to the node are undirected we simply speak of the degree of a node. We will not discuss directed graphs any further because we will not use them in our research. If the origin and target of an edge \( \{v,u\} \) are the same node, \( v = u \), we call the edge a **loop**. It is also possible for two nodes to share multiple edges. If a graph allows for multiple edges between two nodes and allows for loops to exists we call the graph a **quiver**. If a graph has no **multiplicity** in edges we call the graph **simple**.

The properties of a graph derive from the nodes and edges they have. If all nodes in a graph have the same degree we call the graph **regular**. If all nodes in a graph share an edge with all other nodes in the graph we call the graph **complete**. If a subgraph of a graph is fully connected (complete) we call this subgraph a **clique**. A **k-clique** is a clique of the **order** \( k \), meaning it contains \( k \) nodes. An clique that is not a subgraph of another clique is called a maximal clique. An **N-clique** is a subgraph made up of nodes that have a maximum distance of \( N \) to all other nodes in the N-clique.

A graph is called **connected** if for each pair of nodes there exists a path from one node to the other. If there is at least one pair of nodes that is disconnected the whole graph is called **disconnected**. In this case at least two subsets of nodes \( \subseteq V \) and edges \( \subseteq E \) exist that form disconnected **subgraphs**. Disconnected subgraphs are called **components**. A **k-connected** graph is a graph in which it takes a set of \( k \) edges to be removed before the graph is disconnected. Finally a **tree** is a graph in which their exist no paths that include any node more than one time.

The **cluster coefficient** of a node is the chance that this node is connected to another node provided that this node and that other node share a neighbour.

### 2.3.2.2 Graphical representations of graphs

The graphical representation of a graph is set up in the following fashion. Nodes are depicted by points, edges are depicted by lines. If an edge is directed a line is replaced by an arrow. If nodes and / or edges have labels those can be connected to the points and / or lines.
As an example we present Figure 3: An arbitrary graph. It depicts a connected, complete, irregular, undirected quiver with three nodes, six edges one loop, a diameter and a radius of 1 and an average degree of 4.

![Figure 3: An arbitrary graph](image)

### 2.3.3 Network models

The values for the graph variables discussed in Graph definitions on page 13 for a given graph determine the nature of the graph. We can cluster graphs into graph models. Graphs that are part of a model share similar network variable values and therefore share a similar nature. We will discuss some of these network models that are relevant to our research below.

#### 2.3.3.1 Scale-free model

A *scale free network* is a network that is characterized by a few network nodes that have many links, and many nodes that have few links. In practice, this means that 20 per cent of the nodes, called hubs, accounts for 80 per cent of the links (degree), and that 80 per cent of the nodes account for the other 20 per cent of the links. This is also known as a power law degree distribution. The power law degree distribution was first discovered by Derek de Solla Price on 1965 in citation networks (Price 1965). It got rediscovered by Albert-László Barabási when studying the world wide web (Barabási, a., 1999).

The relatively high number of hubs in the network makes the network very robust against random failures. Around 80% of the nodes can be removed without increasing the average shortest path length or the number of components by much. If the failure however is targeted against the hubs, the network can quickly become disintegrated. This underlines the need for transparency of the network on a more global scale. On an individual scale it is important to know who to connect to. On a network scale it is important to know who is important for the connectivity of the network.

The degree distribution can be explained by a network of tightly connected clusters which themselves are connected by hubs. The members of the clusters have a low degree and high cluster coefficient. The hubs have a high degree and low cluster coefficient. This phenomenon is repeated...
on multiple levels. This structure results in a small-world effect.

### 2.3.3.1.1 Small-world effect

The small-world effect states that two random nodes in a network are connected by a small number of steps. This concept was made famous by Stanley Milgram in his small world experiment (Milgram, S., 1967). Milgram stated that two random Americans are typically connected by a chain of only six acquaintances.

Since Milgrams experiment the small world effect has been found in many networks, not in the least social networks. It is highly likely that we will see the small-world effect in our data. In graph terms the small-world effect translates to a low average shortest path between two random nodes even when the graph gets very large. The average shortest path $L$ is in the order of $\log N$ where $N$ is the number of nodes. (see below)

$$L \propto \log N$$

*Illustration 1: Relation between average shortest path length ($L$) and number of nodes ($N$) in a small-world network*

### 2.3.3.1.2 Generating scale-free networks

Albert-László Barabási proposed a way to grow networks so that they become scale-free networks. When growing the network in each iteration the probability of a new node connecting to an existing node is made dependant on the degree of the existing node. The higher the degree of the existing node, the higher the chance that new nodes connect to this existing node. This preferential attachment results in the power law degree distribution.

### 2.3.3.1.3 Link with co-authorship networks

A large number of natural networks including co-authorship networks have proven to be best fitted by scale-free networks. It is therefore likely that our university co-authorship network is also a scale free network.

### 2.3.3.1.4 Relevance of scale free model and small world effect

The scale free model and the small world effect are relevant to this thesis because they are commonly found in social networks. Also they account for the few people with a very influential position in the network. This is a characteristic we are looking for in our recommendation.

### 2.3.3.2 Random model

To proof that the order in a social network is not created by random chance alone, a network is often compared to a random network. A random network is a network in which the set of nodes and the
set of edges are created randomly. This model was proposed by Paul Erdős and Alfréd Rényi (Erdos, P., & Renyi, A., 1959). A random graph can be generated in 2 ways

1. The number of nodes and edges is set. In this case we take all possible graphs composed of N nodes and M edges and choose randomly between them.

2. Only the number of nodes is set. In this case we take a pair of nodes and start connecting them randomly with chance p. The more p limits to 1, the more we get a complete graph. The more p limit to 0, the higher the chance we get a unconnected graph.

Random graphs tend to have a low average shortest path like scale-free networks, but also a low cluster coefficient unlike scale-free networks.

2.3.4 Research instances of networks

2.3.4.1 Research networks
With the emergence of online libraries for scientific papers we can build graph models of research networks. This allows for a network analysis approach to the understanding of the creation and diffusion of scientific knowledge. From these libraries we can create a number of different networks. We will look at keyword networks, citations networks and co-author networks. For this thesis the latter is the most important.

2.3.4.1.1 Co-author networks
Authors are connected through co-authorship of papers. When authors write a paper, all authors form a relationship with that paper. For instance, if author A, B, and C write a paper (P) together, the author-paper relationships \{A,P\}, \{B,P\}, and \{C,P\} are formed. This relation is graphically represented in Illustration 2: Author Paper graph below, which depicts an Author – Paper network. In social network analysis terms, such an author-paper network is called a two-mode network.
2.3.4.1.2 Transforming the co-author network for our use in this thesis

This graph makes it possible to calculate the average number of authors per paper and average papers per author. The different types of nodes (papers and authors) make it more difficult to calculate other network measures, because we have to conflicting node types. We can transform the author-paper relationships to author-author relationships between authors that have author-paper relationships with the same paper (see Illustration 3: Author Paper graphs with author-author links below).
We see that by doing this we create a cluster between two authors and one paper. When generating these author-author links in author-paper graphs all paper-nodes will have a cluster coefficient of 1. This immediately follows from the way we generated the network and we will prove it below.

1) if paper node P is connected to author node A1 and P is connected to author node A2 and A1 is not A2 → A1 is connected to A2

2) if there is no A1 such that P-A1 then in theory P has cluster coefficient 1 because the node has no edges an therefor the change that if two nodes are neighbours of P, that they share an edge is 1, because there is no instance in which the premise is true. (in reality we might want to say that the node has no cluster coefficient, because it has no edges.)

3) if there is a A1 such that P-A1 then by 1) we have: For all A2 such that P-A2 we have A1-A2

When looking at network variables *average collaborators, average distance* and *average cluster coefficient* between authors the paper nodes and paper-author links have no additional information to the author-author links. Furthermore, when visually analysing co-operation graphs it is much cleaner to look at only author-author links and only author nodes. We can create these graphs by removing the paper nodes and paper-author edges from the graph after adding all author-author nodes. An example of an author-author graph is depicted in Illustration 4 below. In this network we see three network concepts that are important for the type of network that we are describing. A **cluster** is a subgraph in which forms a complete graph. A **hub** is a node though witch a lot of paths travel. Typically a hub has a high degree and connects multiple subgraphs which would otherwise be disconnected. A **bridge** is an edge which, if removed, disconnects the graph.
Finally this observation. All authors can be seen as co-authors of themselves which implies that all nodes should have a loop edge connected to them for each paper they author, making the network reflexive. Because these loops do not aid in any of our relevant calculations and make the graphical representations of the network much less clean, we omit them in our graphs. It should however be noted that by doing this we have no way of representing a paper which is written by a single author. For this thesis that is no problem because we are looking at co-operation.

In the following descriptions of other networks in the scientific sphere we will translate the co-authorship network described above to the individual other network types.

2.3.4.1.3 Co-citation networks
Co-citation networks are networks in which papers are nodes and citations are edges. Co-citation networks are valuable when identifying important papers in a set of papers. Hubs are typically papers that combine research fields or are very important within a field. A co-citation network needs to have directed edges because a citation relationship is, in a vast majority of the cases, not symmetric.

2.3.4.1.4 Mendeley read networks
Mendeley read networks are like citation networks to the extend that they present popularity of publications. Nodes are papers, but edges do not depict citations, but reads by other authors. A
mendeley read network therefore has both author nodes and paper nodes.

### 2.3.4.2 Keyword networks

A keyword network is comparable to an author network. If two keywords are both connected to a unique paper these two keywords are connected. These networks are interesting for researchers because they can discover research fields in clusters of keywords. Also they can identify important keywords by studying their betweenness.

### 2.3.5 network measures

Network measurements can be analysed on different levels. For instance, if we look at the clustering coefficient we can speak of the clustering coefficient of a given node, the average clustering coefficient in a subgraph or the average clustering coefficient in an entire graph.

A number of network measures is provided in Appendix A – Graph theoretical concepts. We will discuss a few a little deeper in this chapter.

#### 2.3.5.1 Network level

1. **Degree**
   
   We speak of degree assuming a unidirectional graph. In an author-author graph this measure corresponds to the number of scientists the author is connected to. Global derivatives of this local measure are average degree and degree distribution. The shape of this distribution is a sign for the kind of network that the authors form. For a lot of networks including research networks this distribution will be a long tail distribution. A tell-tale sign of a scale free network.

2. **Clustering coefficient**
   
   When we look at the cluster coefficient on the network level we look at the average cluster coefficient of all nodes in the network. An average clustering coefficient of 1 for an entire graph implies that the graph is complete. The average cluster coefficient in this case is an example of a global measurement. A certain minimum average cluster coefficient is necessary for a fruitful research network. A high cluster coefficient indicates that most of the nodes in the network are part of at least one highly connected cluster. It is proposed that these clusters are necessary for successful research in existing fields. These clusters are associated with specialization in (small) subsets of the network that operate in a certain discipline (Lambiotte, R., & Panzarasa, P., 2009).

#### 2.3.5.2 Community level

1. **Cliques (cluster)**
   
   A clique is a subgraph in which all nodes are fully connected. Provided that there are no isolated nodes we can remark that a single node with a cluster coefficient of 1 is part of a clique of at least 3
nodes. In fact, all members of any clique have an cluster coefficient of 1 within that subgraph so if a subgraph is a clique, then the average clustering coefficient of that subgraph is also 1. Cliques are important structures in a graph. In co-author graphs they identify research groups that share a common expertise. N-cliques and factions are less strict versions of the same concept as cliques.

### 2.3.5.2.2 N-cliques
An n-clique is a set of nodes which all share shortest paths not longer than N. This measurement can be used when the requirements of cliques are a bit too strict.

### 2.3.5.2.3 Factions
Nodes in a faction do not have to form a complete graph, but do have a higher density in their subgraph than with nodes outside of the faction.

### 2.3.5.3 Individual level
On an individual level we are interested in the influence the local values for degree and cluster coefficient have on the power of the node. To investigate this influence we have to look at the network flow on a local level. This is the flow through a node under inspection.

### 2.3.5.3.1 Network Flow
One way of measuring someone's importance in a network is by measuring his or her centrality. Several centrality measures exist, each having their own application area and method of calculation. The way in which information flows in the network determines the most appropriate centrality measure for that network. There are two major variables that have to be considered when dealing with network flow: the kind of trajectories and the method of spread. The trajectory of information flow is the route it travels through the network. For instance, DHL parcels can travel from person A to person D via person B and C. Here the parcel travels from node A to D over the edges A-B, B-C and C-D. The method of spread of information (or diffusion) in a network describes how information is propagated between nodes. For DHL for example we can state that method of spread is limited to transferring the parcel from one node to the next. The parcel can not be copied or divided in separate parts.

With respect to trajectories, we distinguish paths, trails or walks. Paths do not allow for the same node to be visited twice. Trails do not allow for edges to be visited twice. Walks do not put any constrictions on the trajectory of information flow in a network.

If the information can be copied we can distinguish serial duplication and parallel duplication of information between nodes. Serial duplication occurs when information is passed to one node at a time. Parallel duplication occurs when information is passed to multiple nodes at the same time. If
the information cannot be copied we can only have transfer from one node to one other node.

2.3.5.3.2 Betweenness centrality

For this thesis we can choose between a wide range of centrality measures available for instance, but not limited to: betweenness, closeness, degree centrality, and eigenvector centrality. Each of these centrality measures is appropriate for a different network-flow class. If we consider the distance to powerful people in the domain of scientific co-operation networks, betweenness seems to be the best choice.

First we compute power. Betweenness centrality is positively correlated to power in organizations. We can intuitively see this as follows: Betweenness calculates the number of times a node is on a shortest path between two other nodes. A shortest path between two node A and B is the path that takes the minimum number of steps between node A and node B. Being on the path on which information travels, increases the awareness of the node of the structure of the network in which it is situated. This increased awareness (cognitive accuracy) increases power (Krackhardt, D., 1990). Being on a path over which information travels gives a node control over the flow of the information. Furthermore, being on a relatively high number of paths over which information travels will increase the diversity of the information that the node possesses (Burt, R. S., 2004).

Betweenness considers the shortest paths (geodesic paths) between two nodes. Only counting geodesic paths is a rather strict assumption of network flow and it is not a good representation of network flow in a real life scientific co-operation network. For instance, two scientists divided by a shortest path of length 5 will likely not be aware of this path due to their incomplete view of the network. Furthermore, even if they would have had a complete view of the network, they would be limited by their bounded rationality (see Relevance on page 4) trying to construct a shortest path. A digital representation of a social co-operation network, however, allows for a clear overview and calculation of the shortest path between nodes. Therefore, in calculating our recommendations based on distance to the most relevant powerful people we can use the geodesic paths in the network.

The relation between information flow and relevant centrality measures is well documented by Borgatti. Looking at traffic flow in a network Borgatti distinguishes eight classes: Used goods, money, Gossip, E-mail, attitudes, infection and packages. Borgatti’s classification is based on the two aforementioned major variables – information trajectory and method of spread – together with determinism of traffic flow (Borgatti, S., 2005).

When we consider the shortest path to a powerful person we look for the geodesic path and use a transfer flow. Each 'handshake' that separates the two researchers (over a geodesic path) can be seen
as one transfer. The betweenness of a node is the sum of occurrences of a node being part of a geodesic path between two other nodes, divided by the total number of geodesic paths between these two other nodes. Using this measure we assume that traffic is indivisible, chooses randomly between geodesic paths, is deterministic and can see all the paths between its origin and its target. As we said before, we can assume these things for the purpose of computing the “handshake-path” between a contact solicitor and a powerful target.

When we are considering knowledge transfer in the scientific co-operation network we are dealing with trails and parallel duplication of information. Trails because it is unlikely that information travels the same edge twice. When one scientist sends information to another scientist we will assume that this first scientist will remember this transfer and not send the same paper (or other information) twice. In this case the closeness and degree are more suitable centrality measures. The closeness of a node is defined by the average distance to all other nodes in the network over geodesic paths. A low closeness score is associated with a high chance of receiving information early in the spreading process when it still has the most competitive value (provided that the information can spread by parallel duplication). If the information spreads discriminately through the network – that is to say that it has an inherent preference for one path over another -, a low closeness will not necessarily be an advantage because the information trajectory might have an inherent preference not to travel through the given node. We will assume that information relevant to a node is propagated to this node by its neighbours with the same likelihood as it is passed to other neighbours of its neighbours.

We can look at closeness as a measure for the distance between nodes while betweenness is a measure for the importance of the location of a node in the network. Closeness is therefore more appropriate for indeterministic information flow (spreading of interesting papers through a network) while betweenness is more appropriate for deterministic information flow (the shortest path to an interesting node). We can further illustrate the difference when we consider a graph of two components. In theory the distance between any set pair of nodes in two different components is infinite. Say we connect the two components by connecting one pair of nodes each in a separate component. The relative betweenness of these two nodes will increase greatly because these nodes are now on every geodesic path between every pair of nodes spread over the former two components. The relative closeness of the two nodes will increase significantly less than the betweenness because the advantage of the new bridge between the two former components will be spread more evenly between all nodes in the network.

2.3.5.3.3 Degree Centrality
Degree centrality is based on the number of links an author has. The more links an author has, the
more this author is central in the network. When information is created at a random place in the network, the chance of getting this information early is larger the higher the degree of the author. This however does not account for the typical nature of a co-authorship network in which relatively dense fractions are connected by hubs. In such a network these hubs have on average the highest chance of getting the information early, because they are on a lot of shortest paths. For our purpose the betweenness centrality is therefore a more appropriate centrality measure.

2.3.6 Recap network models and research instances of networks

Networks can be categorized into models. For our purpose the scale free model is very relevant because it is found in most social networks. The scale free model is characterize by a small number of nodes which are very influential. To measure this influence we need a measure. Betweenness centrality is a very appropriate measure. It indicates the relative number of times that a node is on the shortest path between two random nodes. Such individuals can exert power on the network because they can influence the flow of information in the network. Another important network measure is the cluster coefficient. This measure indicates the chance that two nodes share an edge given that they have a common neighbour. A high cluster coefficient is a sign for a high density within a subgraph. This indicates that the authors in this subgraph share a lot of interests. Shared interest is another variable in our recommender.

2.4 Recommender systems

2.4.1 A short history

Recommender systems in general are a subset of information retrieval systems. Their goal is to predict the perceived value of an item to a specific user. Well known services that use recommender systems are Amazon, bol.com, Netflix, last.fm, Pandora radio, Facebook and LinkedIn. In 2006 Netflix started a competition with a one million dollar award for an recommender that improved 10% on Netflix own predicting performance on a given dataset of movies. The competition energized the research field of recommender systems, but the research field was created a lot earlier.

At the beginning of the internet the problem arose that there was too much information for a single user to sift through to find information valuable to that user. First this problem was present in email and usenet communication, but it got much bigger with the arrival of the World Wide Web. The first idea for a recommender system was to let eager users rate content and then profile other users to find similar rating users. The original name for the field was therefore collaborative filtering. David Goldberg coined the phrase at Xerox PARC as part of the development of the Information Tapestry system for retrieving documents from a growing corpus. In 1997 GroupLens research led
by Paul Resnick (also from the Xerox PARC team) created MovieLens which uses collaborative filtering to recommend movies to users. The technology created by GroupLens was at the basis of the early Amazon recommendations engine.

2.4.2 Collaborative filtering
Collaborative filtering models the past behaviour of a user and makes predictions based on actions of similar users. It takes over the task of asking a friend for a recommendation, given that you know the friend has the same taste as you, or at least understands your taste. The advantage of a recommender system is that it can 'ask' a lot of people that proved to have the same taste as you do. A further advantage is that the system does not have to understand the content it is recommending. It has shifted the problem from comparing content to comparing users. Amazon and bol.com are both examples of collaborative filtering. Think of the phrase that can often be read on these sites: “others also bought”. Social networks often use collaborative filtering to recommend contacts.

2.4.3 Content based
Next to collaborative filtering another technique often used in modern recommender systems is content based filtering. Content based filtering compares profiles of items in the search domain and recommends items that are similar to items that the user in the past showed an interest in. The similarity is calculated in a process that builds a preference vector of properties for each user and matches this vector to those of items in the search domain. The vectors are made up of discrete values for features and attributes that make up an item profile. Content based recommender systems are especially well used in the movie domain. Sites like Rotten Tomatoes and IMDB use content based recommender systems.

2.4.4 Utility-based recommender systems
Utility based recommender systems do not refine a user's profile over time. Instead they focus on the current need of the user declared by a number of options available. For each request a utility-based recommender system builds a utility function for that request and matches it against items. An advantage of utility-based recommender systems is that they do not suffer from the cold start problem. Furthermore they are sensitive in changes of preferences.

2.4.5 Model based recommender system
Memory based recommender systems keep the entire user-item matrix in memory when building a recommendation. A model based recommender system converts a user-item matrix into a much more compact model. The modelling process is done offline periodically. This way the profile of the
user has a lot of information, but can be reduced in size which has computational advantages.

2.4.6 Hybrid recommender systems

Hybrid recommender systems use more than one technique to make a recommendation. Netflix is a good example of a hybrid system because it uses both collaborative filtering by looking for movies viewed by similar authors and content based filtering by looking for movies with similar profiles. A combination of multiple techniques can perform better than any one specialized technique. The winner of the Netflix competition we mentioned earlier used 107 different algorithmic approaches so come to a recommendation. In general hybrid techniques are used to minimize the problems of any one technique.

2.4.6.1 Weighted hybrid recommender systems

Weighted hybrid systems use weights to combine the different recommendation techniques into one recommendation. The different results from each technique can be counted as votes or scores. The results are then multiplied by the weight of their technique and added together. The advantage is that all capabilities of the system weigh in on the recommendation. A disadvantage is that different items may benefit from different techniques.

2.4.7 Graph based recommender systems

A graph theoretic approach to recommender systems views predictability in terms of a graph. Nodes are users and directed edges indicate the predictability of the source user on the target user. The advantage of this approach is that predictability becomes transitive. If author A has rated an item j and A is connected to X through a relatively short directed path, then X can also to some degree use this rating of j by A (Aggarwal, C., Wolf, J. L., Wu, K.-L., & Yu, P. S., 1999). This approach deals with the sparsity of ratings because it maps a lot more users to the same rating. Naturally social networks can be start-of models for graph based recommender systems. People who are friends are known to influence each others preferences or have become friend because of mutual preferences.

2.4.8 Social network-based recommender systems

In stead of using a graph representation as a model for the predictability of one users actions on another as in graph based recommenders, we propose to use social network analysis as an input variable for the recommendation. The network is therefore not used as a model to be maintained, but as a source of characteristics per user that make up a profile of the user. The place and function of the user in the network become characteristics of a user just like music genre is a characteristic of a CD. Notice that in this approach the items to be recommended and the users requesting the recommendations are the same. In that sense a social network based recommender system is both
collaborative filtering based as content based. The users doing the collaborative filtering are themselves the content (items).

2.4.8.1 Problems with recommender systems
Recommender systems can suffer the following problems:

Cold start: The cold start problem arises when there are not enough user information at the start of use of the system. The system can therefore not make good enough inferences until a certain threshold of user information has been added.

Scalability: Dataset can be very large for recommender systems. The computation time of a single recommendation is therefore important especially in utility based systems (like ours) in which less can be pre-calculated.

Sparsity: Although the number of items in a recommender system is usually very high, the number of ratings is usually very low.

2.4.9 Recap recommender systems
There are two main categories in recommender systems. Content based recommender systems profile content and match users to content with profiles which they are interested in. Recommender systems based on collaborative filtering match interests of users to that of other users to predict other more interests for a user. Modern recommender systems employ multiple techniques to arrive at a recommendation. These systems are called hybrid recommender systems. We propose to use social network metrics as profile values for the content.

2.5 Keyword similarity
In order to calculate the similarity between the knowledge of two authors or between the knowledge of an author and the knowledge being searched we have to calculate the similarity between two set sets of keywords vectors of authors. To calculate the importance of an author-keyword link we use the term frequency – inverse document frequency method. To measure the similarity of the sets of author-keyword links for a single author to that of another author we use vector similarity as is done in many content based recommender systems. See Content based on page 26.

The term frequency – inverse document frequency method is a method used to find the importance of a term in a document normalized by the occurrences of the term in a library of documents. Here we discuss this general theory; in the methodology we discuss our application of this method in our search space. In our search space this translates to the importance of an author-keyword link compared to all author-keyword links in the database.

The vector similarity theory is general enough to be applied to our problem without translation.
2.5.1 TF-IDF

TF-IDF stands for term frequency – inverse document frequency. This is a measure for the importance of a term in a document compared to a set of documents. The TF-IDF increases as the number of instances of the term in a document increases, but is corrected by the number of instances of that term in the entire set of documents. A less common term in the document set with a high frequency in a specific document is important to that document. A very common term in the set of documents with a high frequency in a specific document is less important to that document.

The term frequency is simply the number of times a term \( t \) is used in a document \( d \) – \( tf(t,d) \). The inverse document frequency is a measure for the popularity of a term in the entire document set \( D \). The idf is the logarithm of the number of documents divided by the number of documents that contain the term. See Illustration 5: Inverse document frequency formula.

\[
\text{idf}(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}
\]

*Illustration 5: Inverse document frequency formula*

The denominator of the equation can be 0 therefore it is common to use \( 1+(d \in D : t \in d) \) as the denominator. Now the total \( tf*idf \) is given as \( tf(t,d) \times idf(t,D) \). For a translation of this method to the data domain of this thesis see Calculating keyword similarity on page 42.

2.5.2 Vector similarity

We use the vector similarity to calculate the quantity in which two sets of terms are similar to each other. Given that we now have a set of weighted terms (or weighted author-keyword links) we can easily create vectors of these terms by appending them in an ordered list. If an author A is linked to term1 with TF-IDF 22 and to term2 with TF-IDF 104 we get an author-term vector for author1 of \((22,35)\). For a different author (author B) this vector may be \((34,15)\). If we plot these two points in a two dimensional space and draw a directed line from the origin \( C \) to the two points we get Figure 4: Two vectors.
2.5.3 Similarity

The similarity of these two vectors is defined by the angle (θ) between these two vectors. The smaller the angle between the two vectors, the more these two vector are similar. So if two term vectors of two author have a small angle between them, the authors have the authority on the same terms and have the same amount of authority on these terms. Because the angle between the two term vectors and the similarity of these two term vectors are inversely proportional to each other we use the cosine of the angle to get the similarity. If θ is 0 degrees, then the similarity is the maximum value: 1. If θ is 180, when the two vectors are complete opposites the similarity is the minimal value of -1. The similarity can be expressed by formula presented in Illustration 6: Cosine similarity formula.

\[
\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}
\]

*Illustration 6: Cosine similarity formula*

2.5.3.1 DOT product

The DOT product of two points is defined by:

\[A \cdot B = x1 \cdot x2 + y1 \cdot y2\]

This equation can be extended for 3 dimensions:

\[A \cdot B = x1 \cdot x2 + y1 \cdot y2 + z1 \cdot z2\]

For more than 3 dimensions the equations can be extended in a analog fashion.
2.5.3.2 **Euclidean Distance**
The distance between two points is the length of a straight line between these points. This distance is called the euclidean distance:

1. take the difference between the coordinates of the points
2. square all differences
3. add all squared differences
4. square root the final result

The equation that follows from this algorithm for the points A and C:

\[ d_{AC} = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2} = \sqrt{x_1^2 + y_1^2} \]

Because both vectors CA and CB originate in C (0,0), we can simplify the magnitude these vectors as \( ||A|| \) and \( ||B|| \). Because we only have positive values for the magnitude of the vectors we can only get cosine \( \theta \) values between 0 and 1.

2.5.4 **Recap keyword similarity**
The similarity of authority on or interest in a keyword can be calculated using the term frequency inverse document frequency. This algorithm calculates the number of times a keyword is in a document compared to how many times that keyword is in other documents. Translated to our domain it is the number of times an author is linked to a keyword compared to the number of times other authors are linked to that keyword and the total number of author keyword links. We can calculate the similarity of two authors over multiple keywords by building keyword vectors and calculating the angle between these vectors. The smaller the angle the higher the similarity.
3 Objective, research questions

3.1 Motivation for building a network based recommender system

When selecting research partners scientist have to overcome two limitations. Bounded rationality limits the amount of data a person can process. The vast amount of relevant data causes an information overload for the researcher. The transparency of the network in which the scientists operate is also not optimal. A computer science solution can be found in implementing a recommender system.

On the one hand we see vast amounts of network data being created and being made available in the scientific domain (DSpace, Mendeley, google scholar etc.). People are only starting to use this publicly available data, but when they use it, they use it for analysis, not for intervention. It is being used for social network analysis and for research influence analysis. The social network analysis can identify powerful individuals in a network and the research analysis can identify individuals whose work is influential in the field. There is however a lack of systems that uses this knowledge to recommend strategic co-operations. In the recommender system research field we do not see an emphasis on network based recommendations. We conclude that there is a gap in combining social network analysis and author research influence analysis into a recommender system for co-operation between researchers. For this thesis we therefore set out to build such a scientific co-operation recommender system and ask what the value of its recommendations is; both quantitatively as well as in the perception of its users.

3.2 Objective

Incorporates social network based metrics, like network power, and content based metrics like keyword similarity, in order to make recommendations for future co-authors to researchers.

It is established that research teams with cohesive knowledge have more research success (Lambiotte and Panzarasa, 2009). This is why we use keyword similarities. We also assume that network power is important to get scientific output under the attention of the community. That is why we use network centrality.

3.3 Achieving objective

To achieve the objective of making recommendations we build a weighted hybrid recommender system, because this will enable researchers to request recommendations and to freely interact with the relevant data. The recommender is utility based because users will be able to alter the weights of
the hybrid system and add target keywords for each recommendation. The system is model based because it builds a model of the keyword vectors of users. We will call the system COCOON CORE (COalitions for COOperation Networks | COalition Recommendation).

The system will have the following input and output:

**Input:** Papers with connected co-authors and keywords.

**Output:** Scientific co-operation recommendations.

Based on the input data the system considers two factors in the recommendations:

1. The centrality of an author in the scientific collaboration network (the power of an author)
2. The similarity of keywords to which authors are connected in the network (the content similarity of authors)

**Evaluation data:** For the quantitative evaluation of the recommendations we use the citation data of the papers in the network.

### 3.4 Research Questions

In previous chapters we discussed which characteristics of authors are likely to make for interesting research partners. In this thesis we will research how big of an influence these factors have on good recommendations and how well the recommendations are perceived by the target audience. We formalize this research in the following research questions.

**Research question 1:** *What is the value of research co-operations advised by the recommending system compared to the research co-operations not advised by the system?*

We will look at the extent to which the recommender system is able to predict successful research co-operation. In order to evaluate how heavy the two factors should weight in the recommendation we also ask these subquestions:

**Research question 1.1:** *What is the influence of the centrality of the author in the co-operation network on the quality of the recommendation.*

**Research question 1.2:** *What is the influence of the similarity of interest of two authors on the quality of the recommendation.*

**Research Question 2:** *What is the value of our recommending system perceived by the scientists in the network?*

**Research Question 2.1:** *How do users value COCOON CORE’s recommendation when they can adjust it to their personal preference?*
Research Question 2.2: How do users value COCOON CORE’s recommendation when the algorithm fully focuses on influential peers?

Research Question 2.3: How do users value COCOON CORE’s recommendation when the algorithm fully focuses on like-minded peers?

Research Question 2.4: How do users experience the usability of COCOON CORE?

The first two research questions require us to complete our research objective: build the COCOON CORE system, although only the second research question requires us to build the front end.
4 Methodology

4.1 Introduction to the methodology
We start this chapter by presenting a possible workflow of the COCOON CORE system (4.2). We continue by describing the data we use in the COCOON CORE system and the way in which we collected and calculated this data (4.3). We then give a description of the data in graph terms (4.4). We include network presentations of our data. We conclude this chapter with a description of the experiments we ran to answer our research questions (4.5 and 4.6).

4.2 COCOON CORE workflow
Here we will describe the workflow of a COCOON CORE recommendation. This use case is graphically presented in Illustration 7: COCOON CORE workflow.

Our user Polly would like a recommendation for co-authors for her next paper. She starts by deciding on the topics of her paper. She might look at the keywords page (Illustration 28: Keywords Page on page 80) and specific single keyword pages (Illustration 27: Single Keyword Page on page 79) to see which keywords have gained popularity lately. After deciding on the keywords she is going to use, she selects these keywords in the recommendation settings form on the dashboard page (Illustration 8: Recommendation settings). The topics can be chosen from a set list of keywords that exist in the dataset. Next Polly decides on which influence ratio she would like the recommendation to use. 'Find co-authors with influence' will favour authors with a high betweenness and therefore high network authority. 'Find co-authors with similar interest' will favour
authors that share the interests Polly has based on keywords of her existing submissions and the keywords she entered in the previous step. This setting therefore favours like-minded authors. If Polly's own authority in the network is low she might want to favour co-authors with a high authority to help promote the paper. The two weights are inversely proportional so each slider reacts to the other when changed to offset the change in the other slider. After setting the sliders Polly clicks 'GIVE RECOMMENDATION'.

Illustration 8: Recommendation settings

Now the set of keywords that Polly entered in the recommendation form together with the influence weights are send to the compute recommendation function. This function bundles the keywords that Polly entered with 5 keywords she has a high keyword frequency / inverse author keyword frequency for into the request keywords. It then fetches authors with at least one matching keyword, calculates the keyword similarities between the request keywords and the authors keyword vectors. Next it fetches the pre calculated betweenness values for these authors. It multiplies the normalized recommendation values by their recommendation weights and sends the recommendations back to the page.

Polly can now see an sorted recommendation. The author with the highest overall recommendation score is on top (Illustration 9: Recommendations). Polly can now follow the DSpace link of each recommended author or visit the single author page of a highly recommended author (Illustration 24: Single Author Page on page 73) by clicking his or her name. On that page she can explore the direct network vicinity of the author and see his or her H-index history. Alternatively she can change the recommendation variables (slightly) and run the recommendation again.
4.3 Data Collection

4.3.1 Existing DSpace data

The initial dataset is provided by Rory Sie. It was harvested from a publicly available DSpace repository. Harvesting metadata from a DSpace repository is done using the OAI protocol (Lagoze & Van de Sompel, 2001). Using this protocol metadata is requested over an http request to the DSpace endpoint. Over http the endpoint returns an xml file. This file is parsed using PHP and the scientific output dataset is updated with the results. In our research we imported this database and updated the data structure and data with results from the additional sources.

The DSpace dataset is a set of scientific works by our university. For privacy reasons we will not name the university in this thesis. Most of these works come in the form of Book chapters, articles and conference papers (1113 of 2924), but the output also consist of presentations and other submissions. All works are divided over 1361 authors and 3680 keywords and originate from 9 different departments. For a breakdown see Table 1: Overview of the database (snapshot as of April 2012).
<table>
<thead>
<tr>
<th>Type of submission</th>
<th>Number of submissions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Publications</strong></td>
<td><strong>2924</strong></td>
</tr>
<tr>
<td>Book chapters, articles and conference papers</td>
<td>1113</td>
</tr>
<tr>
<td>Presentations</td>
<td>904</td>
</tr>
<tr>
<td>Other</td>
<td>907</td>
</tr>
<tr>
<td><strong>Authors</strong></td>
<td><strong>1361</strong></td>
</tr>
<tr>
<td><strong>Keywords</strong></td>
<td><strong>3680</strong></td>
</tr>
<tr>
<td><strong>Departments</strong></td>
<td><strong>9</strong></td>
</tr>
</tbody>
</table>

*Table 1: Overview of the database (snapshot as of April 2012)*

For the sake of simplicity we will talk about papers in the rest of this document. We consider it legitimate to abstract from the form of the scientific output, because we are considering cooperation. The output type is of no consequence for the fact that two or more scientist have worked together to generate it. Also, because all this scientific output is publicly available it is possible for other scientific output to cite it. This is important for the valuation of the output.

An overview of the meta data available for each submission in the DSpace dataset can be found in Table 2: Meta data for submissions in University repository. For the purpose of this thesis we are interested in the meta data marked with an X in this table. The data is provided to us in a MySQL database (Sie, R. L. L., 2011). The structure of this database is presented in Illustration 10 below. The red objects are relevant to our experiments and we will ignore the blue objects in the database.

<table>
<thead>
<tr>
<th>Meta data</th>
<th>Relevant to COCOON CORE project</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unique identifier</td>
<td>X</td>
</tr>
<tr>
<td>Timestamp: date and time of submission</td>
<td>X</td>
</tr>
<tr>
<td>Creators: the authors</td>
<td>X</td>
</tr>
<tr>
<td>Descriptions: APA reference, sponsors</td>
<td></td>
</tr>
<tr>
<td>Language</td>
<td></td>
</tr>
<tr>
<td>Title</td>
<td>X</td>
</tr>
<tr>
<td>Subjects: keywords that specify the contents</td>
<td>X</td>
</tr>
<tr>
<td>Type: Journal paper, conference paper, book chapter, etc.</td>
<td></td>
</tr>
</tbody>
</table>
4.3.2 New data collection

On top of the initial data we also collect data from two different sources.

1. The first additional data source we use is google scholar. We calculate the H-index of scientists in our network based on citations in papers publicly available on google scholar. We scrape google scholar to get all citations we can find of submissions in out database. We also scrape the year of the citation which enables us to calculate the H-index of an author per year based on a google scholar citations.

2. The second additional data source we use is Mendeley. Mendeley is a research output network on which, among other data, we can extract the number of times papers by a scientists in our dataset are read by the Mendeley scientific community.
4.3.2.1 Google Scholar
Google scholar is a publicly available search engine that indexes scientific publications from online peer-reviewed journals. Others are available, though none are complete. Publish or Perish is a software tool that uses the citations in the metadata provided in the search results by google to calculate the H-index of scientists. Unfortunately as a windows desktop application Publish or Perish does not allow for scripted input to generate the H-indices for our 1300+ scientists. Moreover Publish or Perish does not give the H-index per year and we need the individual citations per paper to separate our training- and test set. We therefore opted to use html-scraping to calculate the H-index from google scholar search results. A problem with google scholar resides in the fact that the number of results is limited to 1000. We had to divide the scraping process over multiple servers to get all traceable citations of submissions in our database. The process of scraping google scholar is described in get submission citation data on page 65 and get citations per year on page 67.

4.3.2.1.1 Ambiguity in identifiers
When automatically processing names we have to be aware that names are often represented in multiple ambiguous ways. The same author can be cited with a different number of initials. On the other hand two different authors can share the same initials and family name, or even both first name(s) and family name (Sun, X., Kaur, J., Possamai, L., & Menczer, F., 2011). For instance, Michel Klein is an Assistant Professor Artificial Intelligence at the VU University in Amsterdam but also a professor of physical science at University of Pennsylvania and also a Clinical Assistant Professor of Pathology & Laboratory Medicine at Brown University.

In our own dataset authors are uniquely identified and therefore two different authors can not be mistaken for one another due to inconsistent use of initials. We must however consider the choice the creator of the database made when harvesting the Open Doar archive of our university.

The ambiguity in naming conventions is likely not a big issue in our primary dataset. In the our university dataset all authors are identified by their full last name and full first name. We assume therefore that the Open Doar repository of the University greatly reduced the problem of different representations for the same author by using full first names and not initials. This leaves us with just the risk of two scientist sharing an identical name. Within a single university we assume an author to be registered consistently. If however authors are not identified consistently it would still not be likely that two authors with the same last name also partially share initials within one university. This risk is further reduced by the fact that the repository is an aggregation of different departments. The structure of the university repository therefore reduces the risk of ambiguity due to name sharing. This is because it is even less likely that you will find two different people with the same first and last name in the same department as opposed to finding such a pair of scientist in a whole
The way in which scientists' names are represented in the dataset influences our search for these scientists in Google Scholar. When calculating the H-index using Google Scholar we consider B.J. van Engelen as the same author as B. van Engelen. Both results will come up when we search for Bart Jan van Engelen. Because of this we risk identifying different authors with the same last name as the same author. This loose identification of authors does however minimize the chance of falsely identifying one author as two different authors, if the author is first registered with one initial and later with two. An author with a name double will have an advantage when counting the citations. We therefore manually checked the top ranking scientists - in terms of influence - to detect this bias if it occurs in our dataset. A further reduction of the problem comes from the fact that we separately search for each individual paper of an author combining the name with a title in the advanced search of Google Scholar. The chance of two different authors with the same name writing two different papers with similar titles is negligible.

4.3.2.2 Mendeley – Reader Meter
We extract Mendeley reads metrics by using an external website (http://readermeter.org/). We created a PHP script that sends http requests to the reader meter server. The GET request includes the first and last name of one of the authors in the university co-author network and the requested return format (json). The server returns a json object which is parsed. The return values are inserted into the adjusted MySQL COCOON CORE database. The script repeats this for all scientists in our dataset. We use these metrics to validate the Google metrics we calculate from Google Scholar. For more information on the process see get mendeley author metrics on page 64.

4.3.3 New data calculation
4.3.3.1 Calculating the H-index
The H-index is based on citations of papers. We will check for these citations on Google Scholar. The workflow for calculating the H-index per author per year is as follows. We harvest all citations of all years for all submissions in the DSpace database. Then we calculate how many submissions an author has in each year and how many citations each of those submissions got. This information is used to calculate the H-index. For an in depth description of this process see Compute H-index on page 59.

4.3.3.2 Calculating betweenness
The betweenness of an author is based on its place in the scientific co-operation network. To calculate the betweenness we have to generate the scientific co-operation network first. We do this by taking all authors with submissions as nodes and adding an edge between a pair of nodes if the two authors have co-authored a submission together. If we see a author as a node in this author-
author network we can see his betweenness as the total number of fractions of shortest paths between all nodes that pass through this node. For more information on this process see Compute betweenness on page 59.

4.3.3.3 Calculating keyword similarity
Calculating the keyword similarity is done by calculating the similarity of two author-keyword akf/ikf vectors. Calculating the similarity between the vectors in done in the way described in Vector similarity on page 29. The vectors we are comparing consist of keyword weights. Each weight describes the authority the author has (or search vector has) on a certain keyword. To calculate this authority of an author on a keyword we use a version of the Term Frequency / Inverse Document Frequency method described in TF-IDF on page 29. For our application we interpreted the Term Frequency as the number of times this specific author is linked to this specific keyword. We interpreted the Inverse Document Frequency as the total number of links between any author and any keyword divided by the total number of links between any author and this specific keyword. For more information on this process see Compute author-keyword frequency/inverse keyword frequency vector similarity on page 61.

4.3.3.4 Calculating the overall recommendation score
The overall COCOON CORE recommendation score of a co-operation is based on two influence factors: the betweenness centrality of an author (the power the author can exert on the network) and the keyword similarity (the similarity between the author and the user). These two factors are normalized and multiplied by their importance weights selected by the users. The sum of these scores form the recommendation score of the author. The total formula for the recommendation score for one author is:

Normalized betweenness * weight betweenness + normalized keyword similarity * weight keywords similarity.

The complete recommendation is an sorted list of author – recommendation score pairs. The authors with the highest recommendation scores are the recommended authors to co-operate with.

Example:
Here we give an example of the recommendation score calculations. It is a calculation of one prospect co-author. The total recommendation is a sorted list of all prospect co-authors. All authors with a connection to at least one target keyword is considered a prospect co-author. We calculate the overall recommendation score of author B for author A.

First we lookup the betweenness for the given year for all prospect authors. The normalized betweenness of an author is the log of its betweenness divided by the log of the maximal betweenness of all prospects. Say that the betweenness os B is 32930 an the maximal betweenness
of all prospects in 232076 than the normalized betweenness of author B is 0.84194988038.

The normalized keyword similarity between author A and B of the author is the log of the keyword similarity between author A and B. The keyword similarity is already on a scale from 0 to 1 so we don't have to rescale like we did with the betweenness. Say that this normalized keyword similarity is 0.878112234641. Given a weight distribution setting of 80/20 we get an overall recommendation score for author B of $0.84194988038 \times 80 + 0.878112234641 \times 20 = 84.7735857062$ in a recommendation for author A.

4.4 DSpace data description in graph terms

4.4.1 Author graphs

For each submission in the database the authors are retrieved and authors that co-wrote a paper are connected with an author-author link. All co-authors of a paper therefore automatically form a cluster. All authors and all co-author links combined form a co-author network. A graphical representation of this co-author network is presented in Illustration 11: Co-author network 2012. The graph that is made up of our data is an author-author graph. That is to say that all nodes in our graph are authors. Edges in our graph represent scientific co-operations between these authors. A scientific co-operation is defined as writing a paper together. Scientist can write multiple papers together. These multiple papers are represented by multiple edges between the nodes that represent these scientists. We call these edges that have an identical source and identical target identical edges. Co-authoring papers is a symmetric relationship; if person A co-authors a paper with person B, person B by definition co-authors a paper with person A. The direction of the edges in our dataset do not hold extra information and we therefore consider the graph to be undirected. For our purpose we can have a broader definition of identical edges. Edges are identical if they share the same two nodes no matter which node is the source and which node is the target.

Identical edges have an impact on the betweenness. If we merge identical edges, the degree of the authors that wrote multiple papers together would drop. The betweenness of these authors could also drop if their connection is on a shortest path. In such a case the identical edges each count for one shortest path, benefitting the betweenness of both authors.

When we simplify graphs the connection of authors is binary. Either two authors are connected or they are not connected regardless of the number of papers these authors have written together. The number of papers that two authors have written together – or the weight of their connection – is however relevant information. One can imagine that the influence of two authors on each other is correlated to the number of papers they have written together.
We use both kinds of author-author graphs. We use the simplified graphs when we just want to see if two people are connected. This makes the graphs more transparent. We use graphs with identical edges when we calculate network variables and when we look at the entire graph in stead of a subgraph to facilitate identifying strong ties.

4.4.2 Keyword graphs

The keyword dataset consists of keywords that are registered in the metadata of paper. Two keywords are connected if they are connected to the same paper. A graphical representation of the keyword graph is presented in Illustration 12: Keyword network 2012.

Illustration 12: Keyword network 2012
4.5 Research question 1

4.5.1 Introduction
To test the workings of our system, we use two types of evaluation: an ex ante evaluation for the first research question and a user satisfaction evaluation for the second research question. For the first question we will perform calculations on hard data we harvest from different sources (Open Doar endpoint, Mendeley reads, google scholar).

4.5.2 Design and procedure
4.5.2.1 Time splitting the author network
The data we have is essentially a collection of research co-operations. Each paper is a co-operation between two or more scientists. By changing the maximum submission year of papers that are allowed for calculation of the co-operation network we time split our data. We will call this maximum submission year the cut-off year. By time splitting our data we enable ourselves to use one co-operation network as a training set and the next as a test set.

Notice that co-operation network training sets generated this way form subsets of their successors. The network for 2005 is a subset of the network for 2006 which is a subset of the network for 2007 etc. This relation is transitive. The number of authors and author-author links can only increase in
these networks. See Illustration 13 and Illustration 14 for a graphical representation of two author-
author networks time split (or cut off) based on time. We use this splitting in the front end for users
to be able to see the network states on different moments in time. For our evaluation however we
use the complement of the training set as the test set. This complement includes all co-operations
based on submissions between the the 1st of January of cut off year +1 and the 31st of December of
2012.

4.5.2.2 evaluation
Experiments 1.1 and 1.2 answer research question 1: “What is the value of research co-operations
advised by the recommending system compared to the research co-operations not advised by the
system?”. The experiments are ex ante evaluations. These ex ante evaluations are done by using the
dataset as both a training- and test set. Ex ante evaluation therefore also implies that we split up the
database in two parts. We use the first part as training set, and the second part as test set. We make
this split in the middle after the year 2008. This way we can use 2005-2008 as the training- and
2009-2012 as the test set.

The major advantage of using ex ante evaluation is that one can evaluate the workings of a system
without interfering in daily practice. This saves time and money, and reduces bias of multiple
interventions by the same system.

One baseline performance would be for the system to be able to recommend the co-operations in the
test set after training on the training set. This would indicate that the recommendations are as good
(or bad) as the co-operation choices the scientists make on their own.

We would however like to improve on the current co-operation choices so we need to make a
distinction between successful co-operations and less successful co-operations in our dataset.
In this respect the optimization task is to maximize the number of predicted successful co-
operations in the network of 2009-2012. At the same time we want to minimize the number
recommended co-operations that are unsuccessful in this test set. Because of the fact that we can not
measure the influence of co-operations in the past that were not formed, we must concentrate on the
coop-operations that were formed. Although there might be research co-operations that were not
formed, but would have been successful, we can not incorporate those co-operations because we do
not have the positive outcomes of these co-operations to support their inclusion. If however the
system recommends a significant amount of successful recommendations, the confidence in the
quality of the recommendations that can not be evaluated also increases. Ideally the
recommendations would fall in the category Formed and Successful co-operations. After that it is
interesting to look at recommendations that were not formed. We can not check their success, but
these are the recommendations that could help to make the overall research influence rate higher.
This can not be proven by this or any experiment based on this data. No assumptions about the
coop-erations that did not form should be made. Recommendations that fall in the category Formed and Unsuccessful are undesirable. See Table 3.

<table>
<thead>
<tr>
<th></th>
<th>Successful</th>
<th>Unsuccessful</th>
</tr>
</thead>
<tbody>
<tr>
<td>Formed</td>
<td>Higher Recommended</td>
<td>Lower Recommended</td>
</tr>
<tr>
<td>Not Formed</td>
<td>No Data</td>
<td>No Data</td>
</tr>
</tbody>
</table>

*Table 3: Co-operation Categories*

The output of the recommender are recommendation scores. The co-operation recommendation is
therefore not binary (either recommended or not). Although it is transparent to think in the
categories of Table 3 and the abstract goal is as presented there, we reframe the optimization task to
suit the continues scores in stead of the categories. We state that it is desirable for the average
recommendation score for co-operations based on the training set that turned out to be formed and
successful in the test set to be high. And inversely it is desirable for the average recommendation
score for co-operations based on the training set that turned out to be formed and unsuccessful in
the test set to be low.

After time splitting our data we can see which co-operations in the test set were advised by our
system based on the trainings dataset. If we compare the co-operations that are advised by our
system to the co-operations that were not advised by our system we can see the difference in value
that the scientists got out of participating in the advised co-operations as supposed to the value other
scientist got from participating in co-operations not advised by the system.
4.5.3 Research question 1 - Using Δ H-index

4.5.3.1 Design and procedure
We take influence of scientific output as a measure of value in a scientific co-operation network. Influence of the scientific output of a scientist is quantified by the H-index of scientists. As a target variable for successful co-operations we use the Δ H-index. We will evaluate the H-index delta for researchers that participated in recommended co-operations compared to the H-index delta for researchers that participated in non-recommended co-operations.

The basic idea behind this is that scientists that participate in more valuable scientific co-operation teams will produce scientific output that is of more influence on the scientific community than that of scientists that participate in less valuable scientific co-operation teams. For experiment 1.1 we opted to use the H-index as measure for scientific output of an author as it addresses all problems with simpler measures and is the strictest of the more complex measures discussed in Scientific quality measures on page 7.

Ideally the authors that participated in recommended co-operations would see a high positive Δ H-index. Authors that did not participate in recommended co-operations would see a neutral or low positive Δ H-index values. See table 4 below. Because H-indexes are always calculated over all citations of an author it can only grow or stay the same over time. We will therefore not see any negative Δ H-index values.

<table>
<thead>
<tr>
<th>Recommended based on training set</th>
<th>Not Recommended based on training set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Formed in test set</td>
<td>High positive Δ H-index</td>
</tr>
<tr>
<td></td>
<td>Neutral of low positive Δ H-index</td>
</tr>
</tbody>
</table>

*Table 4: Author Δ H-index categories*

4.5.3.2 Data analysis
We analyse the data by calculating the average recommendation score of connections in test set based on training set for both authors with a positive Δ H-index and authors with a neutral Δ H-index.

4.5.4 Research question 1 - Using Citations
A disadvantage of using the Δ H-index as target variable is that we have to ascribe an increase in H-index to all co-operations. An increase in H-index will likely be the result of citations of a subset of the cooperations that an author has participated in. Just using the H-index however forces us to consider all co-operations that the author has participated in as valuable, because in this setup we do not use citations of individual papers. In experiment 1.2 we therefore use the total number of citations of papers authored with the same co-author as target variable.
4.5.5 Research question 1.1

4.5.5.1 Data analysis
Calculating the correlation between a scientists betweenness and his or her H-index. This is basically an indication for the inherit correlation between the network position of a scientist and the influence of his or her scientific output. The higher the correlation the more predictive power the betweenness has on the H-index and therefore the more influence it should have on the recommendation. Experiment 1.3 answers research question 1.1 - “What is the influence of the centrality of the author in the co-operation network on the quality of the recommendation.” - together with experiments 1.1 and 1.2

4.5.6 Research question 1.2

4.5.6.1 Design and procedure
Because of the way the network is generated there is an inherent bias for people with similar interest to share a connection. Two authors are connected if they share a submission. But because they share a submission, they also both are connected to the keywords connected to that submission. If authors have multiple submissions with different other authors the interests will likely not be exactly the same as their co-authors, but the keyword similarity will on average be higher than the keyword similarity between the author and an author to which he or she in not connected. In this situation it is interesting to look at the cluster coefficient of the author. If the cluster coefficient is high the keyword similarity of connected authors will be higher than when the cluster coefficient is low. In other words: the higher the cluster coefficient of an author, the higher its keyword similarity with its neighbours. This simply follows from the way the network is set up.

4.5.6.2 Data analysis
We are going to calculate the correlation between the cluster coefficient of an author and the H-index of an author. If this correlation is high, then presumably the correlation between the keyword similarity and the H-index is also high. Experiment 1.4 answers research question 1.2 - “What is the influence of the similarity of interest of two authors on the quality of the recommendation.” - together with experiments 1.1 and 1.2.

In graph theory experiment 1.3 and 1.4 in a way calculating opposite forces or roles in the network. Hubs have low clustering coefficients because their edges are more often bridges. Clusters are more common in the periphery. So do hubs (with a high betweenness) have a high H-index or do authors that are part of a cluster (high cluster coefficient) have a high H-index? To check whether these two metrics do indeed have an inversely proportional relationship we check their correlation to each other. This is done in Experiment 1.5.
4.5.7 Influence of different recommender techniques

The afore mentioned calculations give clues to how big an influence betweenness should have on the recommendation compared to the keyword similarity of authors, but it does not verify which setting works best on this data. This is not a research question, but it is a question closely related to research question 1.1 and 1.2.

4.5.7.1 Design and procedure

To answer this question we will run test evaluations of the recommender with different weights for both factors to see which give the best results. The results of these runs can be found under experiment 1.1 and 1.2. The recommendations are deterministic. Every time the same recommendations settings are put in, the same recommendation comes out. We therefore do not have to evaluate the results over multiple runs. The different configuration settings are summed up in Table 5 below.

<table>
<thead>
<tr>
<th>Configuration number</th>
<th>Betweenness setting</th>
<th>Keyword Similarity setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>90</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>80</td>
</tr>
<tr>
<td>4</td>
<td>30</td>
<td>70</td>
</tr>
<tr>
<td>5</td>
<td>40</td>
<td>60</td>
</tr>
<tr>
<td>6</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>7</td>
<td>60</td>
<td>40</td>
</tr>
<tr>
<td>8</td>
<td>70</td>
<td>30</td>
</tr>
<tr>
<td>9</td>
<td>80</td>
<td>20</td>
</tr>
<tr>
<td>10</td>
<td>90</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>100</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5: Recommendation configuration settings

4.5.7.2 Data analysis

Using the same results as in research questions 1.1 and 1.2, but looking specifically at the configuration settings with the optimum score.

4.6 Research question 2

4.6.1 Introduction

The way we are going to answer research question number 2 - “What is the value of our recommending system perceived by the scientists in the network?” - is by polling the scientists in the network. For this research question we do not restrict ourselves to past data. Instead we present the system to the scientists and ask how they value the recommendations. The overall usability of the
system can influence the perceived value of the recommendations given by the system. We therefore measure this usability by means of a standard usability test.

### 4.6.2 Participants

The experiment has 24 participants out of the 89 currently employed by the department. We do not know the total number of current researchers in all departments of our university, but we do know that there are 566 authors in our database that have contributed a submission after January 1st 2011. All participants are currently active in the university and have contributed at least one submission. The participants have been with our university between 1 and 35 years (M = 9.48; SD = 7.84). The group consists of 13 males and 10 females. The occupation of the participants ranges from PhD candidate to full professor.

### 4.6.3 Experiment 2.1 - Recommendation valuation

#### 4.6.3.1 Design and procedure

For Experiment 2.1: Perceived value of the recommendations results the participants received instructions in a document which detailed how to perform the tasks assigned to them. We described how they should log in and how to operate the dashboard to generate a recommendation.

The participants were then asked to request six recommendations. Three recommendations logged in as themselves and three recommendations logged in as an other user (the same other user for all participants). We use this default user to benchmark the values. If the valuations of both sessions are the same, the influence of the user specific keywords on the perceived value of the recommendations are not big. The default user results also compensate for the bias of participants to give a higher valuation when they know that the results are tailored to them. The three requests per login session differed in the slider settings. We set up the two login sessions to account for different performance for different users. Each researcher is linked to a different part of the keyword network and the quality of the results might therefore differ per user. Furthermore we split up the participants in two groups which performed the login sequence in opposite order. See Table 6: Task sequence for two groups below. This is done to check against a sequence bias. The order in which the login sessions are performed could influence the scores of each session. If the scores for the two different groups are not significantly different we can conclude that there is no sequence bias in this experiment.

<table>
<thead>
<tr>
<th>Group 1 (N=12)</th>
<th>Default user recommendation</th>
<th>Individual recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 2 (N=12)</td>
<td>Individual recommendation</td>
<td>Default user recommendation</td>
</tr>
</tbody>
</table>

*Table 6: Task sequence for two groups*
For each login session participants were first asked to put the influence slider to 100% (setting 11 Table 5). This experiment will answer research question number 2.2: “How do users value COCOON CORE’s recommendation when the algorithm fully focuses on influential peers?”. Then they were asked to put the interest slider to 100% (setting 1 Table 5). This experiment will answer research question number 2.3: “How do users value COCOON CORE’s recommendation when the algorithm fully focuses on like-minded peers?”. Finally they were asked to set the sliders freely. This experiment will answer research question number 2.1: “How do users value COCOON CORE’s recommendation when they can adjust it to their personal preference?”. After setting one slider, the other automatically adjusts to counterbalance the change. This way the sum of both sliders is always 100%.

For each recommendation request the participants were asked to rate the recommendation on a Likert scale from 1 to 5. These questions can be found in Appendix C – Recommendation valuation questionnaire.

4.6.3.2 Data analysis
We analyse the data by looking at median scores on each question as well as the distribution of scores for each question.

4.6.4 Experiment 2.2 - Usability test
4.6.4.1 Design and procedure
To measure the usability of the system we opted to use the System Usability Scale (SUS) in Experiment 2.2: SUS Usability questionnaire results. This experiment will answer research question number 2.4: “How do users experience the usability of COCOON CORE?”. SUS is an industry standard usability test with over 5000 users and 500 reported studies. SUS measures usability and learnability. The test can be found in Appendix D – SUS Usability questionnaire. It consists of 10 questions which are to be rated on a scale of 1 to 5. Questions 1-3 and 5-9 address usability and questions 4 and 10 address learnability. Answers are given on the same Likert scale we used for the recommendation valuation questions.

Ratings on each question of the SUS test are reversed for each subsequent question. For odd questions a high score is good. For even questions a low score is good. To arrive at a final score for the test we:

1. take each rating for odd-numbered questions minus 1 (rating 4 results in score 3)
2. take each rating for even-numbered questions and subtract them from 5 (rating 4 results in score 1)
3. multiply all scores by 25
Each question now has a score between 0 and 100. The average score of all scores is the score for the SUS test. On average a system gets a score of 68 points on a SUS test.

Both experiment 2.1 and 2.2 answer research questions 2, 2.1, 2.2 and 2.3.

4.6.4.2 Data analysis
We analyse the data by looking at median scores on each question as well as the distribution of scores for each question.
5 Implementation

5.1 Introduction
In this chapter we discuss the implementation of the COCOON CORE system. We give an overview of the structure of the database and the information that is stored in the database. We give a detailed description of how we harvested new data, like citations per paper, and calculate new derived data, like the H-index and the corrected author-keyword frequency. We present each process of the recommender both offline and online. From calculating network variables in R to generating live interactive graph views in the browser. We end with a description of the front end the external libraries we used and the data types we employ.

5.2 System architecture
Our implementation is centred around the COCOON CORE database. This is the original database provided by Rory Sie extended with additional data from external sources and calculated, derived data. The data stored in this new structure is retrieved and calculated by offline processes and used by online processes to generate recommendations, graphs and tables. Below all processes are centred around the central database divided in different swim lanes. We will discuss them in separate chapters. COCOON CORE is available online (username required) (http://bartjan.me/recommender/implementation/login.php).
5.3 Databases
We include an overview of the database structure to show how we store the information we need for the recommendation. The structure is also an overview of which sub-metrics are needed to calculate our target matrices mentioned in the methodology.

5.3.1 Internal COCOON CORE Database
The Internal database is implemented in MySQL. We use the submissions to derive knowledge.

In addition to calculate variables on the fly (for instance author-author relationships in a specific year) we also pre calculate derived data and store that in the database. The database structure with our additions is depicted in Illustration 16: COCOON CORE database Entity Relationship Model below.
The folded tables at the top of the structure represent tables that were in the original database structure which we do not use. We will therefore not discuss them here any further.

5.3.1.1 Original tables
The central column consists mainly of tables that were included in the original database. The contents of these tables can be found in Appendix G – Original database tables. Here we will focus on the new tables which contain information specifically harvested and calculated for the recommender.

Illustration 16: COCOON CORE database Entity Relationship Model
<table>
<thead>
<tr>
<th>author_metrics</th>
<th>author_metrics_id</th>
<th>Unique Identifier</th>
<th>ADDED</th>
</tr>
</thead>
<tbody>
<tr>
<td>author_id</td>
<td>Unique Identifier of author</td>
<td>ADDED</td>
<td></td>
</tr>
<tr>
<td>metric_id</td>
<td>Unique Identifier of metric</td>
<td>ADDED</td>
<td></td>
</tr>
<tr>
<td>year</td>
<td>Cutoff year for submissions used in metric calculation</td>
<td>ADDED</td>
<td></td>
</tr>
<tr>
<td>value</td>
<td>Value of metric</td>
<td>ADDED</td>
<td></td>
</tr>
</tbody>
</table>

Table 7: ER - author_metrics

<table>
<thead>
<tr>
<th>metrics</th>
<th>metric_id</th>
<th>Unique Identifier</th>
<th>ADDED</th>
</tr>
</thead>
<tbody>
<tr>
<td>metric_name</td>
<td>Name of metric</td>
<td>ADDED</td>
<td></td>
</tr>
</tbody>
</table>

Table 8: ER - metrics

<table>
<thead>
<tr>
<th>keyword_metrics</th>
<th>keyword_metrics_id</th>
<th>Unique Identifier</th>
<th>ADDED</th>
</tr>
</thead>
<tbody>
<tr>
<td>keyword_id</td>
<td>Unique Identifier of keyword</td>
<td>ADDED</td>
<td></td>
</tr>
<tr>
<td>metric_id</td>
<td>Unique Identifier of metric</td>
<td>ADDED</td>
<td></td>
</tr>
<tr>
<td>year</td>
<td>Cutoff year for submissions used in metric calculation</td>
<td>ADDED</td>
<td></td>
</tr>
<tr>
<td>value</td>
<td>Value of metric</td>
<td>ADDED</td>
<td></td>
</tr>
</tbody>
</table>

Table 9: ER - keyword_metrics

<table>
<thead>
<tr>
<th>author_keyword_links</th>
<th>id</th>
<th>Unique Identifier</th>
<th>ADDED</th>
</tr>
</thead>
<tbody>
<tr>
<td>author_id</td>
<td>Unique Identifier of author</td>
<td>ADDED</td>
<td></td>
</tr>
<tr>
<td>keyword_id</td>
<td>Unique Identifier of keyword</td>
<td>ADDED</td>
<td></td>
</tr>
<tr>
<td>submission_date</td>
<td>Date of submission linking the author to the keyword</td>
<td>ADDED</td>
<td></td>
</tr>
</tbody>
</table>

Table 10: ER - author_keyword_links
### author_keyword_frequency_metrics

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Added</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Unique Identifier</td>
<td>ADDED</td>
</tr>
<tr>
<td>author_id</td>
<td>Unique Identifier of author</td>
<td>ADDED</td>
</tr>
<tr>
<td>keyword_id</td>
<td>Unique Identifier of keyword</td>
<td>ADDED</td>
</tr>
<tr>
<td>year</td>
<td>Cutoff year for submissions used in metric calculation</td>
<td>ADDED</td>
</tr>
<tr>
<td>author_keyword_frequency</td>
<td>Number of times the author is linked to the keyword</td>
<td>ADDED</td>
</tr>
<tr>
<td>inverse_keyword_frequency</td>
<td>The frequency of the keyword linked to a submission relative to the total number of submissions</td>
<td>ADDED</td>
</tr>
<tr>
<td>akf_ikf</td>
<td>Author-keyword freq / inverse keywords frequency. How much is this author linked to this keyword relative</td>
<td>ADDED</td>
</tr>
</tbody>
</table>

Table 11: ER - author_keyword_frequency_metrics

### submissions_citations

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Added</th>
</tr>
</thead>
<tbody>
<tr>
<td>submissions_citations_id</td>
<td>Unique Identifier</td>
<td>ADDED</td>
</tr>
<tr>
<td>submission_id</td>
<td>Unique Identifier of submission</td>
<td>ADDED</td>
</tr>
<tr>
<td>citation_title</td>
<td>Title of citing paper</td>
<td>ADDED</td>
</tr>
<tr>
<td>citation_title_link</td>
<td>Link to citing paper</td>
<td>ADDED</td>
</tr>
<tr>
<td>citation_year</td>
<td>Year of publication of citing paper</td>
<td>ADDED</td>
</tr>
<tr>
<td>citation_authors</td>
<td>Authors of citing paper</td>
<td>ADDED</td>
</tr>
</tbody>
</table>

Table 12: ER - submissions_citations

### submissions_metadata

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Added</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Unique Identifier</td>
<td>ADDED</td>
</tr>
<tr>
<td>submission_id</td>
<td>Unique Identifier of submission</td>
<td>ADDED</td>
</tr>
<tr>
<td>title</td>
<td>Title of submission</td>
<td>ADDED</td>
</tr>
<tr>
<td>date</td>
<td>Date of submission</td>
<td>ADDED</td>
</tr>
<tr>
<td>google_scholar_citations</td>
<td>Number of citations counted by google scholar</td>
<td>ADDED</td>
</tr>
<tr>
<td>citations_link</td>
<td>Link to first google scholar citations page</td>
<td>ADDED</td>
</tr>
<tr>
<td>checked</td>
<td>Flag used to see if the submission has been looked up on google scholar</td>
<td>ADDED</td>
</tr>
<tr>
<td>last_citations_run</td>
<td>Date of last run for individual citations for this submission</td>
<td>ADDED</td>
</tr>
</tbody>
</table>

Table 13: ER - submissions_metadata
5.3.2 External Databases
We do not have the structure of the external databases we use. We gather data from these databases using API's in the case of Mendeley and a web interface in the case of google scholar citations.

5.4 Calculations
Here we include an explanation of the three most important calculations for the recommendation. This makes it easy to recreate the information on which we draw our conclusions.

5.4.1 Compute betweenness
In order to calculate the betweenness we use the get_graph REST server we build to export the author-author graphs per year and import the graph in the R statistical computing environment (http://www.r-project.org/). We used the igraph package for R (http://igraph.sourceforge.net/) to calculate the betweenness of the authors in the author author graph per year and posted the results back to our database (http://igraph.sourceforge.net/doc/R/betweenness.html).

Igraph calculates the betweenness of a node according to the following algorithm:

given Graph G = \{V,E\}, where V is the set of Vertices (Nodes) in the Graph and E is the set of Edges in the Graph.

1. for each node (V3)
   1. for each pair of nodes (V1, V2) in the graph (G)
      1. calculate the fraction of all the shortest paths between these nodes (V1,V2) that passes through V3
      2. Add this fraction to the total betweenness

Updating the betweenness of all nodes for different years is done with the use of a bash script:

- query the REST server (curl). (Multiple queries for multiple years)
- redirect output of curl to file (> ou_author_author_graph_<year>_<generated time>.xml)
- Run R script to calculate betweenness of nodes per year. (R CMD BATCH ./calculate_betweenness.R)
- run PHP script to post betweenness to DB

For the technical implementation see Calculation processes on page 62.

5.4.2 Compute H-index
For our calculation of the H-index of an author we need to have the submissions of that author and
For this paragraph we will assume that we have an array of submissions with the corresponding number of citations for those submissions. How we create this array from our database can be found in 'calculate_author_metrics_from_citations.php'. Below we will include the output of a run of the algorithm at each stage. The run is for author_id 1 and year 2005. Our data at this starting point will look like this:

\[ \text{A) submissions\_citations}[x] = y; \] submission x has y citations. Actually the submission id is of no consequence for the H-index calculation so we can suffice with: there is a submission with y citations. The keys in the following output are not submission id's they are just auto incremented.

**Input:**

\[ \begin{align*}
0 & \Rightarrow 1, \\
1 & \Rightarrow 1, \\
2 & \Rightarrow 1, \\
3 & \Rightarrow 1, \\
4 & \Rightarrow 1, \\
5 & \Rightarrow 1, \\
6 & \Rightarrow 1, \\
7 & \Rightarrow 1, \\
8 & \Rightarrow 1, \\
9 & \Rightarrow 1, \\
10 & \Rightarrow 1, \\
11 & \Rightarrow 1, \\
12 & \Rightarrow 1, \\
13 & \Rightarrow 1, \\
14 & \Rightarrow 1, \\
15 & \Rightarrow 1, \\
16 & \Rightarrow 1, \\
17 & \Rightarrow 1, \\
18 & \Rightarrow 1, \\
19 & \Rightarrow 2, \\
20 & \Rightarrow 2, \\
21 & \Rightarrow 2, \\
22 & \Rightarrow 2, \\
23 & \Rightarrow 2, \\
24 & \Rightarrow 2, \\
25 & \Rightarrow 2, \\
26 & \Rightarrow 2, \\
27 & \Rightarrow 2, \\
28 & \Rightarrow 2, \\
29 & \Rightarrow 2, \\
30 & \Rightarrow 2, \\
31 & \Rightarrow 2, \\
32 & \Rightarrow 2, \\
33 & \Rightarrow 2, \\
34 & \Rightarrow 2, \\
35 & \Rightarrow 3, \\
36 & \Rightarrow 3, \\
37 & \Rightarrow 3, \\
38 & \Rightarrow 3, \\
39 & \Rightarrow 3, \\
40 & \Rightarrow 3, \\
41 & \Rightarrow 3, \\
42 & \Rightarrow 3, \\
43 & \Rightarrow 3, \\
44 & \Rightarrow 4, \\
45 & \Rightarrow 4, \\
46 & \Rightarrow 4, \\
47 & \Rightarrow 4, \\
48 & \Rightarrow 5, \\
49 & \Rightarrow 5, \\
50 & \Rightarrow 5, \\
51 & \Rightarrow 5, \\
52 & \Rightarrow 6, \\
53 & \Rightarrow 7, \\
54 & \Rightarrow 7, \\
55 & \Rightarrow 7, \\
56 & \Rightarrow 8, \\
57 & \Rightarrow 8, \\
58 & \Rightarrow 11, \\
59 & \Rightarrow 11, \\
60 & \Rightarrow 11, \\
61 & \Rightarrow 11, \\
62 & \Rightarrow 11, \\
63 & \Rightarrow 16, \\
64 & \Rightarrow 21, \\
65 & \Rightarrow 22, \\
66 & \Rightarrow 27, \\
67 & \Rightarrow 37, \\
68 & \Rightarrow 38, \\
69 & \Rightarrow 43, \\
70 & \Rightarrow 47, \\
71 & \Rightarrow 216
\end{align*} \]

It is good to notice that the values have been sorted.

The H-index states that:

\[ A \text{ scientist has index } h \text{ if } h \text{ of his or her total number of } Np,\text{tot} - \text{have at least } h \text{ citations each and the other } (Np,\text{tot} - h) \text{ papers have } h \text{ or less citations each.} \]

It is therefore clear that we need a data structure like this:

\[ \text{C) submissions\_with\_at\_least\_x\_citations}[x] = y; \] there are y submissions with at least x citations.

To get from A to C we first create the following data structure:

\[ \text{B) submissions\_with\_x\_citations}[x] = y; \] these are y submissions with x citations.

We use array_count_values to transform A into B. This function counts the number of times a value is present in an array.

**Output:**

\[ \begin{align*}
1 & \Rightarrow 19, \\
2 & \Rightarrow 16, \\
3 & \Rightarrow 8, \\
4 & \Rightarrow 5, \\
5 & \Rightarrow 4, \\
6 & \Rightarrow 1, \\
7 & \Rightarrow 3, \\
8 & \Rightarrow 1, \\
9 & \Rightarrow 6, \\
10 & \Rightarrow 1, \\
11 & \Rightarrow 1, \\
12 & \Rightarrow 1, \\
13 & \Rightarrow 1, \\
14 & \Rightarrow 1, \\
15 & \Rightarrow 1, \\
16 & \Rightarrow 1, \\
17 & \Rightarrow 1, \\
18 & \Rightarrow 1, \\
19 & \Rightarrow 2, \\
20 & \Rightarrow 2, \\
21 & \Rightarrow 2, \\
22 & \Rightarrow 2, \\
23 & \Rightarrow 2, \\
24 & \Rightarrow 2, \\
25 & \Rightarrow 2, \\
26 & \Rightarrow 2, \\
27 & \Rightarrow 2, \\
28 & \Rightarrow 2, \\
29 & \Rightarrow 2, \\
30 & \Rightarrow 2, \\
31 & \Rightarrow 2, \\
32 & \Rightarrow 2, \\
33 & \Rightarrow 2, \\
34 & \Rightarrow 2, \\
35 & \Rightarrow 3, \\
36 & \Rightarrow 3, \\
37 & \Rightarrow 3, \\
38 & \Rightarrow 3, \\
39 & \Rightarrow 3, \\
40 & \Rightarrow 3, \\
41 & \Rightarrow 3, \\
42 & \Rightarrow 3, \\
43 & \Rightarrow 3, \\
44 & \Rightarrow 4, \\
45 & \Rightarrow 4, \\
46 & \Rightarrow 4, \\
47 & \Rightarrow 4, \\
48 & \Rightarrow 5, \\
49 & \Rightarrow 5, \\
50 & \Rightarrow 5, \\
51 & \Rightarrow 5, \\
52 & \Rightarrow 6, \\
53 & \Rightarrow 7, \\
54 & \Rightarrow 7, \\
55 & \Rightarrow 7, \\
56 & \Rightarrow 8, \\
57 & \Rightarrow 8, \\
58 & \Rightarrow 11, \\
59 & \Rightarrow 11, \\
60 & \Rightarrow 11, \\
61 & \Rightarrow 11, \\
62 & \Rightarrow 11, \\
63 & \Rightarrow 16, \\
64 & \Rightarrow 21, \\
65 & \Rightarrow 22, \\
66 & \Rightarrow 27, \\
67 & \Rightarrow 37, \\
68 & \Rightarrow 38, \\
69 & \Rightarrow 43, \\
70 & \Rightarrow 47, \\
71 & \Rightarrow 216
\end{align*} \]

Because the values were sorted in data structure A the keys are sorted in data structure B.

To get from B to C we copy each key-value pair to a new array and add the value of the current
iteration to all values with a lower key. This way we get a cumulative array of frequencies. There are 72 submissions with at least 1 citation, 53 submissions with at least 2 citations etc.


From this data structure we can look up the H-index by traveling up the keys. First we cover the border cases. If the structure is empty there are no citations so the H-index is 0. If the current evaluation is of an H-index higher than the maximum citation number in the array (216) we can break from the evaluation. The H-index can never be higher than the maximum number of citations for a single submission.

Now we travel from the lower keys to the higher keys. As long a the value is higher than the key we can set the H-index to that key. After all, if there are 24 submissions with 5 or more citations, then the H-index is at least 5. In our example output we can travel up to an H-index of 11 in this way. There are 15 submissions with at least 11 citations. We have to keep checking though.

The next key value pair has a higher key than value. There are 14 submissions with at least 16 citations. This also means that there are 14 submissions with at least 14 citations. Based on this key-value pair we therefore have to higher the H-index to 14. (note: we tweaked the real value for this key – value pair in order to make this last point)

The next key value pair is 21 – 8. Because 21 is greater than 8 and we already have a higher H-index than 8 we do not update the H-index. Furthermore, because of the cumulative nature of the values in this array we know that the values will only go down as the keys go up. We therefore can stop evaluating and conclude that the H-index for author_id 1 in the year 2005 is 14.

For the technical implementation see Calculation processes on page 62.

5.4.3 Compute author-keyword frequency/inverse keyword frequency vector similarity

For this calculation we need the author keyword links that are implied by the submissions. We assume here that we have all author – keyword links for all given years. To compute the akf/idf of a given author and a given keyword for a given year we first get the number of author – keyword links between this author and this keyword that we derived from submissions from or before the given year. For author 1 and keyword 1 in or before the year 2012 this number turns out to be 73. We call this the author keyword frequency.

Second we need the number of links between this keyword and any author in or before this year.
For keyword 1 for the year 2012 in our database this number is 359. We call this the keyword link frequency.

As last variable we need the total number of author – keyword links in or before 2012. In our dataset this number is 42,224. We call this the total links. The ratio between the keyword link frequency and the total links is a measure for the rareness of the keyword. If an author has a keyword link to a rare keyword, this is more important than if the author has a keyword link with a very common keyword. We use this keyword link frequency and the total links to correct for the overall popularity of the keyword.

We calculate the inverse keyword frequency by log(total links / 1 + keyword link frequency). We use the addition of one in the divider to avoid dividing by 0. The smaller the keyword link frequency (devisor) is compared to the total links (divided) the larger the inverse keyword frequency will be and therefore the higher the author-keyword frequency/inverse keyword frequency. The author-keyword frequency/inverse keyword frequency is defined by the product of the author – keyword frequency and the inverse keyword frequency.

Once we have all akf/ikf for all authors and all keywords and all years we can generate akf/ikf vectors for different keywords for an author and compare these vectors using vector similarity. This is a general process and is described in Vector similarity on page 29.

5.5 Offline Processes
The offline processes consist of calculation processes that calculate new variables. We include these processes to indicate how we go from raw data to the information needed for the recommendations and other informative elements found in the recommender. We will only document these processes in a depth that shows how they come to their results. We will not document logging, debugging, type casts, encoding casts, the joining of tables to get related results, exact curl options etc. Only the primary processes are outlined here. The workflows of these processes can be found in Appendix B – Implementation workflows. The source code will be available separately.

5.5.1 Calculation processes
Calculation processes take live, harvested or derived data and calculate new metrics from that data.

5.5.1.1 batch get results
Input: working directory

Output: author-author graphs and keyword-keyword graphs in GML per year

This php script loops over two different graph types and all years from start year to current year to create .xml files of graphs constructed of authors or keywords connected by submissions up to a
5.5.1.2 calculate akf ikf
Input: author-keyword links

Output: author-keyword frequency (akf), inverse keyword frequency (ikf), akf * ikf.

This php script calculates the author-keyword frequency - inverse keyword frequency. This is the number of times an author is linked to a keyword through a submission corrected by the relative frequency that keyword is linked to any author compared to other keywords. To calculate this we use the cached author-keyword links generated by update_author_keyword_links. These links are based on submissions. Below the relations between the input and output data structures is depict.

The submission date is included to be able to see the relative connection strength between an author and a keyword grow over the years.

5.5.1.3 calculate author metrics from citations
Input: citations per author per year

Output: Google scholar citation metrics per author per year

This php script calculates the following metrics per author per year: citation_count_google_scholar, publication_count_google_scholar, single_most_citations_google_scholar, h_index_google_scholar. In order to calculate these metrics we use the relationships represented in the image below.

For a more elaborate explanations of the calculation of the H-index see the paragraph Compute H-index on page 59.

5.5.1.4 calculate results
Input: working directory with author-author graphs and keyword-keyword graphs in GML per year
**output:** network variables per graph and per node

This R script calculates the following global variables for all graphs in an working directory: "degree","betweenness","cluster_coefficient","degree_simplified","betweenness_simplified","cluster_coefficient_simplified","degree_keywords","betweenness_keywords","cluster_coefficient_keywords","degree_keywords_simplified","betweenness_keywords_simplified","cluster_coefficient_keywords_simplified". It also exports the betweenness and degree for each node in all graphs in the working directory.

**5.5.1.5 get results**

**Input:** none

**Output:** none

This bash script initiates the calculation of the betweenness and the degree for the authors and keywords per year. The first substantial number of submissions in our database are from 2005, so this is the year when we start calculating. The working directory is set to include the time to avoid overwriting previous runs.

**5.5.2 Update Scripts**

Update scripts perform two main tasks:

1. Harvest information from external resources which are needed for calculation of our recommendation variables.
2. Link information from local resources to make calculations more efficient and transparent.

**5.5.2.1 get mendeley author metrics**

**input:** author first name and author last name

**output:** hr_index_mendeley, gr_index_mendeley, single_most_read_mendeley, publication_count_mendeley, bookmark_count_mendeley
This script harvests author metrics from Mendeley. These results are based on the number of times an paper is read on Mendeley.

5.5.2.2 get submission citation data

input: submission title, author first name and author last name

output: number of citations and citations list link.

This php script looks up submissions on google scholar and parses the first page to look for the number of citations and the link to the first citations page. The link is marked Illustration 20 below.
We require for all words in the title in DSpace to be in the title on google scholar. We also require the last name of the author to be coupled to the paper on google scholar. Furthermore the paper should be dated between 1900 and the current year. Only results that meet these requirements are inserted into our database.

Out of 2926 submissions in our database we found citations for 888. The total number of citations found is 11068. That is on average 12.46 citations per paper, but these citations are not evenly distributed. 1472 submissions have multiple versions in which case the citations can not be parsed from google scholar. 347 submissions can not be found and 219 have 0 citations.

Discussion: the requirement that all words in the title of DSpace must be in the title on google scholar together with the requirement that the author must be coupled with the paper results in a number of false negatives. Some papers marked as -1 (not found) in our database when looked up manually without the use of advanced search on google scholar will give the correct paper.
However we give more priority to not counting false positives. Our submissions might not be complete, but the submissions that are in our database are correct. We prefer to work with less data that is clean than with more data that might be corrupted.

Google does not provide a API for google scholar. This created the need to scrape the pages. However, google does not seem to allow this either. It redirects requests to a captcha when to many requests originate from the same IP address in a small amount of time. We were able to reduce the run time of the scraping process to around an hour (instead of days) by spreading the requests over different servers. We set up ssh tunnels to these servers and bind these tunnels to a local port (ssh -d). We then use these servers as proxies to sent the requests. When a captcha is detected we switch to another server and resume our requests.

5.5.2.3 get citations per year
input: citations link
output: individual citations

This php script parses the individual citations found spread over different pages. It also retrieves the date of the citations which is vital for the calculation of H-indexes over years. Simply using the number of citations that we found in the process get_submission_citation_data does not enable us to see the growing number of citations a paper attract over time.

The citations (when they exceed the number of 10) are spread over different pages. We saved the link to the first page during the process get_submission_citation_data. The other pages we call by manipulating the start variable in the URL. See Illustration 21 below.
5.5.2.4 get submission metadata
input: DSpace unique identifier
output: submission title and submission date

In order to harvest the citations from google scholar we need the titles of the submissions. These were not included in the original database. We use the DSpace unique identifier which is included in the database to retrieve the title and the date of submissions from the original DSpace repository.

5.5.2.5 initiate usernames
input: author first and last name
output: author username

The recommender is not open to non university users. Also the recommendations depend on the keywords attached to the author requesting the recommendation. We therefore need to identify the person that requests the recommendation. This is done by means of a login. This php script initiates the usernames op the authors. It uses the first and the last name of the author. The passwords have been inserted manually (one universal password that users can change).

5.5.2.6 transport reader meter results
input: reader meter results from older database
output: reader meter results from newer database

This php script transports reader meter results from earlier versions of the COCOON CORE database to newer versions of the COCOON CORE database. The structure of the database had changed over the course of the project into the structure documented here. Because of the fact that
the reader meter service went offline by the beginning of 2012 we had to keep using the results we harvested in 2011. This created the need to transport the data from different versions of the database. Update: the service has seen very sporadic uptime. We we able to perform a complete harvest on 12-05-2012.

5.5.2.7 update author keyword links
input: submissions
output: author-keyword links

Illustration 22: Entity Relation excerpt author_keyword_links
This php script processes author-keyword links implied by submissions. The resulting new table is used to avoid joining five separate tables and is therefore a performance gain. It also simplifies the sql queries in these cases, making them less vulnerable to syntactic and semantic errors.

5.5.2.8 update metrics from r
input: working directory and author-author- and keyword-keyword graphs
output: none

This php script processes the betweenness and degree results calculated by R for authors and keywords over all years. This script makes use of the File_CSV_DataSource library (http://pear.php.net/package/File_CSV_DataSource).

5.5.2.9 copy latest betweenness to authors
input: none
output: none

Illustration 23: Entity Relation excerpt author_metrics-author
This php script copies the latest betweenness value for all authors to the authors table. This is done to be compatible with legacy code in the front end, but it also results in a performance gain for on the fly calculations because there is no need to join the author_metrics table when generating an author graph.

5.6 Online Processes
Online processes are processes that are executed on the fly. For each call to the process from the
front end the process is run. We do however use caching to limit the cpu cycles needed to deliver (large) graphs. We include these processes to show how we calculate information for the utility based functions of the site, most notably the recommendations with custom settings.

5.6.1 Interfaces
Interfaces enable the front end to extract information from the database. These processes contain execution logic.

5.6.1.1 authors
input: search term, cut-off year, max results
output: authors

This php script returns a list of authors in XML that adhere to the input parameters. This interface is called by the front end in order to retrieve authors from the database. An author will be included if:

- the search term is in its first- or last name
- the author has written a submission in or before the cut-off year
- the maximum number of results has not been reached

The maximum number of results is used in the general search function. This function has an auto update so the first few keystrokes can generate a lot of results.

5.6.1.2 keywords
input: search term, cut-off year, max results
output: keywords

This php script returns a list of keywords in XML that adhere to the input parameters. This interface is called by the front end in order to retrieve keywords from the database. A keyword will be included if:

- the search term is (part of) the keyword
- the keywords is linked to a submission in or before the cut off year
- the maximum number of results has not been reached

The maximum number of results is used in the general search function. This function has an auto update so the first few keystrokes can generate a lot of results.

5.6.1.3 get author graph
input: cut-off year, author_id, depth
output: author-author graph in GML
This php script returns an author-author graph in GML. Only authors with submissions in or before the cut-off year are included in the graph. If an author_id and depth is provided, this interface will return an author-author graph centred around the author with author_id and include all his or her connections in the depth degree. The output is cached in a file. If the same request to this interface is made in the future, this file will be served until the cache folder is emptied. We use two different algorithms to build the graphs. One in which we build graph around a central author and one in which we lookup all authors and links simultaneously. The first algorithm would be very expensive if we build a total graph for a particular cut-off year.

5.6.1.4 get keyword graph
In analogy to get_author_graph.php. Only the second algorithm is implemented

5.6.1.5 get author keyword graph
input: cut-off year

output: author-keyword graph in GML

This php script returns an author-keyword graph in GML. Only author keyword links based on submissions from or before the cut-off year are included.

5.6.1.6 calculate recommendation
input: cut-off year, request_author_id, co-operation centrality, keyword similarity, selected keywords.

output: list of recommended authors sorted by overall score

This php script returns a list of authors that are recommended for co-operation for the author requesting the recommendation. The recommendation is based on the betweenness of authors as well as the similarity of the keywords that the authors are connected to unified with the additional keywords selected in the recommendation settings by the requesting author. The co-operation centrality and keyword similarity settings determine how heavy these variables weigh in the recommendation. Together these two variables represent a value of 100%.

5.7 Front End
The front end consists of pages on which the user can find information and interact with the recommender. The front end is served data from the interfaces and REST full services described in Interfaces on page 70. We will describe all front end pages below.

5.7.1 Single Author Page
The single author page, depicted in Illustration 24: Single Author Page, contains all relevant information on an particular author.
The user can view the betweenness of the author in the network over time to get an impression of how influential the author is and since when this influence has developed. The H-index is also presented over time to give the insight into the quality of the scientific output of the author over time. Users can for instance check the growth of the H-index of an author against the benchmarks Hirsch proposed (H-index on page 8). The degree of the author is also plotted to give the user an overview of the number of authors that this author has co-authored with.

A graph of all authors that this author is connected to by paths of a maximal length of 3 is shown on the bottom of the page. This way the user can navigate the co-author network centred around the author currently under review. The user can click on other authors in the graph to navigate to the page of that author. The author that is the subject of the page is depicted as a node of a different colour than the rest.

There are multiple ways in which a user can navigate to the single authors page. In all author co-operation graphs the nodes that represent authors are clickable. Alternatively a user can navigate to a single author page by clicking the authors name in an author table on the Authors Page or on the Dashboard Page. Finally users can search for the name of an author in the general search function in the header.
Illustration 24: Single Author Page
5.7.2 Authors Page

The authors page, depicted in Illustration 25: Authors Page, contains an overview of all authors in the network. The data on the page can be cut off on a given year. Only authors that have published papers in our database in or before the cut-off year are included in the data on the page.

The authors page contains a table with all relevant authors. This table can be searched for a particular name. Also the number of authors shown per page can be changed. The H-index for the cut off year of the authors in the table are included in the last column.

Finally an author co-operation graph is included on the bottom of the page with all relevant authors of the current cut off year. Because of the small-world effect in the collaboration network we know that authors are typically connected to other authors that are interesting to them by a small path. Authors however do often not know this path. The graph is intended to make the network in which the authors work more transparent to them; helping them find the connection path to authors that they identified as interesting.

The authors page can be reached via the main navigation menu on the left of each page.
**Illustration 25: Authors Page**

The image shows a web page from COCOON CORE, which appears to be a tool for exploring authors. The page is divided into sections for authors and keywords. The authors section is filtered by a cut-off date set to 2012. Below this, there is a table listing authors' names and their H-index for the year 2012. The table includes columns for ID, Author, and H-index 2012, with entries for various authors including Rob Vogel, Bjo Oliver, Ros Ponnambalam, Maulke Hendrik, Hans Heuvel, Joostyn Handel, Bas Genders, Peter Van Rooijen, Peter Sloe, and Jan Van Bruggen. A graph is also displayed, possibly representing author co-authorship relationships.
5.7.3 Dashboard Page

The dashboard page, depicted in Illustration 26: Dashboard Page, is the main page of the recommender. This page is used to request recommendations for scientific co-operation. The page almost entirely consist of a form. The cut off year can be set to only include recommendations based on data from or before that year.

A list of all keywords is shown to the user. The top 5 keywords of the author that is currently logged in are set as target keywords for the recommendation. The user can use the list of keywords to add 1 to 6 extra target keywords to the recommendation if the user is interested in researching a specific topic. Keywords are added to the recommendation by placing them in the right list. The keywords can be searched to easily find specific keywords the user is looking for.

Two sliders are given to input the ratio of importance the user would like to give to the two factors in the recommendation: co-author influence, measured by betweenness, and co-author similar interest, measured by keywords similarity between the recommended co-authors and the user. The sliders must always add up to 100%. Therefore if the user manipulates one slider, the other is automatically adjusted.

If the user clicks on 'give recommendation', the recommended co-authors are calculated based on the current recommendation settings on the page. The result of the recommendation is a table that is inserted under the recommendation form which shows the recommended authors sorted by overall weighted score.

As a start off point for more of a browsing experience a co-author graph centred around the currently logged in user is presented on the bottom of the page.

The dashboard page is the default landing page of the recommender. Alternatively the page can be reached via the main menu on the left of every page.
5.7.4 Single Keyword Page

The single keyword page, depicted in Illustration 27: Single Keyword Page, contains all relevant information on a particular keyword. The betweenness and degree centrality of the keyword in the keyword network is presented over time. This way the growth or stagnation of popularity of a keyword can be seen over time. Authors might be interested in doing research in research topics that are starting to become popular.

The single keyword page can be reached via the keywords page or via the general search function in the header of every page. This search function autocompletes after every keystroke and presents all author names and keyword names that match the search query currently in the search box.
5.7.5 Keywords Page

The keywords page, depicted in Illustration 28: Keywords Page, lists all keywords in our data. The keywords can be cut off at a given year. Only keywords that are connected to papers from or before the cut-off year are included on this page. The table can be sorted on betweenness and on keyword name. It can also be searched for a keyword if the user wants to find a specific keyword.

On the bottom of the page the keyword network is included for all keywords in the current cut off year. This graph can be browsed; clicking on a keyword will redirect the user to that keywords single keyword page.

The keywords page can be reached via the main menu on the left of every page.
The login page, depicted in Illustration 29: Login Page, allows the user to log into the system. A login is necessary because we need to know for which user we are computing a recommendation and although the data is public the authors did not give explicit permission for it to be used in this way. If a user requests any page of the recommender while not logged in, he or she will be redirected to this page.
5.7.7 Update Password Header

The update password header, depicted in Illustration 30: Update Password Header, allows the user to change his or her password. It is accessible on every page by clicking on 'account view' in the header of every page.
5.8 Caching
We use caching to be able to serve graphs faster to clients once they have been created. Cache in our implementation is file based. Generated graphs are stored in files with filenames that represent hashed server requests. If the same request is received again by the server, the contents of the cached files are served and the graph is not calculated again. There is a small time penalty for writing the file for the first request, but caching the graphs saves time and CPU cycles for every similar request in the future.

5.9 Libraries and datatypes

5.9.1 Libraries
Here we list the external libraries we used in our implementation.

5.9.1.1 Cytoscape web
Cytoscape web is a javascript library designed to visualize graphs. We use this library to present graphs to the user with relevant (sub)graphs of the co-operation network. This helps to overcome the information overload.

5.9.1.2 simplehtmldom 1.5
This is an html parser library for php. It reads in an html string and parses the html into an object based on the hierarchy of the tags. The object can be searched for classes of elements and can be traveled down the hierarchy to find the values we are looking for. We used this library to parse the individual citations on google scholar.

5.9.1.3 File CSV DataSource-1.0.1
File CSV DataSource is a PHP object library that reads in csv files and makes the data accessible as a php object. We use this library to import the network variables calculated by R into our COCOON CORE database.

5.9.1.4 R
R is a statistical computing software environment and programming language. It is widely used for data analysis.

5.9.1.5 igraph package for R
Igraph is a package for R that includes functions to calculate network analysis metrics for graphs.

5.9.1.6 Datatables
Datatables is a plugin for jQuery, a javascript library. We use datatables to present the lists on the site. Datatables allows for sorting of the tables on the client site and searching within the table (http://datatables.net/).
5.9.1.7 Flot
Flot is a jQuery library that creates graphical plots of datasets on the client side. We use flot to create the plots of data over time on the author an keyword page.

5.9.2 Datatypes
Here we list the main datatypes we used to transfer information from one process to the other.

5.9.2.1 GraphML
GraphML is a file format used to represent graphs. It uses an XML based syntax. We chose to use graphML for its compatibility with other programs (R) and scripts (Cytoscape web) we use in our implementation.

5.9.2.2 JSON
JSON is a human readable text based data structure. It is capable of representing objects and is used to serialize data before transmitting it over a network. JSON can be easily parsed into a javascript object which makes it a good choice for transmission of data to the jQuery libraries we use.

5.9.2.3 CSV
Comma separated values (CSV) is a widely used data format. Rows of data are stored in lines and comma's separate the columns in each row of data. We use CSV to transport network analysis data from R to our COCOON CORE database.

5.10 Summary of design choices
Here we give a top level account for the design choices in the system. We choose to pre calculate everything we could pre calculate in order to speed up the recommendation process. Caching is introduced to speed up the delivery of graphs to the front end. The calculations en harvest operations are all scripted to facilitate easy updating in the future. We use REST server scripts to make it easy to extend the front end. The front end itself is heavy on javascript to facilitate a responsive user-interface based on content refreshes in stead of page refreshes.
6 Results

6.1 Experiment 1

6.1.1 Experiment 1.1: Average $\Delta H$ for recommended co-authorships

6.1.1.1 $\Delta H$-index and new submissions

First we ran an evaluation of the training set of all connections based on submissions from 2005 up to 2008 and the test set of all connections based on submissions from 2009 through 2012. We found a positive $\Delta H$-index between the training set and the test set for 166 of the 570 authors in the training set. Of those 166 authors 129 submitted new papers in the test set. Of the 404 authors with a neutral $\Delta H$-index 119 submitted new papers in the test set. See Table 14 below.

<table>
<thead>
<tr>
<th></th>
<th>Positive $\Delta H$ in test set</th>
<th>Neutral $\Delta H$ in test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>New submissions</td>
<td>129</td>
<td>119</td>
</tr>
<tr>
<td>No new submissions</td>
<td>37</td>
<td>285</td>
</tr>
<tr>
<td>Total</td>
<td>166</td>
<td>404</td>
</tr>
</tbody>
</table>

*Table 14: Delta H signs in test set*

First we must address the 37 authors that have a positive $\Delta H$ value which we can not ascribe to new submissions. The increase in H-index must come from citation of papers that are not in our test set. There are two possible explanations. The increase in H-index could have been explained as to result from co-operations outside of our database; the author has moved to a different university. Alternatively the increase in H-index for these authors results from older papers in the training set. Over time valuable older papers keep getting cited, so older papers can have as much of an impact on the H-index as new papers.

It is good however that 77.7% of the authors that saw an increase in H-index between the training- and test set have contributed new submissions in the test set. These authors engaged in valuable co-operations we would like COCOON CORE to recommend based on the training set.

In the column for the number of authors with a neutral $\Delta H$ between the training set and test set in Table 14 above we see that 285 (75.5%) of authors did not add new submissions. This is to be expected. No new submissions result in no new citations which results in no increase in H-index (not considering new citations to older papers). The other 119 authors with a neutral $\Delta H$-index did contribute new submissions. For the purpose of this experiment we consider these submissions (and therefore co-operations) as unsuccessful.

6.1.1.2 Evaluating $\Delta H$-index and recommendation score.

In order to evaluate COCOON CORE recommendations we will first assume that any increase in H-index can be ascribed to the new submissions (and new co-authorships) made in the test set. In
order to calculate the value of our recommendations we take the average recommendation score for all co-authors in the test set of authors in the training set that had a positive Δ H between the training- and the test set. We compare this average to the average recommendation score for all co-authors in the test set of authors that had a neural Δ H between the training- and the test set.

Given that an author has a positive Δ H-index; if COCOON CORE gives higher recommendation scores to co-authors based on the training set that actually become co-authors in the set-set, then we can conclude that COCOON CORE is able to advice or predict successful co-operations. This results in the hypothesis that the average recommendation score of co-authors in the test set based on the training set will be higher for authors that have a positive Δ H-index as opposed to authors with a neutral Δ H-index. We can formulate this in two hypothesis:

1–H0: The value of research co-operation recommended by the system is equal to the research co-operation recommended not by the system.

1–H1: The value of research co-operation with recommendation is greater than the value of research co-operation without co-operation.

We used the following algorithm to set these two subgroups of recommendations up against each other:

For all authors with a submission in the test set we:

1. get the H-index of the author at the end of the training set
2. get the H-index of the author at the end of the test set
3. calculate the Δ H-index for this author
4. get recommendations for this author based on the training set with influence slider set to various values (0%-100%)
5. get all co-authors in the test set
6. For each recommendation based on different influence settings:
   1. get recommendation score based on the training set for co-authors in test set
   2. calculate the total recommendation score based on the training set for all co-authors in test set
   3. calculate the number of co-authors in test set
   4. calculate the average recommendation score for all co-authors in test set
   5. add total recommendation score, the number of co-authors and average
recommendation score for all co-authors to the group of positive Δ H-index or negative Δ H-index

7. calculate all averages for both groups

6.1.1.2.1 Results for experiment 1.1
The results can be found in Table 16 below. Table 15 also shows the average total cumulative score and the average number of connection of an author in the test set for the recommendations based on influence setting 50%.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Average recommendation score of connections in test set based on training set</td>
<td>19.44</td>
<td>0</td>
<td>22.07</td>
<td>0</td>
</tr>
<tr>
<td>Average total cumulative score in test set</td>
<td>575.18</td>
<td>0</td>
<td>325.93</td>
<td>0</td>
</tr>
<tr>
<td>Average number of connections in test set</td>
<td>33.99</td>
<td>0</td>
<td>17.87</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 15: Score metrics for recommendations based on training set with betweenness (influence) setting 50%

<table>
<thead>
<tr>
<th>Betweenness influence</th>
<th>Average recommendation score for Positive Delta H with new submissions (129)</th>
<th>Average recommendation score for Neutral Delta H with new submissions (119)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>25.06554689227</td>
<td>28.239680644922</td>
</tr>
<tr>
<td>10</td>
<td>23.940702956252</td>
<td>27.00556337377</td>
</tr>
<tr>
<td>20</td>
<td>22.815859020237</td>
<td>25.771446102617</td>
</tr>
<tr>
<td>30</td>
<td>21.691015084222</td>
<td>24.537328831463</td>
</tr>
<tr>
<td>40</td>
<td>20.566171148205</td>
<td>23.303211560313</td>
</tr>
<tr>
<td>50</td>
<td>19.441327212189</td>
<td>22.06909428916</td>
</tr>
<tr>
<td>60</td>
<td>18.316483276174</td>
<td>20.834977018008</td>
</tr>
<tr>
<td>70</td>
<td>17.191639340156</td>
<td>19.600859746853</td>
</tr>
<tr>
<td>80</td>
<td>16.066795404141</td>
<td>18.366742475701</td>
</tr>
<tr>
<td>90</td>
<td>14.941951468125</td>
<td>17.132625204549</td>
</tr>
<tr>
<td>100</td>
<td>13.817107532109</td>
<td>15.898507933398</td>
</tr>
</tbody>
</table>

Table 16: Average recommendation scores for different betweenness (influence) settings
Looking at Table 15 we see that the average number of connections in the test set for authors without new submissions is 0; both for authors with and without a positive Δ H-index. This is a validation for our calculations because authors without submissions in a given subset of the data naturally also do not have any co-authors (connections) in this subset.

We would expect to see a higher average recommendation score for connections that resulted in a positive Δ H-index in the test set. This is however not what we see in Table 15. The average recommendation score for co-authors of authors that have a neutral Δ H-index is higher than that of authors with a positive Δ H-index. In Table 16 above we can see that this is systematically the case for all betweenness (influence) settings.

Furthermore we see the average score for recommendations go down for both groups the more emphasis is given to the betweenness (influence) in the recommendations. See Table 16. This is to be expected for this experiment in which we only consider recommendations for co-authors in the training set that actually became co-authors in the test set. If you become co-author the changes are greater that you already were co-author and/or at least have more similar keyword vectors than two random authors. The more weight this similarity gets in the recommendation, the higher the recommendation score for the subset we are examining here will be. A lower betweenness weight results in a higher keyword similarity weight because of the inversely proportional relationship between the two in the recommendation. See 5.7.3 Dashboard Page on page 76. That is why, in this experiment, the higher the betweenness (influence setting), the lower the average recommendation score.

6.1.1.3 Conclusion experiment 1.1
We must conclude that the recommender does not successfully separate the valuable co-operations from the less valuable co-operations in our data set.

6.1.2 Experiment 1.2: Evaluating citations and recommendation score
In the previous experiment we assumed that a positive delta H-index, was the result of the co-operations an author engaged in. Using the H-index however does not allow us to trace back the increase of influence of the authors scientific output to individual papers. In this experiment we make the citations per paper the target variable in stead of the H-index of the author. The citations can be more specifically ascribed to a paper, opposed to the overall H-index which is calculated over all scientific output of an author.

In this experiment we create pairs of a recommendation score and a citation number. For all cited co-operations for an author in the test set we look up the recommendation score for that co-operation in the recommendation based on the training set. If the co-author is not in the
recommendation we give this co-author a score of 0. If the author submitted more papers with the same co-author in the test set, we add up all citations of all papers they submitted together. Now we calculate the correlation between the recommendation score and the number of citations for that co-authorship.

### 6.1.2.1 Results for experiment 1.2

Table 17 below shows these correlation coefficients for runs with different betweenness settings (network influence).

<table>
<thead>
<tr>
<th>Betweenness influence setting</th>
<th>Correlations between co-author recommendation score and number of shared citations in test set</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.1177649</td>
<td>n= 6512, p=0</td>
</tr>
<tr>
<td>10</td>
<td>0.1308471</td>
<td>n= 6512, p=0</td>
</tr>
<tr>
<td>20</td>
<td>0.1447474</td>
<td>n= 6512, p=0</td>
</tr>
<tr>
<td>30</td>
<td>0.1591216</td>
<td>n= 6512, p=0</td>
</tr>
<tr>
<td>40</td>
<td>0.1734138</td>
<td>n= 6512, p=0</td>
</tr>
<tr>
<td>50</td>
<td>0.1868582</td>
<td>n= 6512, p=0</td>
</tr>
<tr>
<td>60</td>
<td>0.1985595</td>
<td>n= 6512, p=0</td>
</tr>
<tr>
<td>70</td>
<td>0.2076662</td>
<td>n= 6512, p=0</td>
</tr>
<tr>
<td>80</td>
<td>0.2135881</td>
<td>n= 6512, p=0</td>
</tr>
<tr>
<td>90</td>
<td>0.216157</td>
<td>n= 6512, p=0</td>
</tr>
<tr>
<td>100</td>
<td>0.2156414</td>
<td>n= 6512, p=0</td>
</tr>
</tbody>
</table>

*Table 17: Correlation coefficient of recommendation score and citations for different betweenness (influence) settings*

Correlation coefficients are symmetric so the order of the two variables is of no influence on the coefficient. Also the correlation coefficient is invariant to scale so we do not have to bring the recommendation scores and the citations to the same scale. N is the number of data pairs that were used to calculate the correlation coefficient. P is the chance that we would have seen this coefficient while the real coefficient is 0. We see that for this size of n and these coefficients that chance is 0. The confidence in these coefficients is therefore very high. The correlation coefficients all fall between 0.1 and 0.3 and therefore show a small positive correlation between the recommendation score and the number of citations.

### 6.1.2.2 Conclusion experiment 1.2

Looking at Table 17 above we observe that the higher the betweenness (influence) setting the stronger the correlation between the recommendation score and the the number of shared citations. This would suggest that the network influence of an author is a greater predictor for the number of citations a paper is going to get than the keyword similarity. Furthermore we conclude that there is a small but significant positive relationship between the recommendation score of an co-operation
and the successfulness of a co-operation when the betweenness is given almost all the recommendation weight.

### 6.1.3 Experiment 1.3: Correlation between betweenness and H-index

In this experiment we calculate the correlation coefficient between the betweenness of an author an his or her H-index. We use the H-index based on google scholar citations and the Hr-index based on Mendeley reads. If the correlation between the betweenness and the H-index is high, then the network influence of an author has a high predictive value for the H-index. We can formulate this in two hypothesis:

1. **H0**: The centrality of the author in the co-operation network is of no influence on the chance that this author is part of a successful co-operation.

1. **H1**: The centrality of the author in the co-operation network has a positive influence on the chance that this author is part of a successful co-operation.

This predictive value can not be mistaken for a causal relationship. It only stated that to some degree the two variables move together. This can give a hint for the optimal betweenness setting for COCOON CORE.

### 6.1.3.1 Results for experiment 1.3

The results can be found in Table 18 below.

<table>
<thead>
<tr>
<th></th>
<th>Betweenness 2012</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>H-index google scholar 2012</td>
<td>0.6781081</td>
<td>N=1361, p=0</td>
</tr>
<tr>
<td>H-index google scholar 2012 – non zero results</td>
<td>0.691771</td>
<td>N=611, p=0</td>
</tr>
<tr>
<td>Hr-index mendeley 2012</td>
<td>0.2193422</td>
<td>N=1361, p=0</td>
</tr>
<tr>
<td>Hr-index mendeley 2012 – non zero results</td>
<td>0.2286955</td>
<td>N=1249, p=0</td>
</tr>
</tbody>
</table>

**Table 18: Correlation between H-index (google scholar), Hr-index (mendeley) and betweenness**

We calculate the H-index with non zero results because not all citations can be found on google scholar. An H-index of 0 based on google scholar does therefore not necessarily mean that the real H-index is 0. We use non zero results for Hr-index, because not every researcher might publish on Mendeley. The Hr-index might therefore not be a good representation of the influence of the research of the author. As we can see in Table 18 the correlation coefficient does not significantly differ between the calculation with or without the zero values. Furthermore N is the number of pairs in the correlation calculation. P is the probability of obtaining a correlation coefficient at least as extreme as the one that was observed, assuming that the actual correlation is 0.
6.1.4 conclusion experiment 1.3
The betweenness and the H-index have a strong positive correlation. This means that the betweenness (network influence) of an author is a strong predictor for his or her H-index. Furthermore we see that the correlation between the betweenness and the Mendeley Hr-index is much weaker. This might indicate that authors with a high betweenness are less present on Mendeley or that interest on Mendeley is more directed at less influential scientists. Obviously there is no way to check either of these hypothesis with our current data.

6.1.5 Experiment 1.4: Correlation between clustering coefficient and H-index
When checking the correlation coefficient between the cluster coefficient and the H-index we are indirectly checking the correlation between the keyword similarity and the H-index. We can formulate two hypothesis about the relation between the keyword similarity and the H-index:

1.2–H0: Similar interest of two authors has is of no influence on the chance that these authors form a successful co-operation.

1.2–H1: Similar interest of two authors has a positive influence on the chance that these authors form a successful co-operation.

6.1.5.1 Results for experiment 1.4
The results are presented in Table 19 below.

<table>
<thead>
<tr>
<th>H-index</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster coefficient (pair-wise deletion)</td>
<td>-0.1561323, N=1361, m=607, p=0.0001</td>
</tr>
<tr>
<td>Cluster coefficient (all observations)</td>
<td>0.0603143, N=m=1361, p=0.0261</td>
</tr>
</tbody>
</table>

Table 19: Correlation coefficient between cluster coefficient and the H-index
Transitivity in igraph reports NaN for nodes with degree 0 or 1. This is because you need at least two edges to have a chance to form a cluster. In the first correlation calculation we delete the betweenness – cluster coefficient pair when the cluster coefficient is NaN. In the second calculation we use all observations by using cluster coefficient 0 for nodes with degree 0 or 1. N is the number of pairs, m the number of pairs after deletion and p the chance of observing this coefficient when the real coefficient is 0.

6.1.5.2 conclusion experiment 1.4
We see a very weak negative correlation when we delete the value pairs of authors with degree smaller 2. The correlation if we use the authors with a degree smaller than 2 is negligible. In our discussion of Research question 1.2 on page 49 we proposed that a high positive correlation in this experiment would indicate that the keyword similarity is an important predictor for the H-index. Furthermore we said that this would indicate a force in the network opposite to the betweenness. A
low negative coefficient in this experiment would therefore imply that the force of the betweenness in the network is much stronger. We've also seen this in Experiment 1.3: Correlation between betweenness and H-index on page 89.

### 6.1.5.3 Limitation experiment 1.4
For the last conclusion we do however assume that an author with a high betweenness usually has a low cluster coefficient. Also we assume that a high cluster coefficient leads to a high keyword similarity with ones neighbours. This last assumption we will not check because it follows from the way the network is set up. See Research question 1.2 on page 49. The first assumption we will check below.

### 6.1.6 Experiment 1.5: Correlation between clustering coefficient and betweenness
In this experiment we will check the assumption that a high betweenness goes together with a low cluster coefficient.

#### 6.1.6.1 Results for experiment 1.5
The results can be found in Table 20 below.

<table>
<thead>
<tr>
<th></th>
<th>Betweenness 2012</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster coefficient (pair-wise deletion)</td>
<td>-0.2175372</td>
<td>N=1361, m=607, p=0</td>
</tr>
<tr>
<td>Cluster coefficient (all observations)</td>
<td>-0.03110301</td>
<td>N=m=1361, p=0.2515</td>
</tr>
</tbody>
</table>

*Table 20: Correlation coefficient between betweenness and cluster coefficient*

#### 6.1.6.2 conclusion experiment 1.5
We see that there is a low negative correlation between the two. This means that to some extend the assumption that a high betweenness and a low cluster coefficient go together is true. But even if the two are not opposing phenomena's in the network, we can still conclude that in our dataset the betweenness is a much better predictor of the H-index than the cluster coefficient and therefore the keyword similarity.

### 6.2 Experiment 2

#### 6.2.1 Experiment 2.1: Perceived value of the recommendations results

##### 6.2.1.1 Sequence effect
First we check against the sequence effect by performing a t-test on the two different participant groups for all questions.

##### 6.2.1.1 Results experiment 2.1.1
The results can be found in Table 21 below. The individual valuations per question can be found in
Appendix C – Recommendation valuation questionnaire.

<table>
<thead>
<tr>
<th>question</th>
<th>t</th>
<th>df</th>
<th>Sig.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a</td>
<td>0,000</td>
<td>22</td>
<td>0,737</td>
</tr>
<tr>
<td>1b</td>
<td>-0,92</td>
<td>22</td>
<td>0,371</td>
</tr>
<tr>
<td>1c</td>
<td>-1,999</td>
<td>22</td>
<td>0,653</td>
</tr>
<tr>
<td>2a</td>
<td>3,924</td>
<td>22</td>
<td>0,177</td>
</tr>
<tr>
<td>2b</td>
<td>-0,705</td>
<td>22</td>
<td>0,707</td>
</tr>
<tr>
<td>2c</td>
<td>0,240</td>
<td>22</td>
<td>0,736</td>
</tr>
<tr>
<td>N = 24</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 21: Results of independent samples t-test. We assume a normal distribution and equality of variances.

6.2.1.2 Hypothesis research question 2

We formulate the following hypothesis for the research questions 2.1 – 2.4.

2.1–H0 The scientists give a **low or neutral** valuation to the recommendations when they can adjust the weights freely

2.1–H1 The scientists give a **high** valuation

2.2–H0 The scientists give a **low or neutral** valuation to the recommendations when the recommendation is entirely based on network influence

2.2–H1 The scientists give a **high** valuation

2.3–H0 The scientists give a **low or neutral** valuation to the recommendations when the recommendation is entirely based on interest similarity

2.3–H0 The scientists give a **high** valuation

2.4–H0 The scientists **do not** find the system easy to use

2.4–H1 The scientists **do find** the system easy to use

6.2.1.3 Results Median valuations per question

Illustration 31 below shows the median valuation per question. We see that the users are rather positive (4/5) about the recommendations when they set the betweenness setting to 100 (1a) and neutral (3/5) about the recommendations when they set the keyword similarity to 100 (1b). This is in line with the results from the correlation coefficients in Experiment 1.3: Correlation between betweenness and H-index and Experiment 1.4: Correlation between clustering coefficient and H-index. When users are allowed to tweak the influence settings on their own (1c) the valuation is moderately positive (3,5). The fact that participants value the recommendations lower when they are free to choose the influence settings than when they are instructed to put the betweenness setting
to 100% indicates that this last setting might be the most favourite setting. This is also in line with the correlation findings in Experiment 1.3: Correlation between betweenness and H-index and Experiment 1.2: Evaluating citations and recommendation score.

Looking at the valuations of the recommendations in the default user login session (2a-c) we see values comparable to the individual session (1a-c). Only the valuation of the recommendations when participants are allowed to set the influence freely is neutral as opposed to moderately positive in the individual session.

Illustration 32: Proportion of valuation values for each recommendation question Shows the distribution of valuations for each recommendation question. Here we notice that the valuations for 2a (63% > 3) are better than for 1a (50% > 3).
Illustration 31: Median valuation of each recommendation question

Illustration 32: Proportion of valuation values for each recommendation question
6.2.1.3.1 Conclusions experiment 2.1

In Table 21 we see that the significance (Sig.) for all questions falls well between the 5% significance edges. Therefore the valuations of both groups on all questions do not significantly differ. Thus we can conclude that there is no sequence bias in our experiment.

Furthermore the participants are rather positive about the recommendations when they set the author network power setting to 100 and neutral about the recommendations when they set the interest similarity to 100.

6.2.2 Experiment 2.2: SUS Usability questionnaire results

6.2.2.1 Results experiment 2.2

Table 22: Summary of System Usability Scale (SUS) results below shows the statistical summary for the normalized overall score of COCOON CORE on the SUS test. We see a median of 67.5 which is very close to the average score of any system (68). Compared to the usability the participants are used to COCOON CORE has in their opinion an average usability. Furthermore we see that based on the sample size of 24 participants we have a 95% confidence interval that a score is between 57.25 and 72.42.

<table>
<thead>
<tr>
<th>Measure</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min (minimal / lowest value)</td>
<td>25</td>
</tr>
<tr>
<td>M (mean value - sum of all values divided by the number of values)</td>
<td>65.27</td>
</tr>
<tr>
<td>GM (Geometric mean – nth square root of product of all n values)</td>
<td>65.25</td>
</tr>
<tr>
<td>Mdn (median value – middle of all values when ordered by size)</td>
<td>67.5</td>
</tr>
<tr>
<td>Max (maximum value / highest value)</td>
<td>90</td>
</tr>
<tr>
<td>95% confidence interval (95% confident that a random score is in this interval)</td>
<td>57.57 – 72.42</td>
</tr>
<tr>
<td>N = 24</td>
<td></td>
</tr>
</tbody>
</table>

Table 22: Summary of System Usability Scale (SUS) results

In Illustration 33 we can see the median scores for all questions in the SUS test. As opposed to the previous user questionnaire test the scale for the SUS test is 0-4 after adjusting the scale to calculate an overall score (see Experiment 2.2 - Usability test on page 52). We see that the participants are moderately positive towards the usability of COCOON CORE, giving a 3 out of 4 on most of the questions. The frequency distribution of the scores per question can be found in Illustration 34. We see that participant are especially positive about question 4 (96% > 2) and 10 (70% > 2). These questions reflect learnability.
6.2.2.2 conclusion experiment 2.2
The average overall score on the SUS test implies that the usability could not have significantly influenced the scores we discussed in Experiment 2.1: Perceived value of the recommendations.
results.

The fact that participants are especially positive about question 4 and 10 suggests that the information provided by the system is quite accessible to the user. This was one of our goals, to offer more network and influence transparency to the users. This conclusion is backed up by the fact that users feel that there are few inconsistencies in the system (question 6) and that it is not unnecessarily complex (question 2).

### 6.3 Summary of all results

With respect to the first research question we found that we could not convincingly prove the hypothesis that the value of the co-operations recommended by COCOON CORE are more valuable that co-operations not recommended by the system $1-H_1$. We therefore have to accept $1.1-H_0$ at this point, although we saw a weak but significant correlation between the recommendation score and the number of citations for a co-operation when the recommender only considered network influence (Experiment 1.2: Evaluating citations and recommendation score).

When looking at the individual recommendation variables we found that we could accept $1.1-H_1$. The betweenness has a strong predictive power for the H-index (Experiment 1.3: Correlation between betweenness and H-index). We were not able to disprove $1.2-H_0$: the keyword similarity did not have a predictive power on the H-index in our data (Experiment 1.4: Correlation between clustering coefficient and H-index). We must note that we measured this via the correlation coefficient so this lowers the confidence of the experiment.

Looking at research question 2 we can say that on average the participant we moderately positive about the results. Per research question we can conclude that we accept $2.1-H_1$: the participants give a high valuation for recommendations purely based on network influence. We can not reject $2.2-H_0$: the participants were neutral about the recommendations when the recommendations were purely based on similarity of interest. We accept $2.3-H_1$: the participants are moderately positive of recommendations when they can set their own weight distribution over the two recommendation factors. We can also conclude that the usability of the system does not have a significant negative effect on the valuations of the recommendation. The overall SUS usability score is average – 67.5 (Experiment 2.2: SUS Usability questionnaire results). With some reservation we also accept $2.4-H_1$: The scientist do find the system easy to use because it scored average on the usability test with high scores on learnability, consistency and complexity.

In general we see that although there is no strong evidence that the recommendations lead to more influential scientific output, the network influence of an author has a strong correlation to the H-index of an author. We will also see that participants of the user evaluation experiments are
moderately positive about the recommendations and that in many respects our university network is comparable to that of other universities.

For further comments and discussion of the results we refer back to chapters Results and Discussion.
7 Discussion

7.1 Introduction to discussion
In this chapter we will discuss the limitations of the methodology and the interpretation of the results for all experiments in greater depth. We will reference forward to future research where we think that improvements can be made to this research. There were results are discussed we will point out some reflexions which more freely supplement the conclusions in the next chapter which can definitely be made. We also included an external validity study based on the network variables of our network compared to those of other university networks and generated networks.

7.2 Experiment 1

7.2.1 Methodology
7.2.1.1 Training set / test set
Illustration 35 Shows the number of new submissions per year in our database. The number of added submissions is higher in the later years. The last submissions is from the 24\textsuperscript{th} of April 2012, so the number for 2012 will still go up.

Our method of splitting the data might be a bit simple. We cut the period over which there were submissions in the database in half (2005-2008 and 2009-2012). If we take a look at the number of submissions is each set for this division we see that 1064 submissions are in the training set and 1862 submissions are in the test set. A difference of 798 submissions. We could have made this difference smaller by selecting a training set of all submissions from 2005-2009 (1662) and a test set of all submissions from 2010-2012 (1264). This results in a difference of 398. Obviously the date that splits the two sets straight in the middle is somewhere between these two options.

Given that we have a small number of positive and negative examples in our test set we might also

Illustration 35: number of submissions in DSpace per year
have chosen to split the data in such a way that we optimize the number of authors that have both submissions in the training set and in the test set.

Given that we were not that far of from the middle of the dataset and the fact that researchers come and go in partly overlapping time periods we think that the division between the training set and the test set might not be optimal, but at the same time not that far off. We therefore do not think that an optimal split will render very different results.

7.2.1.2 Remove authors that have not published in x year
They might have left the research field or academic society. Recommending these authors in that case is pointless. Also when looking at the present data authors that have no publications in the test set might water down the value of the recommendations. We can not know, based on the training data alone that these recommendations are pointless, but removing them with knowledge of the test data violates the strict separation of the two datasets. The system will also not have this advantage looking into the real future (after 2012). We therefore opted not to 'clean' the training set from authors that do not reappear in the test set.

7.2.1.3 External authors
We only consider co-operations within the university. External authors might be included, but their submissions might not reflect all their research. A low centrality for external authors in such a case does not reflect their co-operation position in the scientific field. Also, the betweenness of authors in our network that are connected to external authors might be much higher if they form a bridge between our network and that of other universities. Including data from other universities however is out of the scope of this thesis, so we have to accept that some authors betweenness might be underestimated in our data. In future research the data of multiple universities might be merged to identify the role of these bridges. See future research chapter 9.2 Combine data from multiple universities on page 115.

7.2.1.4 Small percentage can actually be evaluated
The number of recommendations we can evaluate against the real world is quite small (129 positive and 119 negative examples). This sparsity, usually a problem in collaborative filtering recommenders, introduces uncertainty into the evaluation results. Better recommendations which we can not check in the data do not help the performance of the recommendation on the test data. It is tempting to speculate that the performance of the recommender is better than its results on the test set reflect, but this naturally can only be proved or disproved with an intervention test wherein scientist are obliged to follow the recommendations of the system.

7.2.1.5 H-index weak dependant variable for quantitative analysis
Having a limited number of positive or negative examples in our test set increases our need for a discriminating target variable. As a natural number and with a log tail distribution the
discriminatory power of the H-index is quite weak. Table 23 Shows a statistical summary of the H-indexes in 2012. We see that although the value range is from 0 to 22, the fist quarter is filled with only 0's and 1's and that the median is only 4.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Min</td>
<td>0</td>
</tr>
<tr>
<td>1st quarter</td>
<td>1</td>
</tr>
<tr>
<td>Median</td>
<td>4</td>
</tr>
<tr>
<td>Mean</td>
<td>7</td>
</tr>
<tr>
<td>3rd quarter</td>
<td>13</td>
</tr>
<tr>
<td>Max</td>
<td>22</td>
</tr>
</tbody>
</table>

*Table 23: Statistical Summary of the H-indexes for all authors in 2012*

The H-index is designed to make an increase in H-index exponentially harder the higher the H-index becomes. Although this gives a very robust metric for research output influence, it reduces the H-index discriminating power between the authors. Moreover the changes in the H-index between the training set and the test set are difficult to ascribe to specific papers. The H-index however still serves as useful metric for users to quickly see the influence of an author based on our dataset of scientific papers. Also the H-index could be a good recommendations variable. See future research chapter 9.1 H-Index as recommendation variable on page 115.

**7.2.1.6 Correlations are no causal relations**

We use correlations to look for connections between metrics. Although a significant correlation is a sign that the two metrics influence each other it is incorrect to say that there is a causal relationship between the two. Although we might speculate about how a high betweenness results in high H-index we have not proven that with our correlation calculations. A future research could use multiple linear regression to examine these hypothesis.

**7.2.2 Results**

**7.2.2.1 COCOON CORE performance**

There is the possibility that COCOON CORE recommends better co-operations than those that have been formed in reality. These better co-operations can not be checked in the current network data because they have not occurred, but they do result in lesser performance on the current data because they are preferred by the system above existing, less successful, co-operations. This naturally can only be tested with an intervention study wherein the system is used for a couple of years an when researchers form other co-operations than they would have without the recommender.

It is also possible that the number of positive and negative test pairs in our data set is to low to find a positive selection if there should be one. Also the weighted technique might not be best suited to
find good recommendations. More on both these points below.

For now however we must conclude that there is no strong evidence in this data that the recommender is able to predict the valuable co-operations based on the test set of papers.

### 7.2.2.1.1 Citations better target variable than H-index

For an explanation as to why the system does not perform well on the given dataset using the H-index as the target variable we look at the citation distribution of the submissions. We must conclude that insignificant co-operations are the norm and influential co-operations are scarce. Of the 2853 submissions only 1034 (36,2%) were found on google scholar. Of these submissions only 888 (31,1% of all submissions) had more that 0 citations. Only 264 submissions have more than 10 citations. For the test set this last number drops to 21. We can see that the distribution of the citations found on google scholar is a long tail distribution. See Illustration 30 below. The authors with a positive Δ H-index on average form nearly twice as many new co-operations. See Table 15 on page 86. This could lower their average recommendation score because they suffer more from the bad co-operations which were not recommended, but are calculated in the average recommendation score.

![Illustration 36: Google Scholar citation distribution](image)

Also the new co-operations in the test set can occur with new authors that were not present in the training set and were therefore not available as recommended authors. For successful authors in the training set this actually can be a big problem because they attract a lot of co-operations with new authors due to preferential attachment. See Generating scale-free networks on page 16.
7.2.2.1.2 Weighted hybrid technique
Our weighted hybrid technique assumes that we can use universal weights for all authors. This might not be the case. It is possible that different authors benefit more from different recommendation techniques. In our experiments we used the average scores for all authors on automated runs. Authors themselves can of course change the sliders as they like. An author with a low centrality in the network might choose to use more betweenness in his or her recommendation to find an influential researcher. An influential researcher might choose to use more keyword similarity because he or she is looking for a co-author with specific knowledge. Individual users might be more capable of choosing their own weight setting and might therefore get better recommendations than the automated runs can get due to their universal weight settings over all authors. See future research chapter 9.5 Tweak hybrid system on page 116.

7.2.2.2 Influence of different recommender techniques
Experiment 1.1 is biased towards keyword-similarity. Therefore we should not look at the results of this experiment to determine the ideal betweenness (and keyword similarity) settings. The second experiment seems to suggest that a higher betweenness setting results in a better correlation between the recommendation score and the number of citations of the co-operation. This trend seems to peak at setting 90 for betweenness, but the value for setting 100 is not significantly lower, so we must be hesitant to call 90 an optimum. Naturally we can not have a setting higher than 100 so the results of experiment 2 seem to suggest that the betweenness of an author is much more important than the keyword similarity. Experiment 1.3: Correlation between betweenness and H-index (cor coef: 0.69 p=0) and Experiment 1.4: Correlation between clustering coefficient and H-index (cor coef: 0.06 p=0) seem to underscore that conclusion. Given that the recommender has not actually performed very well on this dataset any conclusions should be made with caution and could better be based on the correlations of experiment 1.3 and 1.4.

We used the cluster coefficient to look for the keyword similarity between each author and his or her co-authors. While in theory the average keyword similarity of a user with its co-authors must increase when the cluster coefficient increases, the strength of this effect has not been measured by us, so we can not definitively say that there is no correlation between the keyword similarity and the H-index.

7.3 Experiment 2

7.3.1 Methodology
7.3.1.1 Possible lack of insight users
A disadvantage of a user evaluation study is that users do not always know what is best for them. When you recommend movies based on interest such an evaluation may work, because users can be
satisfied or dissatisfied with the recommendation. However, for a scientist co-operation does not merely consist of interest, the recommendation should focus on the success rate, and scientists are not always aware of the factors that influence the success of a paper.

7.3.1.2 SUS Usability questionnaire subjective
The answers to the SUS test questions are subjective. This in general makes it harder to draw confident conclusions based on these answers. There are however a few factors that raise the confidence in the results of the test. The questions of the SUS test are selected for their discriminating power. When setting up the SUS test participants were very divided on these 10 questions and less divided on the other 40 candidate questions that did not make the final test.

Furthermore the participants are professional users that are familiar with the problem domain. They also themselves form the target group so that makes their answers more significant.

7.3.2 Results
7.3.2.1 Overall trend
Overall the valuation of COCOON CORE is moderately positive. Participants are especially interested in the powerful user aspect of the recommender, but the similar author setting also scores 3 out of 5. We might not be able to convincingly link the recommendations to successful scientific output in experiment 1, but the recommender might nonetheless make users more aware of their neighbours and their role in the entire network they operate in. The system might raise this awareness more for new members of the community than for older members.

A PhD candidate:

“Why should I want to put the slider at another place than 100 which represents the maximum relevancy? I can see any advantage to a search at 50 though I can see that it delivers different names (less prominent possible co-authors matching better my humble situation of a Phd and that I can more easily approach?)”

An assistant professor:

"When looking for co-authors with similar interest you know most of the recommended people However the graph is interesting to explore to find new (young/newsters) !”

We also see that the PhD candidate reasons about the best setting for his / her particular situation. This can make the recommender perform better than on the universal weight settings we used in experiment 1.

With on average a moderately positive score for the recommendations it would be interesting to see the results of recommended co-authorships when they actually occur in the future. Only then will we be able to research if the recommender actually helps in raising the quality of the scientific
output of users and of the community

7.3.2.2 Favourite setting
Participants value the betweenness setting of 100% higher (4/5) that a freely chosen betweenness setting (3.5/5) implying that they value the 100% setting the most, even when they are free to choose. This can indicate that participants have a feeling for these correlations in Experiment 1.2: Evaluating citations and recommendation score and Experiment 1.3: Correlation between betweenness and H-index or that the benefits of an influential co-author are just more apparent to them than the benefits of similar co-authors. Some feedback of the participants mention the non-transparency of the keyword similarity factor in the recommendation.

A PhD candidate:
“Some explanation on the keyword similarity, when the recommendations are given (why this person, what is the overlapping topic of interest?)”

Another PhD candidate:
“A nice addition to the tool would be if you look at the similarities between yourself and another author, for example a comparison of used keywords.”

It is logical that the valuations for the recommendations for 1a and 2a are the same because the individual session and the default user session differ in the keywords that are used in the recommendations. If the keyword similarity importance is 0, then the two recommendations will be the same. For the same reason we would have expected to see a lower valuation for recommendation for 2b compared to 1b. The fact that they are the same again is a sign that users do not find the keyword similarity very important. When we look at the distribution of valuations for all questions we see that participants value 2a higher than 1a. Considering that the recommendations for these two questions are very much alike we might ascribe the more positive valuation under 2a to the expectation level of the participants. Participants might expect a better result under 1a because it is personalized for them. They might expect less under 2a because it is a recommendation for a default user. When they set the same recommendation under 1a and 2a of against their expectations, then they arrive at a higher valuation of 2a than 1a.

7.3.2.3 SUS
The overall SUS score is almost exactly the average score of any SUS test. And although the scores on the usability test are on average moderately positive there are naturally a few points in which the recommenders usability could be improved to better suit the task at hand.

Questions with lesser scores are questions 5 and 1 (58% < 3). This means that participants feel that the functions of the system could be better integrated. For this thesis we focussed primarily on
aggregating and calculating the data needed for the recommendations. The recommendations themselves are not that transparent and the functions and the data are spread out over multiple pages. A future version of the system could integrate relevant data and functions more on each page. The scores for question 1 suggest that participants do not think that they would use the system very often. This could be due to the fact that it is not often that you are looking for a new co-author.

It is also clear that improvements on the documentation of the system at the appropriate points could enhance the usability.

An Assistant Professor:

“What's 'betweenness' and 'norm.betweenness'; there are lot's of '0 ' in case of 100% interests... don't understand.”

The betweenness and normalized betweenness are bound to be low when the interest setting is at 100%. In that case only the keyword similarity is counted in the weighted recommendation process and betweenness of a candidate co-author is ignored. The betweenness is not evenly divided over the users so when it is not considered during the recommendation process a lot of solutions with betweenness 0 will come up.

7.4 External Validity

Because networks with the same network model show the same characteristics, we propose that if the co-operation network of our university is comparable to that of other universities the conclusions about recommendations based on our data should have some validity for recommendations based on data of other universities. Some co-authorship network variables are remarkably alike over different universities and within research fields (Newman, M. E. J., 2001) (Newman, M. E. J., 2004). These variables show that all investigated co-authorship networks are scale free and that they all show the small world phenomena. Hence the size of the co-authorship network is not of great influence on the average distance, degree distribution and the cluster coefficient.

In order to check if our network can be described by a scale-free model we compare the degree distribution, average distance and cluster coefficient of the co-author graph of our university with that of a generated scale-free graph and that of a generated random graph. The results can be found in Table 24. Other universities are best fitted by the scale-free model. If our university co-authorship network is also best fitted by this model then we know that the behaviour of the network of our university is comparable to that of the other universities. This is some external validation for our results. The random graph is created with the algorithm described under 1 in Random model on page 16 because we know the number of nodes and edges we want to mimic. This is the number
of nodes and edges in the co-author graph of our university. The scale-free graph is created with the Barabási algorithm described in Generating scale-free networks on page 16.

Next we will calculate overall network variables and compare these to network variables found for other universities (Dijk, D. V., Engelen, B. J. V., Bouwhuis, S. G., & Christidis, A., 2006). If they are comparable then this might be an indication that our recommender would work equally well for different universities. The results can be found in Comparison to universities.

7.4.1 Comparing the university network to network models

In order to verify that our network is a scale-free network we compare the degree distribution in our network to that of a generated scale-free network and a generated random network of the same size as our university network. The three degree distributions can be found in Illustration 37 below.

We can see that the degree distribution of the university network plotted in red is much better fitted by the degree distribution of the generated scale-free network plotted in black than by the degree distribution of a generated random network plotted in blue. Looking at Table 24 we see that although the random network has a average distance comparable to our university network, it has a much smaller cluster coefficient. The scale-free network is also comparable in average distance, but also comes much closer to the cluster coefficient of the university network. Hence we conclude that our university network is a scale-free network like the other university networks. The conclusions about COCOON CORE based on the submission data of our university therefore have some external validity.
Table 24: Generated network metrics versus University network metrics

<table>
<thead>
<tr>
<th>Network Type</th>
<th>Average distance</th>
<th>Clustering coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random network</td>
<td>3.52</td>
<td>6 * 10^-3</td>
</tr>
<tr>
<td>Scale Free Network</td>
<td>1.99</td>
<td>0.02</td>
</tr>
<tr>
<td>University Network</td>
<td>4.33</td>
<td>0.41</td>
</tr>
</tbody>
</table>

7.4.2 Comparison to universities

7.4.2.1 The university co-authorship network 2012

If we look at our university co-authorship network (Illustration 38: Co-authorship graph university 2012) we see that although it is not connected, the largest component is composed of a very large portion of the total number of authors (1240 of 1361). We can see a lot of hubs, primarily in the
centre of the network.

The network variables of our university network are presented in Table 25. In the previous subchapter we confirmed that the network is a scale-free network and we therefore expect the small world phenomenon to appear in the network. We can indeed see that the average shortest path between two authors is 4,33 hops, well short of the 6 degrees of separation put forward by Milgram.

```
<table>
<thead>
<tr>
<th></th>
<th>authors</th>
<th>papers</th>
<th>Papers per author</th>
<th>Authors per paper</th>
<th>Average co-authors</th>
<th>Relative size of largest component</th>
<th>Average distance</th>
<th>Average Harmonic distance</th>
<th>Clustering coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>University</td>
<td>1361</td>
<td>2926</td>
<td>6,66</td>
<td>3,1</td>
<td>9,01</td>
<td>91,11%</td>
<td>4,33</td>
<td>-</td>
<td>0,41</td>
</tr>
</tbody>
</table>
```

*Table 25: Co-authorship network data university 2012*

**7.4.2.2 Other universities**

When we look at the global network variables of other universities (Table 26 on page 110) we can see a number of interesting points. First if we compare the VU to the UVA we see that the UVA repository has ten times the number of authors. This however does not prevent the other network variables to be remarkably alike. This again shows that the underlying model of the networks is scale-free.

Furthermore if we compare the TU Delft with Caltech we see that the two universities show very
similar network variables. This underscores the observation that network variables are dependant on the research field (Newman, M., 2001).

If we compare the variables in Table 26 to those of our university in Table 25 we see that our network is smaller than any of the other networks under examination. This however does not have to be a problem because of the scale free-nature of the networks. The density of our network (average collaborators) ranks amongst the highest and is comparable to that of the VU and UVA. We know that submissions for our network have been added per department. This might be an advantage to the density because density within a department is higher than between departments. Since we have no information about the distribution of papers over departments for the other universities we can not conclude is this is indeed an explanation for the relatively high density of our network. The relatively high clustering coefficient in our network is also a sign that our data is composed of a number of departments. The average largest component of our network is also very high. This negates the necessity to calculate the average harmonic distance which corrects the distance for nodes that are unreachable due to the disconnectedness of the network.

### 7.4.3 Summary of external validity

The co-author network of our university is comparable to that of other universities. The network variables of our university network are in the same order as those of other universities and the degree distribution, average shortest path and cluster coefficient values, like for other universities, indicate a strong fit to a scale-free network. This makes a good case for the external validity of this thesis, but definite conclusions about performance on other universities will have to follow research based on the actual data of these universities.

### 7.4.4 methodology

#### 7.4.4.1 Cluster coefficient of scale-free networks

Although the degree distribution and average shortest path of our co-authorship network is very similar to that of a generated scale-free network, the cluster coefficient though closer than that of a
random network is still to low. Networks generated with preferential attachment tend to mimic the
degree distribution, but do not generate the clusters that are often associated with research groups.
There could be a better fit with a slightly different model. It might therefore be debatable whether we
can fully identify our network as a scale-free network. We point out that the cluster coefficient for
the scale free network is still much closer to that of the university network than the cluster
coefficient of the random network is. Moreover the cluster coefficient of our network is comparable
to those of the other university networks so we at least keep the conclusion the our network is
comparable to other universities networks.

**7.4.4.2 Lack of keywords network data for other universities**

We do not have any data for the keywords in the networks of other universities. Experiments 1 and
2 showed that the emphasis of the recommendation value lies on the betweenness of the co-author
in the author network. The lack of keyword data might therefore not have a big impact on the
external validity of the conclusions for this university.

**7.4.5 Results of external validity study**

In general we can conclude that our network is modelled by the same principals as that of other
universities. Hence our university network is comparable to other university networks. This gives
some confidence that our recommender can get the same moderately positive user valuations for
other universities. It is however also interesting to see how well our system performs on other
universities quantitatively. Next to further tweaking in the recommendation algorithm it is also
interesting how this algorithm performs on other universities data.

Furthermore the individual network variables most closely resemble those of the UvA. It is
interesting to know if this is a result of similar departments being included in the DSpace repository
or that there are more fundamental similarities between the two universities.

Finally we must realize that it has to be empirically proven that the conclusions for COCOON
CORE on this university will hold for other universities, but this comparison will at least give some
initial insight into the external validity of the conclusions of this thesis.
8 Conclusion

8.1 Summary of what we have done

For this thesis we have built a weighted hybrid recommender system which incorporates the betweenness of an author in the co-author network and the keyword similarity of authors in the network to recommend new co-authorships. The recommender is able to take new weight settings and add target keywords for each run. The system also builds a model of the keyword vector for each user. We also build a front end so that researchers can use the system without any technical knowledge and get a more transparent overview of the network they operate in.

Next we ran experiments to see if the recommender recommends successful co-operations in the available data over unsuccessful co-operations. We also ran experiments to see which factors have an determine the scientific output influence of a researcher. Furthermore we held user evaluation tests to see how users value the system. Both with respect to its functionality as to its usability.

Finally we compared the network statistics of our universities co-operation network to that of other university networks to see if the results of our recommender are likely to apply if we were to feed it information of other universities.

8.2 Conclusions

The co-operations of recommended co-authors are not consistently significantly more valuable that those of non recommended co-authors. However, when using citations as a target variable and giving almost all the weight of the recommendation to the power of the author in the network there is a weak positive correlation between the recommendation score of an author and the number of citations this co-operation got. Individual citations overall are better suited than the H-index to find a capability of the recommender to predict valuable co-operations due to the greater discriminating power.

Furthermore we can conclude based on correlation studies that the power of an author in the network is much more important to the successfulness of a co-operation than the similarity of the interest between authors.

Looking at the user evaluation study we can conclude that overall the valuation of COCOON CORE is moderately positive. Participants are especially interested in the powerful user aspect of the recommender, but the similar author option also scores a more than average valuation.

The overall score on the usability test is almost exactly the average score of any SUS test. And although the scores on the individual parts of the test are on average moderately positive we
conclude that participants feel that the functions of the system could be better integrated.

8.3 Similarities with existing research

With respect to the hybrid recommender approach using social network analysis we are not aware of any results of similar projects. We did find a strong positive correlation between the betweenness and the H-index. Earlier research showed no relationship between the betweenness and the g-index (Abbasi, A., Altmann, J., & Hossain, L., 2011). These two observations seem to be at odds with each other. The results of the SUS tests are very similar to many applications of the SUS test because the system scored the average score.

With respect to the similarities with other universities our conclusions are very similar to those of Newman (Newman, M. E. J., 2004). Although the co-authorship networks vary in the exact values for their network variables, they all show the small world effect and are modelled best by the scale-free model.

8.4 Implications

Although previous research emphasized that author interest similarity is very important for scientific output, the results of this thesis indicate that the influence of a co-author is a far better predictor for successful co-operations. This result is the most significant finding of all the experiments we conducted. It clearly indicates that in our data the centrality of an author and the influence of his or her work are strongly connected. Considering that we can not conclude a causal relationship between the two variables we must state that there are many ways in which this correlation could have occurred. One might imagine authors of high quality scientific output might benefit from the attention his or her outputs gets by asking a powerful colleague to co-author the paper. This colleague might only be inclined to do so if the paper indeed has a high quality. Herein lies the challenge for authors not only to produce scientific output of high quality, but also to get this output under the attention of the scientific community. A balance of interests that might require more recommendation variables and/or -techniques than we have used in this thesis to come to a balanced recommendation.

The valuations and interest of the participants of the user study do indicate that there is a demand for the data and conclusions a scientific co-operation recommender system might offer. Getting moderately positive valuations for recommendations might show that the recommender is as good in selecting recommendations as users expect and are used to. Improving on the predictive qualities of the recommender in future research might in that case increase the scientific output of the authors using the recommender. Furthermore if users get the results they more or less expect in their direct vicinity, then the recommender has a significant role to play when we increase the scope of the data,
including more authors from more universities.

We will conclude by saying that using social network variables in a hybrid recommender system is to our knowledge a gap in the research field. None of the conclusions of this thesis are definitive for the research field and there is a lot more to be explored. For some ideas on the next steps we refer to Future research.
9 Future research

9.1 H-Index as recommendation variable

In our study we used the H-index as a target variable, but the H-index can also be used as part of the profile of the user. We now used network influence (Betweenness) as a recommendation variable, but we could also have used scientific output influence (H-index). From experiment 1.3 we know that the two are strongly correlated, but we could give users the choice over which they find more important for their next research project. The delta H-index is also a good predictor for the function of the user. Hirsch argues that a normal researcher's H-index increases by 1 every year. A good researcher's H-index increases by 2 every year. An exceptional researcher's H-index increases by 3 every year. Also he states that a assistant professor has an H-index smaller than 12, an associate professor has an H-index between 12 and 18 and full professors have an H-index of 18 or higher. This could be used in the recommendation. As a professor for instance might have greater time constraints than a PhD candidate. When merging data from multiple universities and therefore multiple repositories we have to realize that the inconsistent naming of authors will become a problem as opposed to in this thesis where all authors were uniquely identified in the data.

9.2 Combine data from multiple universities

We used the data of one DSpace repository, but in line with ever increasing globalization in the research community, it would be very interesting to look at recommendations based on the combined submissions of multiple universities. Where finding likeminded researchers in one university or department might be clear task, finding likeminded researchers in other universities definitely is not.

An Assistant Professor:

“The influencers recommendations provide insight into influencers in the system but for known people will not provide new insights when one is familiar with the community on which the dbase derives its recommendations. It might provide triggers for influential co-authors. When one is new to the community or the dbase/community is very large”

Leydesdorff and Wagner look at the international collaboration of authors. They state that the number of international collaborations grow linearly but the citations to these collaborations grow exponentially. This would indicate that international collaborations result in highly influential research. Furthermore Leydesdorff and Wagner state that the core of the international collaboration gets smaller and more densely connected (Leydesdorff, L., & Wagner, C. S., 2008).

When fed with a combined international database of submissions COCOON CORE would also
make it possible to find the hubs between universities or research fields and might aid in international proposals for European research project budgets.

9.3 External source for keywords
In our dataset keywords are linked to papers and via papers to users. This limits the number and the range of keywords to that of the DSpace database. Future research could use DBpedia to extend the keyword set used by the recommender. A challenge of course is which semantic relationships between keywords in DSpace and other words in the RDF store justify adding a keyword user link. Synonyms is an obvious relationship, but their might be others.

9.4 Use closeness or degree as centrality measure
The rise of social network technologies in the scientific field changes the spread of information. We assumed that this would lead to a more targeted and directed disperse of information. However it could also result in the spread of information in the scientific network by means of copying in multiple directions. In that case in future research we might best use closeness centrality or degree centrality. Because in this case it is not given that the information travels in one single direction over a given shortest path. In this case the number of connections and average distance have more influence on the likelihood that information passes a node than the number of times that a node is on the shortest path.

9.5 Tweak hybrid system
Future research could focus on tweaking the hybrid system. The weighted technique which uses a universal weight for all users in the automated runs might not be the most ideal combining technique. Maybe switching is more appropriate in which the user profile is of influence on the chosen recommendation technique. Alternatively we could keep the weighted technique and learn from users how they adjust the weights and perform collective filtering to match the weight settings of similar users.

9.6 Network centric recommender
We approached the recommendation from the perspective of the user. The goal is to give a recommendation that will optimize the scientific output of the user. If we shift the perspective to the entire network, the goal is to give recommendations that optimize the scientific output of the whole network; in this case our university. The question is: are there optimal network configurations that maximize the scientific output of the network. And if there are, can we build a recommender that makes recommendations that build the best network?
10 References


Cartwright, D., & Harary, F. (1977). A Graph Theoretic Approach to the Investigation of System-


Euler, L. (1956). Seven Bridges of Konigsberg Leonhard Euler. Men and Numbers.


Appendix A – Graph theoretical concepts

**Number of authors:** The number of authors states the total number of different authors in the network. In graph terms this number corresponds to the total number of nodes in the ‘author-author’ graph.

**Number of papers:** This is the total number of papers in a network. In graph terms this corresponds to the number of paper-nodes in an ‘author-paper’ graph.

**Papers per author:** This is the average number of papers authors (co-)authored. In graph terms this number corresponds to the average out-degree of the author nodes in the ‘author-paper’ graph.

**Authors per paper:** This is the average number of co-authors per paper in a network. In graph terms this corresponds to the average in-degree of the paper nodes in the ‘author-paper’ graph.

**Average collaborators:** This is the average number of authors an author collaborated with in the network. In graph terms this corresponds to the average degree of all nodes in the ‘author-author’ graph.

**Largest component:** This is the largest number of authors directly or indirectly connected to each other. Two authors are connected if they co-authored a paper or are connected through co-authorships of other authors. In graph terms this corresponds to the largest number of nodes that are interconnected in the ‘author-author’ graph. Two nodes are linked if there is a path from one node to the other.

**Second largest component:** This is the second largest subset of connected authors.

**Connected graph:** A graph that consists one one component. In other words: all nodes in the graph can be reached from all other other nodes in the graph.

**Complete graph:** A graph in which all nodes share an edge with all other nodes.

**Average distance:** This is the average shortest number of steps one has to take to get from one author to another only using co-authorship of a paper as hops. In graph theory this corresponds to the average shortest path between two nodes in the ‘author-author’ graph. Only existing paths are counted.

**Average harmonic distance:** Technically speaking the distance between two disconnected nodes is infinite. The average harmonic distance compensates for the disruption multiple components have on the average distance. Non-existing paths are counted, but do not let the mean approach infinity because of the division.

**Adjacent Nodes:** Two nodes are adjacent if they share an edge.
**Connectivity k:** A connected graph is k-connected if it takes the removal of k nodes for the graph to break into two components. It is important that we speak of intensionally selected nodes and not randomly selected nodes.

**Cutpoints:** A set of nodes that, if removed, split a connected graph in two components.

**Connectivity λ:** Same as for connectivity k, but for edges instead of nodes

**Bridge:** A bridge is a single edge that, if removed, disconnects a connected graph into two components. A connected graph with a connectivity λ of 1 therefore has at least one edge.

**Hub (or broker):** Same as a bridge, but for a node in stead of an edge.

**Cliques (co-author-based):** Cliques are subsets of nodes in a graph. All nodes in a clique are connected to all other nodes in a clique. In other words: a clique is a subgraph that is complete.

**Maximum clique:** A clique that can not be extended with another node. A maximum clique is therefore not a subset of any other clique.

**N-cliques:** An N-clique is a subset of nodes that are completely connected by paths of length N.

**Factions:** A weaker form of clusters. Nodes in a faction do not have to form a complete graph, but do have a higher density in their subgraph than with nodes outside of the faction.

**Network density:** The number of existing edges divided by the number of potential edges. The total number of potential edges is the number of edges in a complete graph with a given number of nodes.

**Clustering coefficient:** The clustering coefficient is the change that two nodes share an edge given that they have a common neighbour.
Appendix B – Implementation workflows

Offline Processes

Calculation processes

batch_get_results.php

1. Loop over graph types (author-author and keyword-keyword)
   1. Loop over years (start year to current year)
      1. request graph type for year (using CURL to query REST servers get_author_graph.php and get_keyword_graph.php)
      2. write graph in xml that the server returned to the working directory with a descriptive name.

calculate_akf_ikf.php

1. For each author
   1. For each keyword
      1. For each year
         1. get the author keyword frequency by counting the number of links between the author and the keyword via submission in or before the year under current evaluation
         2. if the author keyword links > 0
            1. get the total number of author keyword links for this year. These are all links between any author and any keyword for this year. (total links)
            2. Get the total number of author keyword links between the keyword current evaluation with any author for this year. (keyword link frequency)
            3. calculate the inverse keyword frequency by log(total links / 1 + keyword link frequency). We use the addition of one in the divider to avoid dividing by 0
            4. calculate akf * ikf by author keyword frequency * inverse keyword frequency

calculate_author_metrics_from_citations.php

1. For each author
1. For each year
   
   1. get all relations between authors and citations via author_links and submission links for the current author where the submission is of or before the year of this iteration
   
   2. get the number of rows to get the number of citations
   
   3. get all distinct relations between author and submissions where the submission is of or before the year of this iteration
   
   4. get the number of rows to get the number of submissions for this author for or before this year of this iteration
   
   5. for each submission
      
      1. get the total number of citations for submission
      
      2. add that count to an array (citations_counts)
   
   6. if the count of citations_counts > 0
      
      1. the single_most_citations_google_scholar is 0
   
   7. else
      
      1. the single_most_citations_google_scholar is the submission with the maximum value in citations_counts
   
   8. use array_count_values to go from 'submission → citations count' to 'citations count → frequency'
   
   9. make these frequency cumulative so we get for each citations count the number of papers with this citations count or an higher citation count. Results in array count_values_cum
   
   10. if count of count_values_cum = 0
      
      1. h-index = 0
   
   11. else
      
      1. set h-index to 1
   
   12. for i between 0 and 1000
      
      1. if i > single_most_citations_google_scholar → break (the h-index can never be larger than the highest number of citations for a single submission.
      
      2. Set citations count to i.
3. If the current citation count >= that the number of papers with that citation count we set the h-index to that citation count. Because we have at least i papers with i or more citations.

4. Elseif the number of papers with the current number of citations > the current h-index → we set the h-index to that number of citations. Because if we have 8 papers with at least 10 citations then we also have 8 papers with at least 8 citations. And if the current h-index is 7, it should be changed to 8.

13. insert all metrics in database

14. update last_google_calculation to current time

calculate_results.R

1. change working directory from command line argument

2. set different graph types (authors and keywords)

3. load social network analysis library (igraph - http://igraph.sourceforge.net/)

4. loop over graph types

   1. loop over years 2005 to current year

      1. read in graph

      2. calculate en export betweenness distribution

      3. calculate mean betweenness

      4. calculate en export degree distribution

      5. calculate mean degree

      6. calculate mean cluster coefficient

      7. simplify graph (remove duplicate edges)

      8. redo 1-6

5. export results to files

get_results.sh

1. Set a working directory with the current date and time in the name

2. set the start year to 2005

3. call batch_get_results.php with working directory and start year
4. call calculate_results.R

**Update Scripts**

*get_mendeley_author_metrics.php*

1. select all authors (or select only authors without mendeley results if argument -complete is included)

2. for each author
   1. request mendeley metrics from readermeter.org in JSON format
   2. map metrics ids and values in array
   3. for each id – value pair in array
      1. insert mendeley value in author metrics for this author for current year (this web service only returns metrics for the current year)

*get_submission_citation_data.php*

1. fetch all submissions which have not been checked yet

2. for each submission
   1. query google scholar (using the current proxy server)
   2. parse citations and citations link from return html
   3. check for multiple versions or no match page
   4. insert results into database

*get_citations_per_year.php*

1. get all submissions with a google citation number higher than 0

2. for each submission
   1. for each citations page
      1. for each citation on page
         1. parse title of citation
         2. parse authors of citation
         3. parse year of citation (if we can't find the year of the citation we skip the citation all together because we can not include it in our h-index calculations)
         4. insert all metadata of citation in our database
5. update the last time we scanned for citations of this submission with the current timestamp

**get_submission_metadata.php**

1. for each submission
   1. get unique identifier
   2. get record from DSpace using unique identifier using CURL
   3. initiate SimpleXMLElement with return xml
   4. get title and date from xml element
   5. insert title and date in metadata table

**initiate_usernames.php**

1. for each author
   1. get id, first name and last name
   2. replace all spaces in first name and last name with dashes
   3. combine first name and last name with a dash in between
   4. replace all accents with normal letters (this avoids problems when checking usernames)
   5. update author record with username

**transport_reader_meter_results.php**

1. connect to older database
2. select all old metrics values
3. connect to newer database
4. for each old result
   1. map old values to new value ids and store in array
   2. for new value in the array
      1. update the metric value in the new database

**update_author_keyword_links.php**

1. join keywords on keyword_links, submissions, author_links and authors.
2. select all values for author_id, keyword_id, submissions_date
3. truncate existing author_keyword_links table
4. insert all separate new values from select in author_keyword_links table

**update_metrics_from_r.php**

1. Set working directory
2. for each metric (degree, betweenness)
   1. for each subject (authors, keywords)
      1. for each extension ("", _simplified) – This is used to distinguish full graphs from simplified graphs
         1. for all year (2005 – current year)
            1. select filename
            2. read in csv file with File_CSV_DataSource library
            3. for each result in file
               1. check if value exists for that metric, for that author / keyword, for that year.
               2. If value(s) exist(s)
                  1. delete old values
                  3. insert new value

**copy_latest_betweenness_to_authors.php**

1. For each author
   1. get latest betweenness value
   2. update authors table for current author with latest value

**Online Processes**

**Interfaces**

**authors.php**

1. fetch all authors that comply with the search parameters
2. for each author
   1. add to the return string in xml
3. return xml return string

keywords.php

1. fetch all keywords that comply with the search parameters
2. for each keywords
   1. add to the return string in xml
3. return xml return string

get_author_graph.php

1. hash request URL and check if cache file exists. If so, serve this file and exit
2. initiate xml structure
3. If author_id and depth are provided
   1. if depth is 0 add this author to author list and quit
   2. else
      1. add this author to author list
      2. find all authors connected to this author via a submission from or before cut-off year
         1. for each connected author-author
            1. add connection to edge list
            2. process author with depth is current depth -1
      3. add all authors in author list to return GML
      4. add all edges in edges list to return GML
4. if author_id and depth are not provided
   1. fetch all authors with a submission from or before cut-off year.
   2. add all authors to the return GML
   3. fetch all submissions
   4. for each submission
      1. fetch all co-authors an store in array
      2. add edges to return GML for all pairs in co-author array
5. close xml structure
6. safe return GML in cache file
7. return GML

get_author_keyword_graph.php
1. hash request URL and check if cache file exists. If so, serve this file and exit
2. set up GML return structure
3. find all authors with submissions from or before the cut off year
4. add all authors to return GML
5. find all keywords connected to submissions from or before the cut off year
6. add all keywords to return GML
7. find all author-keywords links generated by update_author_keyword_links.php that originate from a submission from or before the cut-off year
8. add all author-keyword links to return GML
9. close xml structure
10. safe return GML in cache file
11. return GML

calculate_recommendation.php
1. get all selected keywords and add to target keywords
2. get top 5 keywords of requesting author an add to target keywords
3. for each target keyword get the maximum akf_ikf for any author – this way we create the ideal vector for these keywords
4. get similar authors
   1. find all authors with a relation to one or more of the target keywords
   2. for each author in step 4.1
      1. calculate author keyword vector – this is the vector of akf_ikfs of all target keywords.
      2. calculate similarity between author keyword vector and target keyword vector. For more documentation on akf_ikf vector similarity see 'Compute author-keyword frequency/inverse keyword frequency vector similarity'.
1. calculate DOT product vectorA DOT vectorB
2. calculate Euclidean Distance for both vectors
3. calculate similarity between vectors – cos DOT product / (1+(ED1*ED2))
4. add similarity score of author keyword vector similarity to array
3. sort author keyword vector similarity array by value and maintain key-value correlation (arsort)
4. return author keyword vector similarity array
5. calculate normalized betweenness
1. for each author evaluated in step 4
   1. get betweenness for cut-off year for this author
   2. normalize betweenness by using log10 (betweenness has a long tail distribution)
2. get the maximum normalized betweenness
6. calculate overall score
1. for each author evaluated in step 4
   1. add betweenness to result
   2. calculate overall score and add to result (keyword_vector_similarity * keyword_similarity) + ((normalized betweenness/maximum normalized betweenness) * Scoop_centrality). We use the normalized relative betweenness.
3. add author name and link to author page to result.
4. add normalized relative betweenness to results
5. add keyword similarity to result
7. return recommendation author array in JSON format

Include
These processes are typically needed on the majority of the pages.

dbconnect
This php script initiates the connection to the COCOON CORE database and is therefore included in almost all processes.

footer
This is the footer included in the front end. Stating the universities involved in this thesis and the years in which the recommender was built.

**get authors**

**input:** cut-off year

**output:** list of authors

This php script returns a JSON encoded array of authors for a given cut-off year. This service is used by the author table on the authors page.

1. get all authors with submissions from or before the cut-off year with current h-index
2. for each author
   1. add author name, h-index and author page link to array
   2. add array to authors array
3. encode authors array as JSON and return authors array

**get keywords**

**input:** cut-off year

**output:** list of keywords

This php script returns a JSON encoded array of keywords for a given cut-off year. This service is used by the keywords table on the keywords page.

1. get all keywords linked to submissions from or before the cut-off year with current betweenness
2. for each author
   1. add keyword name, betweenness and author page link to array
   2. add array to keywords array
3. encode keywords array as JSON and return keywords array

**nav**

This is effectively the header of all pages. It checks for errors or success of the update password process and displays the page header with a welcome message, an account drop down, a log out button, the general search box and the title of the page. This file also outputs the general navigation to the left of the page.

**user info**
This script sets some basic user information to be used on all pages. The information is also stored in cookies on the users computer. The information stored is:

- first name
- last name
- user name

Among other places this information is used in the header of the page.

**input interaction**

This javascript file includes functions to AJAX calls and page update functions used in the user interaction on the deprecated data explore page visual.html. This page is not linked in the final recommender, but can be accessed @ http://base_url/visual.html.

**process form**

This javascript file contains functions to process the recommendation form on the dashboard page. All functions can be found in process_form on page 134.

**input:** none  
**output:** none

This function reads the recommender variables, queries the recommendation service and presents the recommendations in a table.

1. unhide result table
2. get max cutoff year
3. get selected keywords
4. check is the number of selected keywords is between 1 and 6
5. get author co-operation importance
6. get keyword similarity importance
7. request recommendations at /include/calculate_recommendation.php
8. set first column (total score) to sort descending
9. update recommendation table with recommendations

**visual update**

**input:** graph_type_selection,container,object_id  
**output:** graph of type graph_type_selection
This javascript file contains one function that is used to draw all graphs on COCOON CORE pages.

**input_interaction**

This javascript file includes functions to AJAX calls and page update functions used in the user interaction on the deprecated data explore page visual.html. This page is not linked in the final recommender, but can be accessed @ http://base_url/visual.html.

- **getKeywords**: gets keywords from keywords.php and calls updateKeywordTable
- **updateKeywordTable**: update keywords table on keywords page with keywords
- **appendToKeywordTable**: append one keyword to table with keyword results
- **getOptions**: gets keywords from keywords.php and calls updateOptions
- **updateOptions**: updates the options for the keyword in the recommender settings
- **appendLink**: link to keyword results from search
- **clearOptions**: clear all options in keyword table (used when search changes)
- **readReturn**: read return on form and ignore
- **add_keyword**: append one selected keyword
- **updateWarning**: update warnings from cytoscape

**process_form**

**input**: none

**output**: none

This javascript file includes AJAX calls and page update functions used in the user interaction of the recommender settings form on the dashboard page.

- **update_sliders**:
  - **input**: changed slider element ID
  - **output**: none

The two sliders in the recommender settings need to add up to 100 %.

1. read the value of the changed slider
2. update the value of the other slider to 100-value of changed slider

**update_slider**:

**input**: slider element ID, value
output: none

1. update the value of slider with slider element ID

drawGraph:

input: graph_type_selection,container,object_id

output: graph of type graph_type_selection

1. check layout selection
2. initialize XMLHttpRequest object
3. select service url based on layout selection (author-author, author-keyword, keyword-keyword)
4. get cut-off year and add cut-off year to request
5. if object_id is given add that as author_id to request (this is the case when a graph needs to centre around an specific author)
6. send request
7. set style according to graph type.
   1. In an author-author graph the size of the nodes are relative to the betweenness of the author and the centred author is depicted in a different colour. The label is initialized with the authors name.
8. initiate cytoscape web visualization
9. add a listener function for a click on a node. A click results in redirection to an author page
10. draw graph in container.

Miscellaneous

These processes enable the front end to communicate with the database or with other online processes, but do not contain execution logic.

logout

input: none

output: none

This php script enables the user to log out of COCOON CORE. It removes all browser cookies from this domain en redirects the user to the login page

search
**input:** search terms

**output:** keywords and author

This process returns a JSON encoded list of keywords and authors that contain the search term. This process is called by the search function in the header of the front end.

1. query keywords.php with search term and max results = 5
2. push all results into results array
3. query authors.php with search term and maxresults = 5
4. push all results into results array
5. return JSON encoded return array

**submit**

**input:** username and password

**output:** error codes

This php script handles the user login.

1. check is username and password are provided
2. check is username is in database
3. check md5 hashed password against database
4. return with error code or set user_id cookie en redirect to dashboard

**update password**

**input:** old password, new password, new password confirm

**output:** error message or success message

This php script changes the password of the currently logged in user.

1. if user is not logged in, redirect to login page
2. check old password against database
3. check if new password and new password confirm match
4. update password in database and redirect to dashboard

**logout.php**

**input:** none
output: none

This php script enables the user to log out of COCOON CORE. It removes all browser cookies from this domain and redirects the user to the login page.
Appendix C – Recommendation valuation questionnaire

1a. Individual Recommendation: How do you value the recommendation that is generated if the slider for influence is set to 100?

1b. Individual Recommendation: How do you value the recommendation that is generated if the slider for interest similarity is set to 100?

1c. Individual Recommendation: How do you value the recommendation that is generated if you control the sliders yourself?

2a. Default User Recommendation: How do you value the recommendation that is generated if the slider for influence is set to 100?

2b. Default User Recommendation: How do you value the recommendation that is generated if the slider for interest similarity is set to 100?

2c. Default User Recommendation: How do you value the recommendation that is generated if you control the sliders yourself?
## Appendix D – SUS Usability questionnaire


<table>
<thead>
<tr>
<th>#</th>
<th>Question</th>
<th>Strongly disagree</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>Strongly agree</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I think that I would like to use this system frequently</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>I found the system unnecessarily complex</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>I thought the system was easy to use</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>I think that I would need the support of a technical person to be able to use this system</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>I found the various functions in this system were well-integrated</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>I thought there was too much inconsistency in this system</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>I would imagine that most people would learn to use this system very quickly</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>I found the system very cumbersome to use</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>I felt very confident using the system</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>I needed to learn a lot of things before I could get going with this system</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Question</td>
<td>Recommendation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>----------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Increase the frequency of communication with stakeholders.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Conduct regular meetings with team members to gather feedback.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Implement a feedback mechanism for continuous improvement.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Establish clear goals and objectives for the project.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Ensure that all team members have access to necessary resources.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Encourage open communication and collaboration among team members.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Regularly review and adjust project plans based on feedback.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Provide training and development opportunities for team members.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Foster a positive work environment to promote productivity.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Regularly evaluate project performance and make necessary adjustments.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Maintain open lines of communication with all stakeholders.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Continue to implement feedback mechanisms to improve project outcomes.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Appendix F – Responses Experiment 2 (SUS)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------------------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Question</strong></td>
<td><strong>Response</strong></td>
<td><strong>Description</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How easy was it to find your way around the system?</td>
<td>Easy</td>
<td>User-friendly navigation.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How easy was it to understand the goals of the system?</td>
<td>Easy</td>
<td>Clear and concise objectives.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How easy was it to learn how to use the system?</td>
<td>Easy</td>
<td>Intuitive user interface.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How useful was the system in completing the tasks?</td>
<td>Very useful</td>
<td>Efficient and effective.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How well did the system support you in accomplishing your goals?</td>
<td>Well</td>
<td>Effective and comprehensive support.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How satisfied were you with the overall experience?</td>
<td>Very satisfied</td>
<td>Positive and enjoyable.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How likely are you to use this system again in the future?</td>
<td>Very likely</td>
<td>High potential for repeat usage.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How likely are you to recommend this system to others?</td>
<td>Very likely</td>
<td>Strong endorsement.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>How would you rate this system?</td>
<td>Excellent</td>
<td>Top performance.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: The above responses are based on the SUS (System Usability Scale) survey.
# Appendix G – Original database tables

<table>
<thead>
<tr>
<th>Table</th>
<th>ER – authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>authors</td>
<td>id</td>
</tr>
<tr>
<td></td>
<td>firstName</td>
</tr>
<tr>
<td></td>
<td>lastName</td>
</tr>
<tr>
<td></td>
<td>betweenness</td>
</tr>
<tr>
<td></td>
<td>betweenness_2011_bj</td>
</tr>
<tr>
<td></td>
<td>last_google_calculation</td>
</tr>
<tr>
<td></td>
<td>username</td>
</tr>
<tr>
<td></td>
<td>password</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table</th>
<th>ER – author_links</th>
</tr>
</thead>
<tbody>
<tr>
<td>author_links</td>
<td>id</td>
</tr>
<tr>
<td></td>
<td>author_id</td>
</tr>
<tr>
<td></td>
<td>submissions_id</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table</th>
<th>ER – keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>keywords</td>
<td>id</td>
</tr>
<tr>
<td></td>
<td>name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table</th>
<th>ER – keywords_links</th>
</tr>
</thead>
<tbody>
<tr>
<td>keywords_links</td>
<td>id</td>
</tr>
<tr>
<td></td>
<td>keywords_id</td>
</tr>
<tr>
<td></td>
<td>submissions_id</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table</th>
<th>ER – submissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>submissions</td>
<td>id</td>
</tr>
<tr>
<td></td>
<td>unique_identifier</td>
</tr>
<tr>
<td></td>
<td>date</td>
</tr>
<tr>
<td></td>
<td>language</td>
</tr>
</tbody>
</table>