BuddyFinder-CORDER: Leveraging Social Networks for Matchmaking by Opportunistic Discovery

Technical Report kmi-05-13
September 2005

Jianhan Zhu, Marc Eisenstadt, Alexandre Gonçalves, Chris Denham

To appear in Proc. of International Semantic Web Conference (ISWC2005) Workshop on Semantic Network Analysis, November 7, 2005, Galway, Ireland.

BuddyFinder-CORDER: Leveraging Social Networks for Matchmaking by Opportunistic Discovery

Jianhan Zhu[a], Marc Eisenstadt[a], Alexandre Gonçalves[b], Chris Denham[a]

[a]Knowledge Media Institute, The Open University
{j.zhu, m.eisenstadt, c.m.denham}@open.ac.uk
[b]Stela Group, Federal University of Santa Catarina
a.l.goncalves@stela.org.br

Abstract. Online social networking tools are extremely popular, but can miss potential discoveries latent in the social 'fabric'. Matchmaking services can do naive profile matching with old database technology, and modern ontological markup, though powerful, can be onerous at data-input time. In this paper, we present a system called BuddyFinder-CORDER which can automatically produce a ranked list of buddies to match a user's search requirements specified in a term-based query, even in the absence of stored user-profiles. We integrate an online social networking search tool called BuddyFinder with a text mining method called CORDER to rank a list of online users based on 'inferred profiles' of these users in the form of scavenged Web pages.

1 Introduction and Motivation

Online social networking tools and services, in the form of friendship networks, instant messaging and chatting tools, dating services and business partner tools are extremely popular on the Web today – an extensive and evolving survey/taxonomy of Social Networking Services is provided online in [16]. Such tools and services have evolved from early-adopter 'leisure' tools to mission-critical business collaboration tools, and have attracted increasing attention from the research community [17] [18] [6] [7]. Today's online social networking tools typically involve a large number of users who coalesce into a community of mixed backgrounds and preferences, exhibiting varying degrees of overlapping interests and social cohesion. An online user typically has a number of contacts or buddies in his/her interest groups, with the amount of overlap (shared interests) dropping off dramatically as degrees of separation increase: groups may include work colleagues, family members, friends, conference acquaintances, friends-of-friends, recommended contacts, deliberately sought-out contacts, and of course random or even unwanted contacts. Current social networking tools allow users to manage additions to and deletions from their buddy lists manually, although most allow import from standard productivity-tool address books and preferences to block unwanted contacts.

In this era of powerful search engines (which are getting even better as the semantic web matures), consider the problem of seeking mission-critical or immediate advice characterized by the question ‘Who is available who can help me deal with an urgent problem *now*?’ The Knowledge Management mantra of ‘the right knowledge in the right place at the right time’ rings somewhat hollow if you need to find key people quickly but cannot! Mixed hi-tech and lo-tech solutions (e.g. using Google or a social networking service as a first pass, then emailing or phoning around) may miss out on key availability information, which is one of the great strengths of instant messaging (IM) tools. Indeed, in this context it is not the messaging and chat capabilities of IM that pay the biggest dividends, but rather the *presence* information that (when correct at least!) shows who is available [19]. Modern social networking services such as Tribe [20], and indeed most discussion forums, typically include a ‘presence indicator icon’ so users can see at a glance who is available. Other tools, such as our own BuddySpace [21], also include the option of overlaying presence indicators on top of custom maps to deal with those cases where location is either important or simply ‘feels good’ to the user.

How, then, can we address the problem of finding the right person *now*? This problem falls at the overlap of two seemingly contradictory niches:

- *Good availability, but no need for search:* this is the ‘IM niche’ which is great for seeing who is available now, but since a user’s IM buddy list is populated by people they already know, it may seem rather artificial to have to search through such people for matches
- *Web-centric search, but poor availability info*: the generic problem of searching the web for ‘the right person’ is well known, and indeed addressed by other papers at this workshop, but generally we cannot count on availability information being provided

We address this contradiction by noting a key exception to the assumption that ‘a user’s IM buddy list is populated by people they already know’. In particular, enterprise-wide IM and any service that provides automatic buddy-list population (‘automatic roster or contact list generation’), has the potential for auto-enrolling users into large groups, such as student cohort groups and large just-in-time project teams, where a hierarchically-grouped buddy lists can grow just beyond a ‘familiarity horizon’: in other words, a user does *not* know a lot about everyone on the buddy list, yet it contains, within acceptable privacy limits, availability information for those who may have the knowledge to help solve a user’s problem! A typical case is our own ELeGI project [22], a 23-partner European consortium with several hundred individuals involved. All are automatically subscribed to the IM roster, yet not everyone generally knows much about everyone else. Geographical information is also provided on our user maps, which can be overlaid with presence information, as shown in Fig. 1.

We believe that such usage scenarios will be increasingly typical of future knowledge workers and e-learners. In these cases, it is clear that simply listing online users in alphabetical order [1] [3] and randomly matching [4] cannot help users find their those with the right knowledge accurately and efficiently. Of course, some tools use registration information in profiles of users for search and can provide accurate search results given that these data are accurate and complete. This is the classic approach of the online dating services, and can be very effective in a live instant messaging context too. In Fig. 2, we see how MSN member search [2] enables users to search other users by their MSN nickname, last name, first name, and gender etc.

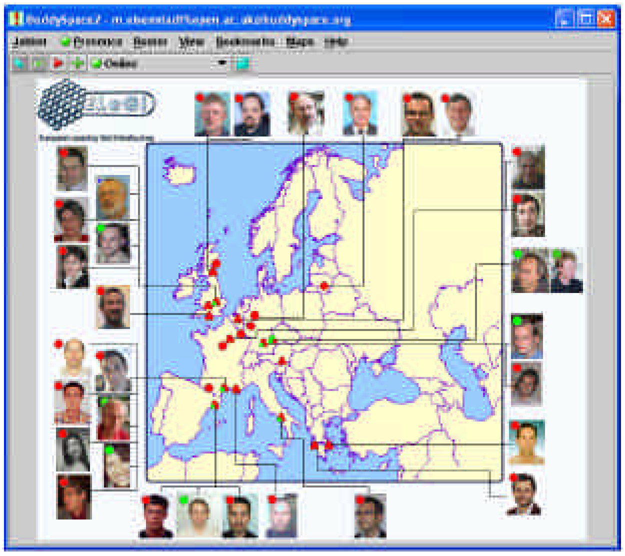
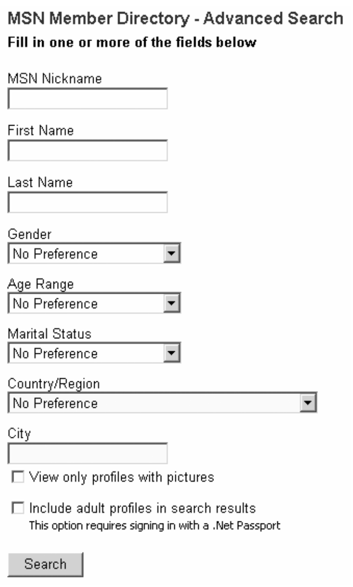


Fig. 1. BuddySpace showing presence (colored dots) and location information.



MSN Member Directory - Advanced Search

Fill in one or more of the fields below

MSN Nickname	
First Name	
Last Name	
Gender	No Preference
Age Range	No Preference
Marital Status	No Preference
Country/Region	No Preference
City	

☐ View only profiles with pictures

☐ Include adult profiles in search results
This option requires signing in with a .Net Passport

Search

Fig. 2. MSN advanced search.

A more powerful approach in principle is to ask online users to write FOAF (Friend of a friend) files to describe themselves [23]. We could then perform buddy search using structured information in these FOAF files. However, these profile data or

FOAF files generally need to be explicitly and voluntarily provided by the users. Overall, we note four shortcomings of the database-profile and FOAF-profile approaches as follows.

- First, credibility of the data is at danger, i.e., users may not have provided true information about themselves or may not provide complete information about themselves, and thus the approaches is susceptible to abuse.
- Second, asking users manually to provide the data creates unacceptable overheads.
- Third, completeness of the data is at risk, i.e., users may not update the data to reflect their ongoing activities.
- Fourth, the search is limited to the information specified in the profile or FOAF file, thus the search cannot extend to a wider range of searches.

Another approach is to use a domain ontology for community of practice (COP) discovery within an organization. ONTOCOPI (Ontology-based Community of Practice Identifier) [18] attempts to uncover COPs by applying a set of ontology-based network analysis techniques that examine the connectivity of instances in the knowledge base with respect to the type, density, and weight of these connections. These COPs can be used for buddy search within an organization. Although effective, this approach also suffers from the following two shortcomings.

- First, it is generally hard to maintain the ontology to reflect the reality on the ground, i.e., part of knowledge in the ontology will become out of date quickly and new knowledge in reality cannot be put into the ontology in time.
- Second, the search is limited to knowledge in the ontology.

In contrast to these approaches, consider that there is significant information about many Web users in their personal homepages and blogs, and (even when those are missing, incomplete, or out of date) their friends or colleagues' homepages, and Web sites of their work place – all potentially relevant to their personal and/or career life. The information can provide a fertile ground for a social networking tool to search for buddies who match a user's preferences, *even in the absence of a user's specific profile information*, and *even in the absence of a user's own web pages*! Our belief is that we can leverage social networks ('whom you know') for opportunistic discovery that can potentially deliver the right knowledge in the right place at the right time, by deploying a mixture of buddy list, presence, and text mining methods. Such a hybrid approach can overcome the shortcomings of the current approaches.

- First, since Web pages about Web users are from various sources, these sources are controlled by different authorities and thus the search is less susceptible to abuse. We can judge the credibility of the data source for including the Web pages in search process.
- Second, there are fewer overheads in collecting the data. Users do not need to fill a form or maintain a FOAF file, and only need to make no or very little effort, e.g., list URLs of pages which can serve as profiles of themselves.
- Third, we can more easily get a more complete profile of a user and are not limited to information in a registration form or a FOAF file.
- Fourth, there are a more variety of searches that can be issued and the search is not limited to information in a registration form or a FOAF file.

In order to use Web pages pertinent to social networking users for searching them, we can directly use current search engines such as content-based search engines [5] and Google which uses both content information and PageRanks [8] of Web pages. PageRanks of Web pages are based on link structures of the Web. For example, given a list of users and their names, each of them has a profile consisting of a list of Web pages. In content-based buddy search, an online user can specify a query as his/her preferences or matching requirements. The query consists of a list of terms and Boolean relations between these terms. The query is used to search for other users as buddies whose profiles contain Web pages matching the query. These buddies are seen as matching the preferences or requirements of the user. For example, a query "Java AND C++" will return users whose profiles contain Web pages matching both "Java" and "C++" keywords. These users are assumed to have interests in "Java" and "C++".

However, a typical query such as "Java AND C++" will return a larger number of matching buddies, who are of different levels of relevance to the query. By giving a relevance score to each of these buddies, we can present a user with a list of buddies ranked by their relevance scores. In the ranked list of buddies, buddies more relevant to the query are on the top and buddies who are probably false positives are at the bottom and can be ignored. We observe that a buddy is more relevant to a term in a query when the buddy's name has more co-occurrences with the term and/or occur close to the term in co-occurred Web pages. In content-based search, we can rank buddies by only taking into account co-occurrences between each buddy and terms in the query.

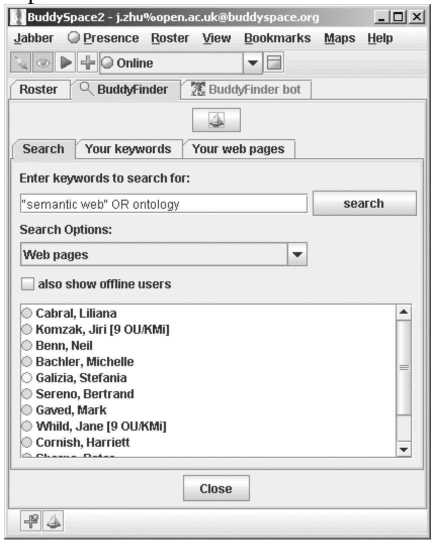
In our previous work, we have proposed a text mining method called CORDER [9], which unearths relations between named entities[1] from Web pages of a community. CORDER is based on communities of practice [10], where a group of people are collaborating together on shared tasks, rather than their institutional division. The documents that a group of people produces mirror what people do and who they work with. CORDER automatically discovers relations between people in the community from these documents. Given a named entity, both co-occurrences and distances between the named entity and other named entities in these documents are taken into account in ranking these named entities. Our experimental results showed that CORDER produced better rankings of named entities than the co-occurrence based ranking method.

In this paper, we apply the CORDER method to buddy search. There are mainly two extensions of the CORDER method in the buddy search scenario. First, in previous CORDER method, we consider relations between named entities. While in buddy search, we consider relations between a named entity and terms in a query. While our previous experiments show that CORDER can identify significant relations between named entities. In buddy search, we need to find out whether CORDER can also identify significant relations between named entities and terms. Second, we need to extend CORDER method to take into account Boolean relations between terms in a query.

Given a query, the ranking of a user is given by taking into account the co-occurrences between the user and terms in the query, the distances between the user and terms in the query, and Boolean relations between the terms. In an online group

[1] Named entities are proper names of various types, e.g., "John Smith" is a "Person" and "Open University" is an "Organization".

consisting of a number of buddies, Web pages in one buddy's profile often talk about not only him/herself but also his/her friends, i.e., other buddies, in the same group. One buddy may belong to various groups on the Web. In buddy ranking, given a buddy, Web pages from both his/her profile and profiles of other buddies in the same groups as his/hers are taken into account in ranking him/her. We integrate CORDER with an online social networking tool called BuddyFinder. BuddyFinder is part of BuddySpace (http://buddyspace.sourceforge.net/) [11], an online instant messaging tool. A number of users are subscribed to BuddyFinder and each of them *has an optional* profile containing a list of Web pages from his/her homepage, blog, or other sources describing his/her personal or career life, and even in the absence of that profile we can make a best guess based on the user's identity and domain (part of their login specification) to find such pages. A BuddyFinder user can use term-based query to search subscribed users and get a list of buddies ranked by CORDER. In Fig. 3, a BuddyFinder user input a query "semantic web" OR ontology and get a list of users ranked by CORDER. He/she can choose to interact with these users by chatting, emails, and viewing their profiles etc.



Enter keywords to search for: "semantic web" OR ontology

Search Options: Web pages

also show offline users

Cabral, Liliana
Komzak, Jiri [9 OU/KMi]
Benn, Neil
Bachler, Michelle
Galizia, Stefania
Sereno, Bertrand
Gaved, Mark
Whild, Jane [9 OU/KMi]
Cornish, Harriett

Fig. 3. BuddyFinder output for a search on "semantic web" OR ontology.

The rest of the paper is organized as follows. In Section 2, we give an overview of the CORDER method. In section 3, we present the BuddyFinder-CORDER system. In Section 4, we conclude and propose future work.

2 CORDER: A Community Relation Discovery Method

Typical questions for knowledge managers are what do your employees know about, which of your customers have they contacts with, and who works well together in teams? However, the knowledge represented in organizational ontologies and other resources is often static, reflecting management's design of what should happen in the organization and not necessarily the real situation on the ground. The real situation is often characterized better by communities of practice [10], the groupings of people who collaborate together on shared tasks, rather than institutional divisions.

In our buddy search scenario, online users as a community have web pages in their profiles describing themselves and each other. These web pages can often more truthfully describe themselves than their registration information or FOAF files, which are often static and not up to date.

We argue that Web pages as profiles of online users mirror what these users do and who they share interests with. CORDER discovers relations from the Web pages of a community of online users. Previous CORDER method is based on co-occurrences of named entities (NEs) and the distances between them. A named entity recognizer, ESpotter [14], is used to recognize people, projects, organizations and research areas from Web pages. BuddyFinder extends the CORDER method to relations between buddy names as NEs and terms in a Boolean query.

In the CORDER method, given a target as an NE, we rank a number of co-occurring NEs as objects. In buddy search scenario, the target is a term in a Boolean query and the term co-occurs with a number of buddy names as objects. We assume that objects that are closely related to the target tend to appear more often with the target and closer to the target in Web pages. Given the target, we calculate a relation strength for each co-occurring object based on its co-occurrences and distances from the target. The co-occurring objects are ranked by their relation strengths. Thus objects which have strong relations with the target can be identified. The relation strength between a target and an object takes into account three aspects as follows.

Co-occurrence. A target and an object are considered to co-occur if they appear in the same Web page. Generally, if the object is closely related to the target, they tend to co-occur more often.

Distance. A target and an object which are closely related tend to occur close to each other.

Frequency. A target or object is considered to be more important if it has more occurrences in a Web page.

Given a target, the higher the relation strength of an object, the closer they are related to each other. We set a relation strength threshold, so that only significant relations having relation strengths above the threshold are selected. Relations having relation strengths below the threshold are considered to result from noise in our data and are ignored. In our study we set the threshold as the value at which a target and an object co-occur with only one occurrence each in only one Web page, and their distance in the Web page is a certain value *D*. Given the target, objects with their relation strengths above this threshold are considered to be related.

Evaluation of the CORDER method on a departmental Website indicates that the method can find NEs closely related to a target and provide accurate rankings. CORDER’s running time increases linearly with the size and number of web pages it examines. CORDER can incrementally evaluate existing relations and discover new relations by taking into account new Web pages. Thus CORDER can scale well to a large dataset.

3 BuddyFinder-CORDER

In Section 3.1, we present the architecture of the BuddyFinder-CORDER system. In Section 3.2, we present the extended version of the CORDER method for buddy ranking. In Section 3.3, we present our initial user evaluation.

3.1 System Architecture

BuddyFinder-CORDER relies on an instant messaging platform Jabber [12] for exchanging information. In Fig. 4, Jabber [12] uses an XML based protocol called XMPP [12] for all transactions between Jabber users and the Jabber server in **B**. When a Jabber user in **A** issues a term-based query to search for buddies, the query is sent from the user's Jabber client using chat messages following the XMPP protocol. The BuddyFinder query is routed by the server to the BuddyFinder Chatbot in **C**, which interacts with various modules to get a ranked list of buddies. The BuddyFinder Chatbot is a standard Jabber client, an architecture which allows a lot of flexibility in terms of where it is hosted and allows BuddyFinder users to interact with it using any Jabber based instant messaging client software. BuddyFinder Chatbot will reply with the results of the query following the XMPP protocol.

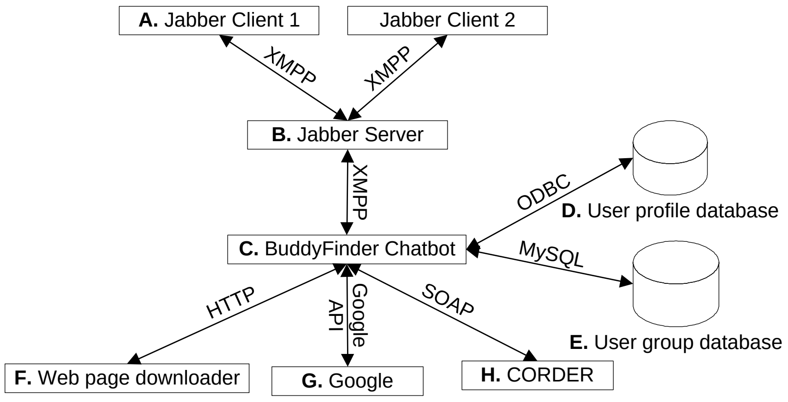


Fig. 4. BuddyFinder-CORDER Architecture.

The user profile database in **D** stores each user's profile consisting of a list of Web page URLs. BuddyFinder Chatbot accesses user profile information via ODBC connection. The initial list of URLs for a user is generated automatically using a Web search component via the Google API. To give an easy start for a user to specify these URLs, the Web search component makes an "intelligent guess" in finding Web pages relevant to him/her. The intelligent guess is based on the domain shown in his/her email address and his/her name, which are stored in the user group database in **E**. For example, if one user's email address is m.eisenstadt@open.ac.uk and name is "Marc

Eisenstadt", we guess the domain of the user is open.ac.uk. We send the query "Marc Eisenstadt site:open.ac.uk" to Google and use URLs of the top 10 Web page as the default profile for the user.

To improve search performance, web pages in users' profiles are downloaded in **F** and cached for search. Each user is subscribed to various user groups, e.g., KMi user group and Open University user group. Given a user, a list of buddies who belong to his/her user groups is generated by issuing SQL queries on the profile database in **E**. When a user sends a query to search, BuddyFinder performs the search on the profiles of the list of buddies. Thus different users can get different search results even when they sent the same query.

When a user searches BuddyFinder, he/she may get a large number of buddies whose profiles all match the query. Some buddies who are not closely related to the query may also be returned simply because their names co-occur with terms in the query on Web pages. We need a ranking algorithm which can rank these buddies in terms of their relevance to the query. A good ranking algorithm should be able to both identify buddies highly relevant to the query by putting them at the top of the list and putting them in the correct order. Ideally, the ranking algorithm should put the buddies in the same order that the user will put them. In Fig. 4, given a search query, BuddyFinder uses the CORDER algorithm, which takes into account co-occurrences, distances in co-occurred Web pages, and relations in the query, to rank search results.

The buddy search process in Fig. 4 is as follows. A user in **A** specifies a search query in a command line, e.g., "find "semantic web" OR ontology". The query is sent to BuddyFinder Chatbot in **C** and be interpreted for a list of terms and Boolean relations between them. The BuddyFinder Chatbot goes to the user group database in **E** to find a list of buddies who belong to the same groups as the current user. BuddyFinder Chatbot goes to user profile database in **D** to get profiles of these buddies.

Each buddy's profile consists of a list of Web pages. Given a buddy, BuddyFinder Chatbot uses a set of Web pages for ranking him/her against the query. First, Web pages from the buddy's profile are included in ranking. Second, since Web pages from profiles of other buddies who are in the same groups as his/hers, such as the buddy's colleagues, contain information relevant to him/her, e.g., co-authoring same papers and members of same projects, these Web pages are also taken into account for ranking him/her.

Given the buddy, for each term in the query, BuddyFinder Chatbot processes the set of Web pages for co-occurrences between the buddy's name and the term and offsets of the buddy's name and the term in co-occurred Web pages. CORDER in **H** is implemented as a Web service and BuddyFinder Chatbot calls the CORDER Web service by sending co-occurrence, offset and Boolean relations information to CORDER as a SOAP message. CORDER calculates a ranking of the buddy against the query based on the algorithm presented in the next section and sends back the ranking to BuddyFinder Chatbot as a SOAP message. BuddyFinder presents the buddies in the order of their rankings to the user.

3.2 Buddy Ranking Algorithm

We extend the CORDER method in mainly two aspects to the buddy search scenario. First, extend relations between named entities to relations between named entities and terms. Second, extend CORDER method by taking into account Boolean relations between terms.

An online user's preferences for buddies are specified in a term-based query, which consists of a list of terms and the Boolean relations between these terms. These terms can either be single words or phrases. We use the CORDER method to calculate the ranking of a buddy as a relation strength between the buddy and the search query. The relation strength is based on co-occurrences of the buddy's name and each term in Web pages, the distances between the buddy's name and each term in Web pages, and Boolean relations between these terms. A list of buddies is ranked by their relation strengths, and buddies who are more relevant to the query are at the top of the list. Given a buddy, Web pages from his/her profile and profiles of other buddies in the same groups as his/hers are used for ranking him/her. The relation strength between a buddy and the query takes into account four aspects as follows.

Co-occurrence of buddy and a term. A buddy and a term are considered to co-occur if they appear in the same Web page. Generally, if a buddy is closely related to a term, they tend to co-occur more often. For a buddy, *B* and a term *K*, we use Resnik [13]'s method to compute a relative frequency of co-occurrences of *B* and *K* as $\hat{p}(B,K) = \frac{Num(B,K)}{N}$, where *Num(B,K)* is the number of co-occurring Web pages for *B* and *K*, and *N* is the total number of Web pages.

Distance between buddy's name and a term. A buddy and a term which are closely related tend to occur close to each other. If a buddy and a term, *B* and *K*, both occur only once in a Web page, the distance between *B* and *K* is the difference between the offsets of *B* and *K*. If *B* occurs once and *K* occurs multiple times in the Web page, the distance of *B* from *K* is the difference between the offset of *B* and the offset of the closest occurrence of *K*. When both *B* and *K* occur multiple times in the Web page, we average the distance from each occurrence of *B* to *K* and define the logarithm distance between *B* and *K* in the *i*th Web page as $\overline{d_i}(B,K) = \frac{\sum_j (1 + \log_2(\min(B_j, K)))}{Freq_i(B)}$, where $Freq_i(B)$ is the number of occurrences of *B* in the *i*th Web page and $\min(B_j, K)$ is the distance between the *j*th occurrence of *B*, B_j, and *K*.

Frequency of buddy's name. A buddy is considered to be more important if his/her name has more occurrences in a Web page. Consequently, a more important buddy on a Web page tends to have strong relations with other terms which also occur on the Web page.

Boolean relation between terms. Terms are connected by AND, OR, NOT Boolean connectors. There Boolean connectors are taken into account in relation strength calculation. For two terms *T1* and *T2*, we consider two kinds of relations between them as AND and OR. The AND operator is used to specify that both *T1* and *T2* must evaluate to TRUE for "*T1* AND *T2*" to evaluate to TRUE. Given a buddy, *B*, we use a set of Web pages to rank him/her. In order for both *T1* and *T2* to evaluate to TRUE, *B* needs to co-occur with *T1* and *T2*, respectively. For example, if *B* co-occurs with both *T1* and *T2* on a page, "*T1* AND *T2*" evaluates to TRUE. If *B* co-occurs with *T1* on one page and co-occurs with *T2* on another page, "*T1* AND *T2*" still evaluates to TRUE. If the relation strengths between *T1*, *T2* and a buddy are $R(B,T1)$ and $R(B,T2)$, respectively, we define the relation strength between "*T1* AND *T2*" and the buddy as $R(B,T1\ \ AND\ \ T2) = R(B,T1)\times R(B,T2)$. The OR operator is used to specify that either *T1* or *T2* must evaluate to TRUE for "*T1* OR *T2*" to evaluate to TRUE. In order for either *T1* or *T2* to evaluate to TRUE, *B* needs to co-occur with either *T1* or *T2*. We define the relation strength between "*T1* OR *T2*" and the buddy as $R(B,T1\ \ OR\ \ T2) = R(B,T1) + R(B,T2)$. For one term *T1*, the NOT operator is used to specify that *T1* must evaluate to FALSE for "NOT *T1*" to evaluate to TRUE. We define the relation strength between "NOT *T1*" and the buddy as $R(B, NOT\ \ T1)$ =0 if $R(B,T1)$ > 0 and =1 if $R(B,T1)$ = 0.

Relation strength between buddy and query. Given a buddy, *B*, and a query, *Q*, we get a list of terms {T_k} from *Q* and their Boolean relations. We calculate the relation strength between *B* and *Q* by taking into account co-occurrences, distance and frequency between *B* and each term in {T_k}, and Boolean relations between terms in {T_k}. The relation strength, $R(B,T_k)$, between *B* and T_k is defined in Equation 1.

$$R(B,T_k) = \hat{p}(B,T_k)\times\sum_i\left(\frac{f(Freq_i(B))\times f(Freq_i(T_k))}{\bar{d}_i(B,T_k)}\right) \tag{1}$$

where $f(Freq_i(B)) = 1+\log_2(Freq_i(B))$, $f(Freq_i(T_k)) = 1+\log_2(Freq_i(T_k))$, $Freq_i(B)$ and $Freq_i(T_k)$ are the numbers of occurrences of *B* and T_k in the *i*th Web page respectively.

We get the relation strength, $R(B,Q)$, between *B* and *Q*, by combining $R(B,T_k)$ using the Boolean relations between them. For example, for a query *Q* = T1 AND T2 AND (NOT T3), $R(B,Q) = R(B,T1)\times R(B,T2)\times R(B, NOT\ \ T1)$.

3.3 Initial User Evaluation

Currently we have 234 BuddyFinder users, each of them can have a profile consisting of up to 10 Web pages from their homepages, departmental Web pages, and blogs etc. We have asked three BuddyFinder users from outside our department to independently evaluate our system. Each of them was asked to compose 10 queries and used them to search in BuddyFinder. We compare CORDER with co-occurrence based ranking method, in which we calculate the relation strength between a buddy and a query term as the number of co-occurrences of the two in Web pages.

For each query, the user got two ranked lists of buddies and he/she was not told which one was created by CORDER or the co-occurrence method. The user was asked to judge the relevance of each of the top 5 buddies in the two lists to the search query by giving a score from -2 to 2, where -2 is highly irrelevant, -1 is irrelevant, 0 is not sure, 1 is relevant, and 2 is highly relevant. The user can judge the relevance by chatting with the buddy and viewing his/her profile etc.

For the first user, we got a total score of 89 and 78 (out of 100) on the 10 queries for the CORDER based rankings and co-occurrence based rankings, respectively. For the second user, we got a total score of 86 and 71 (out of 100) on the 10 queries for the CORDER based rankings and co-occurrence based rankings, respectively. For the third user, we got a total score of 85 and 78 (out of 100) on the 10 queries for the CORDER based rankings and co-occurrence based rankings, respectively.

The initial results show that CORDER produced good rankings for buddy search and provided better rankings than the co-occurrence based method. We are currently working on evaluating BuddyFinder-CORDER on a larger user group and the initial results are promising.

4 Conclusions and Future Work

We have shown that the BuddyFinder-CORDER system can help users' online collaboration by enabling them to search for buddies matching their interests specified in term-based queries. Since current instant messaging tools mostly rely on registration information for buddy search, they are susceptible to fraud, limited information, and out-of-date information. BuddyFinder-CORDER can enable more trustworthy, more versatile, and more up-to-date buddy search. Initial experiments show that BuddyFinder-CORDER can find buddies highly relevant to search queries and provide better rankings than a co-occurrence based method. BuddyFinder-CORDER's running time increases linearly with the size and number of Web pages it examines. Thus BuddyFinder-CORDER can scale well to a large dataset. The initial experiment is still preliminary. We are evaluating BuddyFinder-CORDER on a larger user group consisting of over 200 users.

CORDER's rankings are derived from data mined from a collection of documents. In this way it gives a wider view of the "world" of a domain than data from a single source, such as registration information provided by users. Our future work is twofolded. First, BuddyFinder-CORDER uses a text mining method to deal with mainly unstructured information in Web pages. If we can get structured information about online users in the form of registration data or FOAF files, we can improve the BuddyFinder-CORDER method by taking into account both unstructured and structured data. For organizational use, we are considering using CORDER text mining methods to keep an organizational ontology up to date with knowledge embedded in documents produced by the organization. The ontology could be used by systems such as ONTOCOPI [18] for community of practice discovery and buddy search. Second, BuddyFinder-CORDER enables discovery of indirect relationships among buddies. In our scenario, it can be used to find out buddies who are not directly related to terms in a query but are interesting, e.g., a buddy A is highly relevant to a buddy B, B is highly relevant to a query, A is not directly relevant to the query. There is an indirect relationship between A and the query. BuddyFinder can suggest A to the searcher. We are experimenting with using the closest entities suggested by CORDER to improve the vector descriptions of documents for clustering. Our initial experiments suggest that this approach produces clusters which score as well as the widely used SOM method [16] on a total information gain measure of cluster quality. The execution time of the CORDER enhanced clustering method however increases linearly with the size and number of documents it examines so that it starts to outperform SOM on collections of more than 700 vectors.

Acknowledgements

Work on BuddySpace and BuddyFinder is supported by the European Community under the Information Society Technologies (IST) programme of the 6th Framework Programme for RTD – project ELeGI, contract IST-002205. This document does not represent the opinion of the European Community, and the European Community is not responsible for any use that might be made of data appearing therein.

Work on CORDER was partially supported by the Designing Adaptive Information Extraction from Text for Knowledge Management (Dot.Kom) project, Framework V, under grant IST-2001-34038 and the Advanced Knowledge Technologies (AKT) project. AKT is an Interdisciplinary Research Collaboration (IRC), which is sponsored by the UK Engineering and Physical Sciences Research Council under grant number GR/N15764/01. The AKT IRC comprises the Universities of Aberdeen, Edinburgh, Sheffield, Southampton and the Open University. This research was also partially supported by the Brazilian National Research Council (CNPq) with a doctoral scholarship held by Alexandre Gonçalves.

We are grateful to Jiri Komzak for the implementation of BuddySpace and significant input into BuddyFinder, Martin Dzbor for significant input into the concepts, design, and implementations underlying BuddySpace, and Victoria Uren and Enrico Motta for helpful discussion.

References

1. http://www.msn.com
2. http://members.msn.com
3. http://messenger.yahoo.com
4. http://www.icq.com
5. http://www.verity.com

6. Isaacs, E., Walendowski, A., Whittaker, S., Schiano, D. J., Kamm, C. A.: The character, functions, and styles of instant messaging in the workplace. In Proc. of ACM conference on Computer supported cooperative work (CSCW 2002), 11-20.
7. Setlock, L. D., Fussell, S. R., Neuwirth, C.: Taking it out of context: collaborating within and across cultures in face-to-face settings and via instant messaging. In Proc. of ACM conference on Computer supported cooperative work (CSCW 2004), 604-613.
8. Brin, S., Page, L.: The Anatomy of a Large-Scale Hypertextual Web Search Engine. Computer Networks 30(1-7): 107-117 (1998)
9. Zhu, J., Gonçalves, A. L., Uren, V. S., Motta, E., Pacheco, R.: Mining Web Data for Competency Management. In Proc. of 2005 IEEE/WIC/ACM International Conference on Web Intelligence 2005 (WI'2005), Compiegne University of Technology, France, September 19-22.
10. Wenger, E.: Communities of Practice, learning meaning and identity. Cambridge University Press (1998).
11. Eisenstadt, M., Komzak, J. and Dzbor, M.: Instant messaging + maps = powerful collaboration tools for distance learning. In Proc. of TelEduc03, May 19-21, 2003, Havana, Cuba.
12. Adams, D. J.: Programming Jabber Extending XML Messaging. O'Reilly (2001).
13. Resnik, P.: Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language. Journal of Artificial Intelligence Research (1999). 11:95-130.
14. Zhu, J., Uren, V. S., Motta, E.: ESpotter: Adaptive Named Entity Recognition for Web Browsing. In Proc. of 3rd Professional Knowledge Management Conference, Springer, LNAI (2005).
15. T. Kohonen, Self-Organizing Maps, Springer Series in Information Sciences, Vol. 30, 1995.
16. Meskill, J.: A Social Networking Services Meta-List: http://socialsoftware.weblogsinc.com/entry/9817137581524458/
17. Nardi, B.A., Whittaker, S., Isaacs, E., Creech, M., Johnson, J., and Hainsworth, J.: Integrating communication and information through ContactMap. *Communications of the ACM*, 45:4, 89-95 (2002).
18. Alani, H., S. Dasmahapatra, K. O'Hara, N. Shadbolt.: Identifying Communities of Practice through Ontology Network Analysis. IEEE Intelligent Systems 18(2): 18-25, (2003).
19. Chakraborty, R.: Presence: A Disruptive Technology, Presentation at JabberConf, Denver, (2001).
20. http://www.tribe.net
21. Vogiazou, I.T., Eisenstadt, M., Dzbor, M., and Komzak, J.: From Buddyspace to CitiTag: Large-scale Symbolic Presence for Community Building and Spontaneous Play. In Proc. of the ACM Symposium on Applied Computing, Santa Fe, New Mexico, March 13 -17, (2005).
22. Allison, C., Cerri, S.A., Gaeta, M., Ritrovato, P. and Salerno, S.: Human Learning as a Global Challenge: European Learning Grid Infrastructure. In Varis, T., Utsumi, T. and Klemm, W. (eds.), *Global Peace Through the Global University System*, RCVE, Tampere, 465-488, (2003), http://www.terena.nl/conferences/tnc2004/core_getfile.php?file_id=576
23. FOAF, http://www.foaf-project.org