Semantically Rich Recommendations in Social Networks for Sharing, Exchanging and Ranking Semantic Context

Stefania Ghita, Wolfgang Nejdl, and Raluca Paiu

L3S Research Center, University of Hanover,
Deutscher Pavillon, Expo Plaza 1, 30539 Hanover, Germany
{ghita, nejdl, paiu}@l3s.de

Abstract. Recommender algorithms have been quite successfully employed in a variety of scenarios from filtering applications to recommendations of movies and books at Amazon.com. However, all these algorithms focus on single item recommendations and do not consider any more complex recommendation structures. This paper explores how semantically rich complex recommendation structures, represented as RDF graphs, can be exchanged and shared in a distributed social network. After presenting a motivating scenario we define several annotation ontologies we use in order to describe context information on the user's desktop and show how our ranking algorithm can exploit this information. We discuss how social distributed networks and interest groups are specified using extended FOAF vocabulary, and how members of these interest groups share semantically rich recommendations in such a network. These recommendations transport shared context as well as ranking information, described in annotation ontologies. We propose an algorithm to compute these rankings which exploits available context information and show how rankings are influenced by the context received from other users as well as by the reputation of the members of the social network with whom the context is exchanged.

1 Introduction

This paper explores how we can use communication in social networks to share and extend context information and how semantically rich recommendations between members of interest groups in such settings can be realized. We will build upon FOAF networks, which describe personal and group information, based on the FOAF vocabulary to describe friends, groups and interests. We will focus on how to share context in such a network, how to use these shared metadata to connect the information of different peers in the social network and how to use it for social recommendations.

The next section describes a motivating scenario that shows how context and rankings are exchanged inside a research group. Section 3 discusses how to describe contexts and their corresponding metadata by means of appropriate ontologies and how to use these metadata for extended desktop search with appropriate ranking of search results. Section 4 describes how we exchange context and importance information among the members of an interest group defined through an extended FOAF vocabulary introduced in section 4.1. Section 4.3 presents the algorithm used for the rank computation, together with the results we get after applying the algorithm on a user's context metadata before and after she receives additional resources and context from another group

Y. Gil et al. (Eds.): ISWC 2005, LNCS 3729, pp. 293–307, 2005.

member. We discuss the influence of user trust upon ranking results and comment the results of some experiments. Section 5 gives an overview of related work. We then conclude and sketch some future research issues.

2 Motivating Scenario

As our motivating scenario, let us consider our L3S Research Group context and within this group, Bob and Alice as two members who exchange information. One important task in a research group is exchanging and sharing knowledge, which we will focus upon in this paper. Unfortunately, the most widely used infrastructure for this purpose, email, is poorly suited to support this exchange. When we exchange documents by email, no context is shared (for example which are the interesting follow-up papers, or which are the interesting references for a paper) and any comments about the documents that are included in the email are lost as soon as the attached documents are stored in some directory.

The following example shows how such a sharing scenario can be supported in a more efficient manner. We assume that Bob mails Alice a document which he sent to the DELOS Workshop, with the title "I know I stored it somewhere - Contextual Information and Ranking on Our Desktop". Bob is one of the authors and therefore he already has all the important context for this paper including the cited papers stored on his computer. In this first email, Alice will therefore not only receive the paper but also its immediate context relevant for the research group, containing information about all papers that are referenced in the DELOS paper, information about important authors for this topic or which conferences are relevant. In other words, whenever we send a paper, the metadata associated to that paper will also be sent. From the five references included, Alice decides that "ObjectRank: Authority-Based Keyword Search in Databases" and "Activity Based Metadata for Semantic Desktop Search" are of particular interest for her and she sends back an email to Bob requiring additional information about those. As an answer, she receives from Bob the context information associated with these papers, containing the references that Bob has already downloaded. So the context information will be exchanged progressively, from the immediate context to the more distant one.

Figures 1 and 2 present the context created on Alice's desktop as result of her metadata exchange with Bob. Figure 1 contains only the *cites* relationship among the various resources, while in Figure 2 we represent additional relationships, like *presented_at*, *downloaded_from*, *author*, or *same_session*. Note that the context networks created on the users' desktop are not separated, but just visualized separately in these figures.

By examining the context graph in figure 2, we see that all the papers labeled from **G** to **Q** were presented at different WWW Conferences, in different years, and all were downloaded from the ACM Portal. Papers **A**, **C** and **K** all share the same author, *Bob*, and have been downloaded from the L3S Publication page. Similarly, the publication labeled **B** and the other two papers which were presented at the same session at the VLDB conference were downloaded from the VLDB web site.

All this information is taken into account when computing the importance of the resources on Alice's desktop. For example, when computing the importance of the

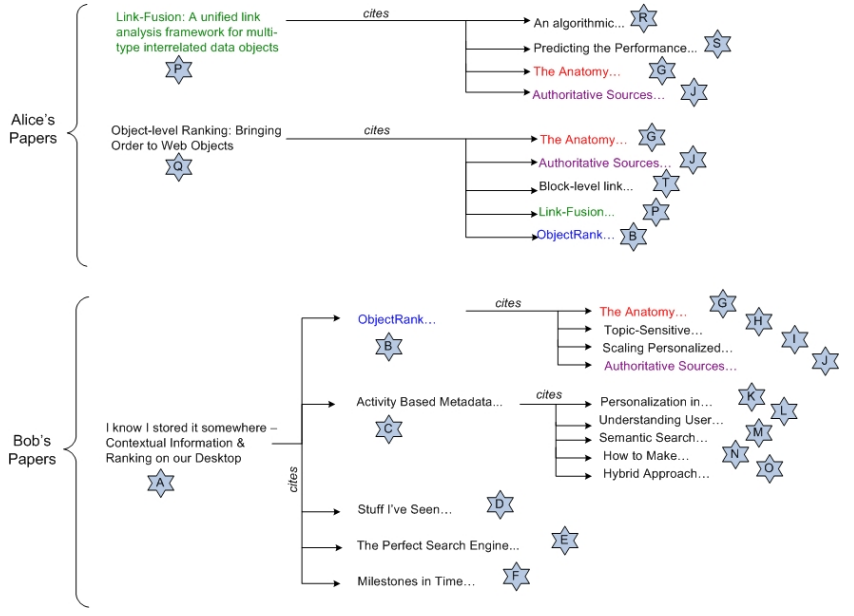

Fig. 1. Publications Context Example - Part 1

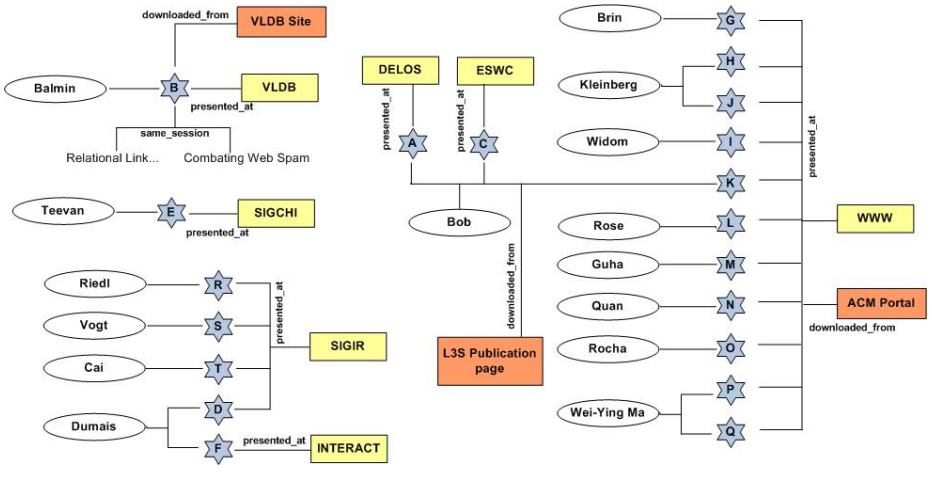

Fig. 2. Publications Context Example - Part 2

conferences, the WWW Conference will be more important than other conferences, since Alice already has a lot of important publications which have been presented there. The number of papers from the same author Alice has already downloaded also influences how important she considers that author. This means that certain authors are more important than others, based on the publications used and cited in the L3S Research

Group, as well as on general citation information about these authors. The fact that Alice knows Bob and Bob is one of the authors of three publications Alice has on her desktop influences the importance of Bob's publications and of course, Bob's importance as author. So he will be definitely more important to Alice than other authors not known to her.

In order to be able to compute the rankings of their documents Alice and Bob have to build a context around the resources they have stored on their desktops. The next section presents in more detail how this context information is created and then describes how this context can be used in computing rankings of search results on the desktop.

3 Representing Context and Importance

3.1 Representing Context

Generally speaking, context information describes all aspects important for a certain situation: ideas, facts, persons, publications, and many more. Context information includes all relevant relationships as well as interaction history. Current desktop search prototypes fall short of utilizing any desktop specific information, especially context information, and just use full text index search. In our scenario we clearly need to use additional context information, and specifically want to exploit the following contexts:

CiteSeer context. The most important aspects we want to record from the CiteSeer context are the publications we are viewing or downloading and how these publications are connected to other publications. Important parts of the available context information are the authors of these publications, the conferences in which they were presented or the year when they were published and even more, the publications which cite them or are cited by them. We want to keep track whether we saved a certain publication on our own desktop in order to be able to find it later and we want to receive suggestions about papers that might be interesting in the same or overlapping contexts.

CiteSeer provides four additional types of links that can be followed after identifying a paper. The most expressive in our case would be the ones that refer to the related documents from co-citations and the papers that appear on the same web site.

Browsing and Desktop context. Browser caches include all information about user's browsing behavior, which are useful both for finding relevant results, and for providing additional context for results. In our scenario, when we search for a document we downloaded from the CiteSeer repository, we do not only want to retrieve the specific document, but also all the referenced and referring papers which we downloaded on that occasion as well.

In general, we view documents stored from emails and from web sites as our personal digital library, which holds the papers we are interested in, plus all relevant contextual information. When we store documents, we can then retrieve them efficiently and restore the original context we built up when storing these documents. Personalized search and ranking on the desktop takes this contextual information into account as well as the preferences implicit in this information. [6] discusses how people tend to associate things to certain contexts. So far, however, search infrastructures neither collect nor use this contextual information.

Scenario specific annotation ontologies. Figure 3 presents our current prototype ontology, used for implementing our motivating scenario. It specifies context metadata for the CiteSeer context, files and web pages, together with the relations among them (described in more detail in [2]). Conceptually, the elements in the rectangles represent classes, circles represent class attributes. We use classes whenever we want to attach importance / rank on entities, attributes otherwise.

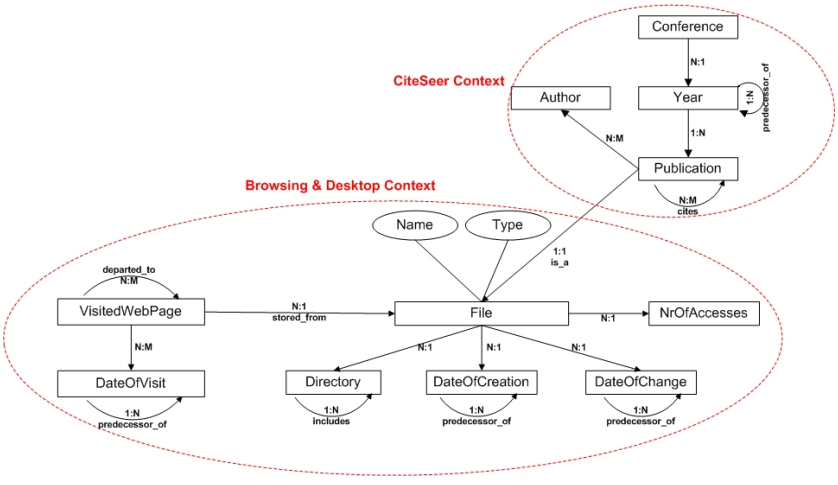

Fig. 3. Context ontology for our prototype

For the browsing and desktop context, we annotate each page with additional information about its basic properties (URL, access date, etc), as well as more complex ones such as in- and out-going links browsed [2]. The user's behavior as the pages or publications he browsed or downloaded provide useful additional information. Files, which are stored from web pages, reside in certain directories, which in turn can include other directories. The creation or change date of a file together with the number of accesses are some other important indicators which have to be taken into account when describing the desktop context. An extended publication ontology makes use of additional knowledge about how CiteSeer pages are connected and what they represent. Publications are referenced by other publications and can cite others, they can have a publication date / year associated with them, as well as a conference or journal. Publications have authors and are stored as documents on the desktop.

Other ontologies describe contexts like conferences, including reviewers, papers, meetings, authors, or private contexts like birthdays, including persons, locations, etc.

3.2 Representing Importance

In addition to the information which resources are included in a specific context, we also want to know how important or valuable these resources are. We therefore have to develop a mechanism which allows us to express this information and use it for ranking search results.

Authority transfer annotations. Annotation ontologies describe all aspects and relationships among resources which influence the ranking. The identity of the authors, for example, influences our opinion of documents so "author" should be represented explicitly as a class in our publication ontology. We then have to specify how these aspects influence each other's importance.

ObjectRank [1] has introduced the notion of authority transfer schema graphs, which extend schemas similar to the ontologies previously described, by adding weights and edges in order to express how importance propagates among the entities and resources inside the ontology. These weights and edges represent the authority transfer annotations, which extend our context ontologies with the information we need to compute ranks for all instances of the classes defined in the context ontologies.[1]

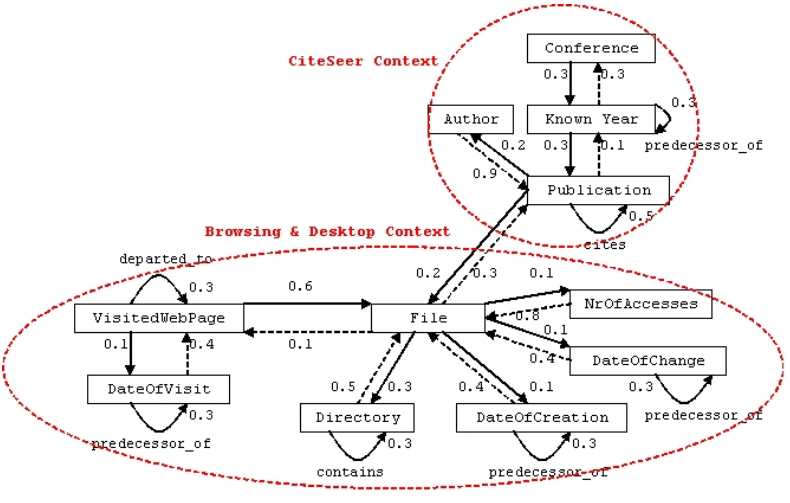

Fig. 4. Authority transfer annotations, including external ranking sources

Figure 4 depicts our context ontology plus its authority transfer annotations. The ontology representing our browsing and desktop context says that a visited web page is important if we arrived at the current one from an important page, if the file under which it is stored is important, or if the date when the page was visited is important. For the CiteSeer context, publications transfer part of their authority to other papers they cite, to their authors, to the files under which they are stored, and to the year when the paper was published. As we can see, citing important papers doesn't make a paper important. As suggested in [1], every edge from the schema graph is split into two edges, one for each direction. This is motivated by the observation that authority potentially flows in both directions and not only in the direction that appears in the schema - if we know that a particular person is important, we also want to have all emails we receive from this person ranked higher. The final ObjectRank value for each resource is calculated based on the PageRank formula.

[1] In contrast to ObjectRank, we do not compute a keyword-specific ranking, but a global one.

Personalized Preferences and Ranking. Different authority transfer weights express different preferences of the user, translating into personalized ranking. The important requirement for doing this successfully is that we include in a user ontology all concepts, which influence our ranking function. For example, if we view a publication important because it was written by an author important to us, we have to represent that in our context ontology.

4 Sharing Context and Importance

4.1 Interest Groups

Interest groups in our context are specialized social networks that have a stated common interest which connects the members of the group. One important reason for creating interest groups resides in increasing the efficiency of the information flow inside that group. All members of the same interest group share the same domain of interest and the social relationships are woven around this type of information sharing. They are all possibly part of the same professional group, just as we described in the motivating scenario, Alice and Bob being in the same research group, the L3S Research Group.

We chose to represent interest groups based on an extension of FOAF in order to describe the social network of participants and we will describe all contexts as RDF metadata, as presented in [2]. Being based on RDF, FOAF inherits some of its benefits, like the ease of aggregating and harvesting it, or combining it with other vocabularies, thus allowing us to capture a rich set of metadata. The basic FOAF vocabulary itself is pretty simple, pragmatic and designed to allow simultaneous deployment and extension. It is identified by the namespace URI 'http://xmlns.com/foaf/0.1/' and described in more detail at the FOAF project page [9].

FOAF terms represent information which can be grouped in the following five broad categories: FOAF Basics, Personal Information, Online Accounts/ IM, Projects and Groups, Documents and Images. The most important for us is the *Projects and Groups* category, which allows us to talk about groups and group membership among others. Groups are represented with the aid of the **foaf:Group** class, which represents a collection of individual agents. The **foaf:member** property allows us to explicitly express the membership of agents to a group. Since the **foaf:Person** class is a subclass of the **foaf:Agent** class, persons can also be members of a group. One can specify the interests of the group members by using specific properties, like **foaf:interest**, **foaf:topic_interest**, or **foaf:topic**, even though it is not yet clear how to use them correctly.

A notable omission in the basic FOAF vocabulary is the inability to express anything related to information sharing in a group. Even though being in a group or social network usually means that we want to share information within this social network, there is no vocabulary to express this in FOAF. The assumption we make in this paper is that people belonging to a common interest group will share a specific set of metadata. In our scenario these are the contextual metadata defined by appropriate annotation ontologies, as discussed in the previous section. When members of an interest group express that they want to share a certain set of metadata, they will agree on an

appropriate ontology defining this set. We therefore suggest to extend the FOAF vocabulary with a new property **foaf:shared_context** which takes as its value the annotation ontology describing the metadata to be shared. Based on this, the FOAF description of the L3S Research interest group and its members Bob and Alice as presented in our motivating scenario looks as follows:

$$
\begin{array}{l}
<foaf:Group> \\
\quad <foaf:name> L3SResearchGroup </foaf:name> \\
\quad <foaf:member> \\
\quad\quad <foaf:Person> \\
\quad\quad\quad <foaf:name> Alice </foaf:name> \\
\quad\quad\quad <foaf:homepage\ rdf:resource = "http://www.l3s.de/\sim alice"/> \\
\quad\quad </foaf:Person> \\
\quad </foaf:member> \\
\quad <foaf:member> \\
\quad\quad <foaf:Person> \\
\quad\quad\quad <foaf:name> Bob </foaf:name> \\
\quad\quad\quad <foaf:homepage\ rdf:resource = "http://www.l3s.de/\sim bob"/> \\
\quad\quad </foaf:Person> \\
\quad </foaf:member> \\
\quad <foaf:shared\_context \\
\quad rdf:resource = http://www.l3s.de/isearch/citeseerContext.rdf/> \\
\quad <foaf:shared\_context \\
\quad rdf:resource = http://www.l3s.de/isearch/browsingDesktopContext.rdf/> \\
</foaf:Group>
$$

4.2 Exchanging Context Within Interest Groups

Sharing context in an interest group is useful and necessary because not only do we want to publish our own work but we also want to find out about additional new resources related to our work and get suggestions about possible further developments in that area. Recommendation then means suggesting additional related information to given items. In our motivating scenario, we have as interest group a set of researchers, and a set of ontologies defining which metadata are shared between them. The contextual metadata corresponding to those ontologies as discussed in section 3 represent the context information we have available on our desktop.

These context metadata are generated locally by a set of metadata generators [2], which record user actions as well as interactions and information exchanges between members of a group. These metadata generators create RDF annotation files for each resource whose context they describe, so for each relevant resource on the desktop (e.g. a specific publication) we will have this additional RDF information available.

For the experiments described in this paper, we have implemented a metadata generator, which deals with publications, and crawls one's desktop in order to identify and annotate all papers saved as PDF files. For each identified paper, it extracts the title and tries to match it with an entry into the CiteSeer publications database. If it finds an entry, the application builds up an annotation file, containing information from

the database about the title of the paper, the authors, publication year, conference and other CiteSeer references to publications. All annotation files corresponding to papers are then merged in order to construct the RDF graph of publications existing on one's desktop.

In our scenario, whenever Bob sends a publication to Alice, who is member of the same interest group, he wants to attach the appropriate context information, i.e. the publication context we have discussed in the scenario. A second (email) helper application therefore checks who is the recipient of the email, which group she belongs to, and therefore which context information/ metadata to attach. On Alice's side, the helper application has to integrate the newly received annotation files into the existing publication graph.

4.3 Sharing Importance

Ranking of Resources - General Algorithm. In our distributed scenario, each user has his own contextual network / context metadata graph and for each node in this network the appropriate ranking as computed by the algorithm described in section 3.2. The computation of rankings on one's desktop is based on the link structure of the resources as specified by the defined ontologies and the corresponding metadata. When sharing information within the group / network we exchange not only contexts but also rankings. So exchanging context information has also an impact on the ranking of results of the desktop search. These values are then recomputed according to the rankings received together with the context from other persons.

Ranking of resources is calculated based on the PageRank formula:

$$r = dAr + (1 - d)e \tag{1}$$

applying the random surfer model and including all nodes in the base set. Parameter d in the equation represents the dampening factor and is usually considered to be 0.85. The random jump to an arbitrary resource from the data graph is modeled by the vector e. A is the adjacency matrix which connects all available instances of the existing context ontology on one's desktop. The weights of the links between the instances correspond to the weights specified in the authority transfer annotation ontology. Thus, when instantiating the authority transfer annotation ontology for the resources existing on the users' desktop, the corresponding matrix A will have elements which can be either 0, if there is no edge between the corresponding entities in the data graph, or they have the value of the weight assigned to the edge determined by these entities, in the authority transfer annotation ontology, divided by the number of outgoing links of the same type. According to the formula, a random surfer follows one of the outgoing links of the current page, with the probability specified by d, and with a probability of *(1-d)*, he jumps to a randomly selected page from the web graph. The r vector in the equation stores the ranks of all resources in the data graph. These rankings are computed iteratively until a certain threshold is reached.

To make these details clear, let us look at the following example: we consider the authority transfer annotation ontology for a publication ontology, as depicted in Figure 5 and then instantiate it. A subset of this data graph is shown in Figure 6.

Fig. 5. Authority transfer annotation ontology for a publication ontology

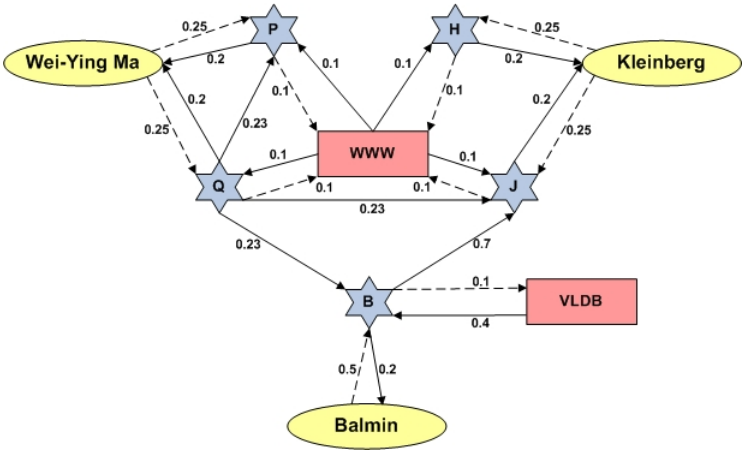

Fig. 6. Data Graph

Table 1. The A matrix

$$A = \begin{array}{c} \\ YingMa \\ Kleinberg \\ Balmin \\ P \\ Q \\ J \\ H \\ B \\ WWW \\ VLDB \end{array} \begin{array}{c} \begin{array}{cccccccccc} Y.Ma & & & & & & & & & VLDB \end{array} \\ \begin{pmatrix} - & - & - & 0.2 & 0.2 & - & - & - & - & - \\ - & - & - & - & - & 0.2 & 0.2 & - & - & - \\ - & - & - & - & - & - & - & 0.2 & - & - \\ 0.25 & - & - & - & 0.23 & - & - & - & 0.1 & - \\ 0.25 & - & - & - & - & - & - & - & 0.1 & - \\ - & 0.25 & - & - & 0.23 & - & - & 0.7 & 0.1 & - \\ - & 0.25 & - & - & - & - & - & - & 0.1 & - \\ - & - & 0.5 & - & 0.23 & - & - & - & - & 0.4 \\ - & - & - & 0.1 & 0.1 & 0.1 & 0.1 & - & - & - \\ - & - & - & - & - & - & - & 0.1 & - & - \end{pmatrix} \end{array}$$

The instantiation of our matrix A from Equation 1 is depicted in Table 1:

According to Figure 6, the authors transfer 0.5 units of importance divided by the number of "authors" links, to their own publications (Wei-Ying Ma to publications P, Q, Kleinberg to H and J and Balmin to B). Publications transfer 0.7 units of importance divided by the number of "cites" links to other papers they cite, 0.1 units to the conferences where they were accepted and 0.2 units of importance divided by the number of "author" links to their authors. Since the papers can be presented to only one conference, the 0.1 units of importance remain undivided.

The values in the matrix are grouped in blocks, formed by considering the cartesian products *Author* × *Author, Author* × *Publication*, etc. The elements from one block can be either 0, if there is no edge between the corresponding entities in the data graph, or they have the value of the weight assigned to the edge between the entities which determine the block, from the authority transfer annotation ontology.

The e vector, modeling the random jump in the PageRank formula, contains an entry for each resource appearing in the data graph. In the original PageRank/ ObjectRank formula, the probability of reaching a certain resource through a random jump is evenly distributed among the resources, and therefore, the e vector has only 1 values:

$$e = (1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1)^T \tag{2}$$

Ranking of Resources on Alice's Desktop. We computed the ranking values for the resources existing on Alice's desktop (see Figure 1 for the labels of resources). The results are presented in Table 2. Note that these values represent Alice's personal rankings according to the context existing around her resources and are not necessarily related to external sources of ranking like CiteSeer or Google.

How Ranks Change When Bob Sends Something. After receiving via email the context existing on Bob's desktop, as we described in Section 2, Alice's ranks change as presented in Table 3. By comparing the values in the two tables, we can see that some of the rankings increase, because existing resources are referenced by the newer ones. For example, the rank of the "ObjectRank" paper, labeled **B**, increases from 0.295982 to 0.301975 since it is now referenced by the paper labeled **A**. As a consequence, all the rankings for the resources which have an incoming link from **B** will increase. This process of rank propagation is an iterative one, according to the links in the data graph, and continues until the rank difference between two iterations is less than a certain threshold. Alice receives not only context from Bob, but also resources, so that she will also have rank values for these new resources.

The context which is received from other members of the interest group is used for building the user's own context, which means that it is also taken into account when creating the adjacency matrix *A*. In order to include the rankings of other users into the computation of the user's own ranking, we work on the vector *e*, which models the random jump. So, if a resource is highly ranked according to the received rankings and the user wants to take this into account, she will have to assign a higher value for the corresponding element in the vector which simulates the random jump.

Of course, even if two users exchange all of their context metadata, they still will not have the same rankings, as local usage information such as number of accesses etc., which influences rankings, always stays local and is not exchanged. Note that in our data graph, group members usually appear as instances of authors or as senders of emails [2], so we can use their rank as one possible indicator of their trustworthiness.

How Alice's Trust in Bob Influences the Rankings. Even inside an interest group, we have to take into account different reputations. If somebody, whom I trust and who is important for me, sends his recommendations, I want his suggestions to be higher ranked than the ones received from a more untrusted person. These different reputations can be represented by influencing the dampening factor. The higher the trustworthiness

Table 2. Alice's personal rank values

Resource	Rank
B	0.295982
G	0.260644
J	0.260644
S	0.248796
R	0.248796
T	0.244728
P	0.185268
SIGIR	0.181705
WWW	0.176995
Balmin	0.175207
Brin	0.172292
Kleinberg	0.172292
Riedl	0.171261
Vogt	0.171261
Cai	0.170888
Q	0.170745
VLDB	0.162604
W. Y. Ma	0.159406

Table 3. Alice's personal ranking values after receiving context information from Bob

Resource	Rank	Resource	Rank
B	0.301975	P	0.181859
E	0.297845	K	0.178391
G	0.287298	Bob	0.175816
I	0.253764	Teevan	0.175459
WWW	0.245065	Brin	0.174496
S	0.244723	Widom	0.171624
R	0.244723	Riedl	0.170860
N	0.242694	Vogt	0.170860
M	0.242694	Rocha	0.170683
L	0.242694	Quan	0.170683
O	0.242694	Guha	0.170683
F	0.242280	Rose	0.170683
T	0.240779	Cai	0.170521
J	0.232862	Q	0.167610
C	0.228117	SIGCHI	0.162730
A	0.210174	INTERACT	0.160311
H	0.200576	W. Y. Ma	0.159224
D	0.191779	Balmin	0.158563
SIGIR	0.189280	VLDB	0.154282
Dumais	0.186942	ESWC	0.152771
Kleinberg	0.186882	DELOS	0.152553

of someone in my interest group, who sends me her own context and rankings, the higher should be the probability to reach the resources in that set.

In our example, we considered there is only one user Alice exchanges context with. In the previous table, Table 3, the rankings are computed as if Alice is fully trusting Bob, so that she does not make any difference between the resources she already has and the ones she receives from him. This translates into a vector e having all elements 1. If Alice doesn't trust Bob 100%, she will have to bias the PageRank on her resources, that is assign values less than 1 to the elements in the e vector corresponding to the resources coming from Bob. This means that the probability of reaching the resources she receives from Bob through a random jump is less than the probability of jumping to one of her own resources. In our experiments we computed Alice's rankings for different levels of trust she has for Bob, and the results are presented in Table 4. A detailed study about the influence of different trust distributions upon the ranking of resources is presented in [3].

As we would expect, the rankings decrease, as trust decreases. Originally highly ranked resources still have a high rank and for example paper **B** in the case of a trust biasing rank computation of 90% or 70% even acquires a higher rank than before the resource exchange takes place. That is because for these resources Alice already has her own ratings and they will increase due to the fact that they are referenced by some of the received resources. For the newly received resources, (Teevan, Dumais, etc.) for a trust level of 1%, the rankings are quite small compared to the rankings of the originally

Table 4. Alice's ranking values after receiving context information from Bob for different trust levels

Resource Label	PageRank					
	90% Trust	70% Trust	50% Trust	30% Trust	10% Trust	1% Trust
B	0.300138	0.296501	0.292864	0.289226	0.285589	0.283952
G	0.286867	0.286448	0.286029	0.285610	0.285191	0.285003
J	0.232428	0.231741	0.231054	0.230366	0.229679	0.229370
S	0.244652	0.244613	0.244574	0.244534	0.244495	0.244477
R	0.244652	0.244613	0.244574	0.244534	0.244495	0.244477
T	0.240721	0.240683	0.240644	0.240606	0.240567	0.240550
P	0.181833	0.181793	0.181754	0.181714	0.181674	0.181656
SIGIR	0.188384	0.186783	0.185181	0.183579	0.181978	0.181257
WWW	0.238370	0.225374	0.212379	0.199383	0.186388	0.180540
Balmin	0.158504	0.158401	0.158298	0.158195	0.158092	0.158046
Brin	0.174403	0.174367	0.174332	0.174296	0.174261	0.174245
Kleinberg	0.185509	0.182843	0.180177	0.177511	0.174845	0.173645
Riedl	0.170800	0.170796	0.170793	0.170790	0.170786	0.170785
Vogt	0.170800	0.170796	0.170793	0.170790	0.170786	0.170785
Cai	0.170465	0.170461	0.170458	0.170455	0.170452	0.170450
Q	0.167588	0.167551	0.167514	0.167478	0.167441	0.167425
VLDB	0.154252	0.154201	0.154149	0.154098	0.154046	0.154023
W. Y. Ma	0.159220	0.159222	0.159218	0.159216	0.159214	0.159213

existing ones. The probability of jumping to one of these resources is 0.01, in contrast to the probability of 1.0 of executing a random jump to the ones Alice already has on her desktop.

5 Related Work

[4] presents a class of model-based recommendation algorithms for creating a top-N list of recommendations. In their approach, they first determine the similarities between the various items and then use them to identify the set of items to be recommended. [4] also addresses the key steps of this class of algorithms: which are the methods used to compute the similarity between items and which are the methods used to combine these similarities, in order to compute the similarity between a basket of items and a candidate recommender item. Opposed to this, in our approach the recommended items are based on user preferences and explicit context information.

Tapestry [5] is a recommender system which, in a sense, is similar to our approach. Tapestry is an e-mail filtering system, designed to filter e-mails received from mailing lists and newsgroup postings. Each user can write a comment / annotation about each email message and share these annotations with a group of users. A user can then filter these email messages by writing queries on these annotations. Though Tapestry allows individual users to benefit from annotations made by other users, the system requires an individual user to write complicated queries. We extend the idea in Tapestry by annotating not only emails but other resources on the user's desktop. In addition, exchange of annotations is handled (semi-) automatically.

The first system that generated automated recommendations was the GroupLens system [8]. The system, like in our case, provides users with personalized recommendations by identifying a neighborhood of similar users and recommending the articles that this group of users finds interesting.

The most interesting work for recommendation infrastructures, which does not require a central recommender server is PocketLens, [7]. The paper discusses on how to preserve privacy in such an infrastructure. In contrast to our work, they do not exploit semantic connections between items, such as we have for citation relationships.

Compared to the usual recommender systems, including the commercial ones such as Amazon.com, which usually suggest single items, we have the potential to make semantically rich suggestions that are represented as parts of a semantic network which we exchange. Additionally we also provide to the user information about other users' rankings. While most recommender systems define groups by relying on the overlap among preferred items, we rely on an explicit group membership denotation based on FOAF metadata.

6 Conclusions and Future Work

FOAF is a nice vocabulary to describe social networks, but most of the current applications are centered around describing social networks and not how to use them. This paper explores how to build upon FOAF and rich semantic web metadata to exchange and recommend context information and resources in a social network. These contextual metadata are described by appropriate annotation ontologies, and are exchanged within FOAF groups as specified by the group members. The exchange of metadata is done by means of additional attachments for each document exchanged via email, extending email exchange from pure document exchange to an exchange of both document and relevant context information. We presented how the computation of ranking is accomplished and how this computation is influenced by the context exchange as well as by the reputation of persons involved in the exchange process.

There are quite a few interesting issues to be investigated in future work, including privacy and security issues. This is especially important if we exploit peer-to-peer infrastructures instead of email attachments to implement a knowledge sharing infrastructure as described in this paper. It is also worthy to note that the ranking we compute for different resources can be compared to the ratings which are used in recommender systems. We can therefore not only share resources which are semantically connected to the ones we are exchanging, but also resources which are ranked / rated highly by peers in our community. An additional interesting aspect is to explore dynamic social networks, where groups are not statically defined from the beginning but dynamically based on the exchange of context metadata. In this case users can initially choose which pieces of metadata information they want to append to a document for certain recipients, and common exchange patterns then determine common interest groups and allow automatic exchange of metadata based on these previous interactions.

Acknowledgements

We want to thank Andrei Damian for his contribution to the publication metadata generator and Paul Chirita for many good discussions on topics related to this paper.

References

1. A. Balmin, V. Hristidis, and Y. Papakonstantinou. Objectrank: Authority-based keyword search in databases. In *VLDB*, Toronto, September 2004.
2. P. Chirita, R. Gavriloaie, S. Ghita, W. Nejdl, and R. Paiu. Activity based metadata for semantic desktop search. In *Proceedings of the 2nd European Semantic Web Conference*, Crete, May 2005.
3. A. Damian, W. Nejdl, and R. Paiu. Peer-sensitive objectrank-valuing contextual information in social networks. In *L3S Technical Report*, 2005.
4. M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. In *ACM Transactions on Information Systems*, January 2004.
5. D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collabarative filtering to weave an information tapestry. In *ACM Press*, December 1992.
6. Teevan J., Alvarado C., Ackerman M. S., and Karger D. R. The perfect search engine is not enough: A study of orienteering behavior in directed search. In *CHI*, Vienna, April 2004.
7. B. N. Miller, J. A. Konstan, and J. Riedl. Pocketlens: Toward a personal recommender system. *ACM Trans. Inf. Syst.*, 22(3):437–476, 2004.
8. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM Press, 1994.
9. The foaf project. http://www.foaf-project.org/.