Who Can Best Answer a Query in my Social Network ?

Fabien Duchateau [#1]

[#]*CWI, Science Park 123*
1098 XG Amsterdam, The Netherlands
[1]fabien@cwi.nl

***Abstract*— Social networks have facilitated communications and exchange between people and many people share their experiences, pictures or interests on the Web. Yet, due to the diversity of social networks, it can be difficult to find the correct person which can answer a given query. Similarly, advertising companies need to target specific users which are most willing to be interested by their products. This paper addresses the problem of finding one or more acquaintances in a social network that have an interest for a given query. To fulfill this goal, we explore three integration strategies based on structure, semantics, terminology and tag co-occurrence. Finally, we have analyzed experiments with hundreds of user profiles to show the benefits and shortcomings of our approach.**

I. Introduction

Recently, social networks have emerged from the Web to build user communities. Due to the rapid growth of these networks, many social applications have arisen to provide users with new functionalities: geolocation[1] enables them to drink free beers, FriendBinder[2] centralizes all acquaintances from various social networks in one place and LastFM[3] recommends music that users may like according to friends' musical tastes. Although uniformity in these networks could facilitate exchange between users, we advocate that the Web has been created with the philosophy of diversity. Besides, social networks may have been designed with a specific goal. For instance, LinkedIn[4] aims at gathering professionals while Flickr[5] is a community focusing on photo sharing. For these reasons, we believe that users should be able to choose and use the social networks that suit them at most.

In his talk, Contractor underlines the problem of choosing the correct person to whom one should communicate with [1]. This problem is close to the one known as *expert finding*, which aims at identifying the persons who have the relevant expertise in a given domain. Some works have already studied expert finding in specific social networks [2], [3], [4], [5], [4]. However, most of these works are specific for one social network, for instance *DBLP*. Thus, they are mainly based on the relationships between users (e.g., co-author). In [3], relations are discovered by browsing web pages and analyzing email exchanges. Contrary to [2], we assume that a user will not contact an unknown person, but (s)he relies on the acquaintances from his/her social network. Besides, we do not assume that a domain expert can be found in the social network: the user may not have any contacts which has the relevant expertise. Other works are related to profile integration and semantic enrichment. For instance, Li et al. cluster tags based on patterns of frequent co-occurrence tags to discover user communities with common interests [6]. However, this work does not rank acquaintances to detect the ones with an answer for a given query. Similarly, the Mypes approach evaluates entropy, i.e., the probability that a tag appears in a user profile [7]. But it mainly aims at enriching the profiles thanks to Wordnet categories. Furthermore, user profiles are aggregated by means of unknown "hand-crafted rules" involving many tag redundancies.

In this paper, we explore the possibility of finding in our social network an acquaintance who is able to answer a given query. In a context where a user may be part of several social networks, information is disseminated in different profiles. Thus, the previously described approaches cannot be applied here because of the strong heterogeneity of these profiles. Besides, the profiles of the same user need to be integrated first to detect redundant interests. We have developed three strategies to integrate profiles of the same user and find the one who can answer the query. These strategies are based on semantics, structure and co-occurrence. To show the benefits of our approach, experiments with real data have been performed. We foresee different applications for our work. The most obvious is related to expert finding, i.e., ranking acquaintances from our social network to discover the ones that have interests or answers for a query. Recommendations are also targeted by our work. Let us imagine that we have an organization that needs to promote a rock festival. This organization is represented on different social networks and its acquaintances are followers and supporters. By integrating the profiles of these acquaintances and querying them with "rock festival", this organization can detect acquaintances that may be interested by this kind of event.

The rest of this paper is organized as follows: Section II provides definitions. In Section III, we present the details of our approach for finding an expert in one's social network. Pre-

[1]http://www.foursquare.com
[2]http://friendbinder.com
[3]http://www.lastfm.com
[4]http://linkedin.com
[5]http://www.flickr.com/

liminary experiments, whose results are presented in Section IV, show the advantages and shortcomings of our approach. Finally, we conclude in Section V.

II. Preliminaries

This section presents definitions for the main notions used in the rest of this paper. A user can have a **profile** on different **social websites**. For sake of clarity, we name **social network of user X**, noted SN(X), all social websites for which user X has a **profile P**.

$$SN(X) = \{P_1, P_2, ..., P_i, ..., P_n\}$$

Many of these profiles enable their users to express their interests by using tags (e.g., *StumbleUpon*, *Flickr* or *Delicious*). Besides, each profile also holds acquaintances, i.e., relationships with other users (follower, member of a same group, etc.). Thus, a profile P is represented by a **set of tags T** and **a set of acquaintances A**.

$$P_i =< T_i, A_i > \quad \text{where } T_i = \{t_1, ..., t_j, ..., t_m\} \text{ and } A_i = \{a_1, ..., a_k, ..., a_t\}$$

Below, we denote t_{ij} the tag t_j included in the profile P_i. Besides, if the user X has an acquaintance noted a_k, this means that this acquaintance owns a profile - and a set of tags - on the same social website.

In our context, a **query Q** is similar to a query search, thus containing one or several terms. For instance, here is a query with two terms : *rock festival*.

$$Q = \{q_1, ..., q_l, ..., q_p\} \text{ where } q_l \text{ is a term.}$$

The idea underlying our work is to find which user(s), in the social network of user X, could answer a query Q. In other words, we want to find all acquaintances in all the profiles of user X, integrate all tagged profiles of each of these acquaintances and finally deduce which of them are best able to answer the query Q. In the next section, we describe our approach to fulfill this goal.

III. Our Approach

This section describes our approach to integrate all profiles of an acquaintance and compare the result of this integration process with the query. By first integrating all profiles of an acquaintance, we detect the main interests of this acquaintance. Then, the comparison with the query enables the computation of a similarity score between the query and the acquaintance, which in our context indicates how efficiently the acquaintance can answer the query. We are finally able to rank all acquaintances according to their similarity score with the query.

Let us present a use case: *Laly* plans to visit Greece and she would like more information about the country and its interesting places. By applying our approach to her social networks, she can discover that one of her *Flickr* acquaintance owns many pictures tagged with *Greece* since this acquaintance visited Greece. Similarly, *Laly* can find on *StumbleUpon* a Greek acquaintance who has written reviews about articles related to Greece.

The rest of this section is structured as follows. We first present different methods to extend and match tags. Then, we describe 3 strategies to answer a query on a user's social network. Each of these strategies exploits one or more methods for extending the tags, thus they have different features. For instance, the tree strategy is heavily based on a structural similarity. Finally, we discuss our approach.

A. Extending and Matching Tags

A crucial step in our approach deals with integration. Schema matching and ontology alignment research fields have provided many methods to discover similar elements in various data sources [8], [9]. Most of these methods can be used for both extending and matching tags:

- **Semantics**. Different resources can be used to obtain tags with similar meanings. DBpedia[6] provides various semantic relationships such as *rdf:type* or *owl:sameAs*. Abbreviations can be found through specific resources such as Tagdef[7] related to *Twitter* tags. For instance, the hashtag *#TVOH* refers to *The Voice Of Holland*, a Dutch TV program.
- **Structure**. Dictionaries or ontologies encompass an internal structure ranging from the general topics to very specific ones. Therefore, it is possible to detect hypernyms (ancestors, categories) of a given tag. For instance, the tag *Rembrandt* is included in the categories *Portrait artists* and *Dutch Golden Age painters*.
- **Terminology**. As tags are provided by users, they may be subject to misspellings or alternative spellings. For instance, the tag *Rembrandt* can be wrongly written *Rembrant* and the tag *web-2.0* can also be tagged *web20*.
- **Co-occurrence**. Given a resource (e.g., picture, article), users tend to associate different tags according to what they perceive. However, they also tend to share the same tags. For instance, the tag *Rembrandt* is often associated to the tags *painting*, *Netherlands* or *art*.

We note that all these methods can be applied locally (within the same profile) or globally (on different profiles). At the local level, they may be used to disambiguate the meaning of a tag. For example, if a profile contains the tags *rock*, *music*, *festival*, and *guitar*, then the tag *rock* is more likely to refer to the *music category* rather than the *mineral stone* according to co-occurrence method. At the global level, these methods mainly aim at integrating tags with similar meanings, i.e, matching them. As explained in [10], there are two ways of matching tags: we can use a thesaurus or an ontology against which the tags are matched thanks to previously described methods,

[6] http://dbpedia.org
[7] http://tagdef.com

and then apply similarity measures between elements of this thesaurus. Or one can match each tag with another one by applying one or more methods. In the next section, we describe our strategies : the *tree* strategy refers to a thesaurus technique while the *cluster* strategy performs direct matching between the tags.

B. Strategies for Answering a Query

We now present the different strategies which are used to integrate various profiles of the same user and then answer the query. Each profile includes a set of tags and a set of acquaintances. We note that a tag may be redundant, for instance different *Flickr* pictures of the same user may be tagged with *Greece*. Thus, extending the tags is not sufficient: profiles of the same user first need to be integrated to detect redundancies, alternative spellings and similar tags.

1) No-Extension Strategy: This strategy only aims at showing the benefits of our approach. Indeed, there is no extension step for the tags and no integration among the profiles.

Converting acquaintance's profiles. We do not extend or integrate the profiles with this strategy.

Converting a query. We do not extend the query with this strategy.

Ranking the acquaintances for answering the query. Each query term is matched against every tag of an acquaintance: if the query term appears in the acquaintance set of tags, then the acquaintance score is increased by 1. Figure 1 illustrates this strategy, with the term q_k being matched to all tags of an acquaintance. The acquaintance which has the more query terms obtains the best score and is ranked at the top.

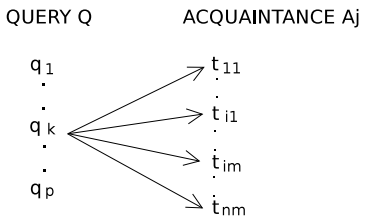

Fig. 1. The No-Extension Strategy

2) Cluster Strategy: The goal of this strategy is to create one or more clusters for the query and for the profiles. Each cluster gathers similar tags based on terminology, semantics and structure. The cluster strategy has been extended with respect to this work [11] by extending the tags with more methods. Below, we detail the different steps of this strategy.

Converting acquaintance's profiles into clusters. All similar tags of the same acquaintance are first integrated in a common cluster, i.e., we apply terminological and semantics methods to detect tags with close meaning. The tags which have not been integrated are gathered around a cluster labeled "Others". For the tags gathered around the "Others" cluster, we also store their number of occurrences. Once all tags have been clustered, they are extended by fetching synonyms with Wordnet and similar terms (*owl:sameAs*) with DBpedia.

Converting a query into clusters. Each query term becomes a cluster label. Each cluster is then extended using Wordnet and DBpedia.

Ranking the acquaintances for answering the query. The basic idea to compare a query cluster and an acquaintance cluster is that both clusters are similar if they share similar tags. We define a metric to compute the similarity between a query cluster C_q and an acquaintance cluster C_a. To ease notation, C_q represents the set of (extended) tags around this cluster and $|C_q|$ the size of this set. Here is the formula to compute the similarity between two clusters:

$$sim(C_q, C_a) = \frac{|C_q \cap C_a|}{\max(|C_q|, |C_a|)} \quad (1)$$

This formula would have no meaning if applied to the "Others" cluster. In the case where a tag from the query cluster is identical to a tag from the "Others" cluster, then we use the number of occurrences of the "Others" tag to compute a similarity between the query cluster and the acquaintance cluster "Others". The assumption is that the more occurrences a tag has in the acquaintance profiles, the more weight (in the range [0, 1]) it should have.

$$sim(C_q, C_{Others}) = \sum (1 - \frac{1}{Occ_t}) \quad (2)$$

where t is a tag $\in C_q \cap C_{Others}$ and Occ_t represents the number of occurrences of tag t.

Finally, we sum the similarity score of all clusters of an acquaintance, thus enabling their ranking to answer the query.

Figure 2 illustrates this cluster strategy. The left clusters are the query clusters while the right ones stand for an acquaintance. All labels X_i represent an extended tag for the cluster. As clusters q_1 and t_{im} share one common tag X_2, their similarity equals $\frac{1}{3}$ by applying formula 1. For the cluster "Others", let us imagine that the tag t_{i1} occurs 8 times in the acquaintance's profiles and that it is similar to the tag X_3. In that case, the similarity between the clusters q_k and "Others" is equal to $1 - \frac{1}{8} = \frac{7}{8}$.

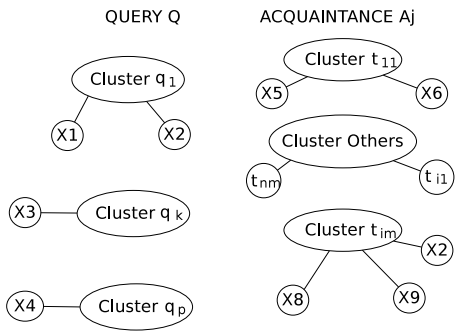

Fig. 2. The Cluster Strategy

3) Tree Strategy: The basic idea behind this strategy is to build a tree for the query and one tree for each acquaintance. These trees are based on the Wordnet hierarchical structure, from which we can compute well-known similarity measures between trees. This means that we do not do direct matching between the tags, but we match them against a common dictionary.

Converting acquaintance's profiles into a tree. Each tag is searched in Wordnet. If it is found, then the tag becomes a leaf node in the acquaintance tree and all its ancestors in the Wordnet hierarchy becomes intermediary nodes between the leaf node and the root node (labeled *entity*). If the tag was not found, then we extend it by using semantics (DBpedia categories) and co-occurrence and we try to find the extended tags in the Wordnet hierarchy to add it in the tree with its ancestors. The tag is discarded if none of its extension could be found in Wordnet. At the end of this process, we obtain one tree which integrates all profiles of an acquaintance. We note that each leaf node in the tree is a profile tag and that all trees have at least a common (root) ancestor.

Converting a query into a tree. The query is converted into a tree based on its terms, similarly as the profiles are. However, the leaf nodes are the query terms.

Ranking the acquaintances for answering the query. As everything is represented by trees, we have to compare the query tree with each acquaintance's tree. Comparing two trees has been studied at large in the literature [12], [13]. However, in our context, we are mainly interested by the leaf nodes. Thus, we compute a similarity score between all query leaf nodes and all acquaintance leaf nodes. The similarity measure between two nodes is from Resnik and it states that *"the more information two concepts share, the more similar they are, and the information shared by two concepts is indicated by the information content of the concepts that subsume them in the [Wordnet] taxonomy"* [14]. The Resnik similarity returns a similarity value between two nodes in the range [0, 1]. Each query leaf node may therefore be matched to one acquaintance tree node (the one with the highest similarity value). At the end, we sum all the similarity values of these matches, resulting in a global score between the query and the acquaintance. This score enables us to rank each acquaintance and find the one who can best answer the query.

Figure 3 depicts this tree strategy with a left tree for the query and the right one for an acquaintance. All nodes labeled *A* to *J* are intermediary nodes (i.e., ancestors from Wordnet hierarchy). Let us study the query node q_k and the acquaintance node t_{im}. Both nodes share a common ancestor *A* - excluding the *Entity* root node - so that the Resnik similarity between q_k and t_{im} would indicate that these nodes match.

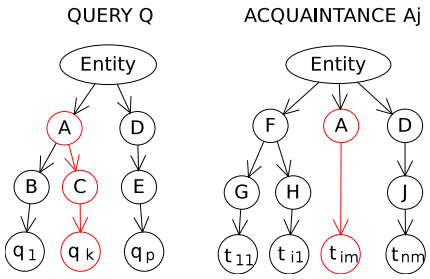

Fig. 3. The Tree Strategy

C. Discussion

This section discusses our approach:

- Several meanings for a tag requires disambiguation. Applying matching techniques at the local level does not always solve the problem. For instance, a tag *rock* in a profile can refer to *music* or *nature*.
- As the tags are filled in by users, we have faced many misspelled tags. Although most of them could be corrected by applying terminological and dictionary techniques, we believe that this step should be performed by the social networks in charge of storing the profiles.
- Our approach does not take into account the availability of users. Yet, this kind of information could be useful to improve the ranking since the more time an acquaintance spends on her social networks, the more chances she could provide a quick answer to the query.
- We cannot consider that our work is the same as the task of expert finding. Indeed, expert finding often implies a sort of authority in the domain. Hypotheses described in [15] reflect these authoritative links (i.e., the impact of a journal or conference of a publication or the social connectedness with an expert). In user's profiles, we rely on comments, tags and interests provided by users. In other words, a user who has the term "cat" a hundred times in her profiles is probably fond of this animal, but it does not mean she is an expert (as a vet could be for instance).

The next section deals with experiments to show the benefits of our approach.

IV. Experiments

This section describes preliminary results of our approach. We mainly aim at showing which strategies provide acceptable results, and how scalable our approach is. We first detail our experiments protocol and the report the results.

A. Experiments Protocol

1) Query Dataset: Alexa [8] is a website specialized in web information. Everyday, it computes the 20 most popular query searches on the Web and it archives them since 2008. By scraping these queries and removing duplicates, we obtained a query set containing 3687 queries. 23% of these queries are composed of 1 term, 55% of 2 terms and the rest includes more than two terms. Finally, 1617 queries have a direct entry on *DBpedia* while (all terms from) 1568 queries can be found on *Wordnet*.

[8] http://www.alexa.com/hoturls

2) Social Networks: We have used the datasets provided by Mypes [7]. It includes 320 users, and each of them has three tagged profiles: one on *Delicious*, one on *Flickr* and one on *StumbleUpon*. The average number of tags per profile is 191 for *StumbleUpon*, 482 for *Delicious* and 532 for *Flickr*. From this users' set, we are able to generate random social networks, i.e, random users are chosen to form a set of acquaintances.

3) Evaluation: Each experiment scenario includes a random query and a random social network with 2 to 200 users. For extending and integrating the tags, our approach is based on the following configuration: *Flickr*[9] for co-occurrence, *Wordnet* for related terms, *DBpedia* for categories and *owl:sameAs* relationships, an average between Jaro Winkler, Monge Elkan and Scaled Levenshtein similarity measures with a 0.8 threshold for terminology [16]. The result of an experiment scenario is a ranking of all acquaintances in the social network, with the top ones which have been considered as the best to answer the query. When possible, a reason justifying the place of a user in the ranking is provided such as a list of tags considered as suitable for query answering.

Evaluation should be analyzed towards two directions: time performance and quality of the ranking. The former is computed in seconds. For the latter, a manual evaluation is required but we could not analyze hundreds of experiments involving hundreds of acquaintances. Besides, estimating the quality degree of a ranking with hundred acquaintances is also difficult and we are still in search of an acceptable quality evaluation. Thus, we describe several interesting cases for each strategy and we provide general statistics for the quality evaluation.

B. Experiments Results

This experiments report is divided into two parts: performance and quality.

1) Performance Evaluation: Figure 4 depicts the time performance (in seconds) of each strategy when the number of acquaintances in a user's social network varies. First, we note that the *no-integration* strategy does not require much time (mainly close to 0 seconds) whatever the number of acquaintances. The *tree* strategy is linear with the number of acquaintances: its time performance slightly increase to reach 40 seconds with 200 acquaintances. This is justified since the comparison of two tree structures is a well-known problem. We could also use in the future a *B-tree* structure to have a direct access to all leaf nodes, thus reducing the time for tree traversal. Finally, the *cluster* strategy requires more execution time than the others (180 seconds with 200 acquaintances). Contrary to other strategies, the *cluster* strategy aggregates tags of the same acquaintance based on various methods. This step is very costly while the matching between the query's clusters and the acquaintances' clusters is performed in a few seconds.

[9]http://www.flickr.com/services/api/flickr.tags.getRelated.html

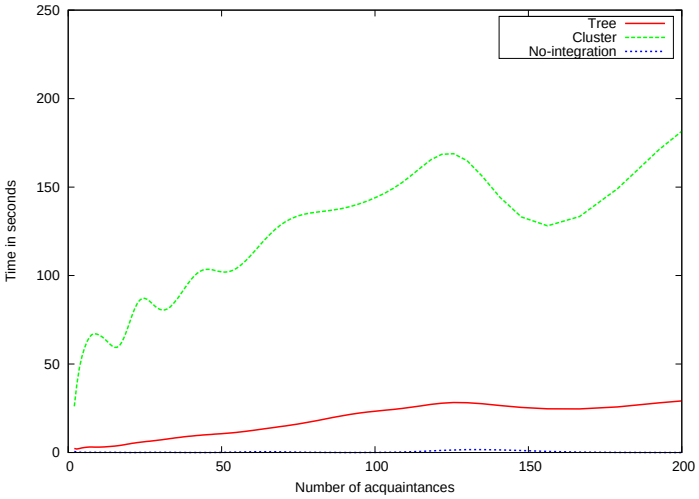

Fig. 4. Scalability of each strategy according to the number of acquaintances

2) Quality Evaluation: In this section, we study some specific cases to show when our approach works or not.

Comparing our strategies. The no-integration strategy only provides a ranking with acquaintances if the query tags is present in the acquaintance's profiles. Thus, it does not provide any result in many cases. *For the query "ipad", the no-integration strategy could rank users with an interest in this device since their profiles contain the tag "ipad".*

Now let us analyze results for the two other strategies, *tree* and *cluster*. As they integrate the profiles differently, the strategies may provide different rankings. *For instance, with the query "costume", the tree strategy was able to detect acquaintances with tags "clothing" because this tag is an hypernym of "costume" in the Wordnet hierarchy. On the other hand, the cluster strategy discovers acquaintances with tags such as "carnival" or "mask".*

However, in many cases, the rankings are very similar between both strategies. *For the query "Brazil", both strategies are able to discover acquaintances with tags like "Brazil", "Brasil" and "Saopaulo". These results are comfirmed by the fact that most other tags in the ranked acquaintances are in Portuguese language.*

In other experiments, the *cluster* strategy is the only one to provide a ranking thanks to its deep extension. *For the query "vikings", it extended the query with terms like "Scandinavia" or "ship". We note that the ranking also contains acquaintances with an interest for Minnesota Vikings, an American football team.* Besides, the cluster strategy tends to rank more acquaintances. *With the "snow" query, the tree strategy returns 6 ranked acquaintances while the cluster one ranks 23 acquaintances.*

Topic drift. When a tag is too much extended, there is a risk of topic drift, i.e., that the extended tags do not reflect anymore the restricted meaning of the initial tag. The cluster strategy is mainly affected by topic drift because we fully extend both the query and the acquaintances' profiles. *For instance, the query "gold" leads us to topic drift. Indeed, it has been extended with tags such as "Nikon", "black", "light". Thus, the ranking includes many acquaintances who are interested in photography. In another experiment, the query "turkey" is correctly identified as a bird by the tree strategy while the cluster strategy returns acquaintances with tags "Christmas" or "Thanksgiving".*

Query disambiguation. We have described issues related to profile tags with different meanings. But it also happens that the query has ambiguous terms. In that particular case, we believe that the query should be disambiguated before searching for an answer. *For example, we have a query "torrent". We first assume that this query refers to the P2P technology. However, the top-ranked acquaintances have profiles with tags such as "nature", "river", "stream" or "waterfall".*

Summary of experiments. We have run 534 experiments. For 311 experiments, the *cluster* strategy provided a ranking, which means that it considered that at least one acquaintance could answer the query. A ranking is obtained in 171 experiments (respectively 135) for the *tree* (respectively *no-integration*) strategy. We note that the *tree* strategy provides more rankings than *no-integration* because it discovers interests nested in the Wordnet structure. If we could compare the *tree* and the *cluster* strategies in terms of precision and recall, the former strategy would tend towards high precision (i.e., most ranked acquaintances have an interest related to the query) to the detriment of recall (i.e., it would miss interesting acquaintances). On the contrary, the *cluster* strategy often ranks more acquaintances (thus promoting a higher recall), but it is also more subject to topic drift (thus decreasing precision).

V. Conclusion

In this paper, we have presented a new approach to find acquaintances who can answer a given query in one's social network. Three strategies are based on different techniques to rank acquaintances. Preliminary experiments have been reported: the *tree* strategy is efficient in terms of performance but it often misses acquaintances that can answer the query. On the contrary, the *cluster* strategy provides better results but to the detriment of time performance. However, we are still thinking about a better evaluation for the quality of the ranking.

Our perspectives several directions. To improve the quality evaluation, we could select users with scattered profiles and answer a query online by ranking their acquaintances. These users could assess the quality of our ranking. In addition, determining the relationship between the user and the tag could help us clarify the degree of expertise between this user and his/her tag. We note that a user who tagged "Amsterdam" in *Flickr* may either have taken pictures in Amsterdam or add a comment about someone's picture about Amsterdam. In a broader way, we could increase the chances of discovering an expert by extending the social network of the user. We could spread the query to the social networks of all the user's contacts. It seems realistic to ask a contact in your social network for help to one of her/his contact.

VI. Acknowledgments

We would like to thank the CWI/INS2 team for useful comments. And we are very grateful to the authors of Mypes for providing us with their profiles dataset.

References

[1] N. Contractor, "From disasters to wow: Using web science to understand and enable 21st century multidimensional networks (invited talk)," in *ESWC*, 2010.

[2] J. Zhang, J. Tang, and J. Li, "Expert finding in a social networks," in *Database Systems for Advanced Applications*, 2007.

[3] Y. Fu, R. Xiang, Y. Liu, M. Zhang, and S. Ma, "Finding experts using social network analysis," nov. 2007, pp. 77 –80.

[4] T. Lappas, K. Liu, and E. Terzi, "Finding a team of experts in social networks," in *Knowledge Discovery and Datamining*. New York, NY, USA: ACM, 2009, pp. 467–476.

[5] J. Li, "Rule-based social networking for expert finding," 2006. [Online]. Available: http://ruleml.org/usecases/foaf/JieLiMCSThesis.pdf

[6] X. Li, L. Guo, and Y. E. Zhao, "Tag-based social interest discovery," in *WWW '08: Proceeding of the 17th international conference on World Wide Web*. New York, NY, USA: ACM, 2008, pp. 675–684.

[7] F. Abel, N. Henze, E. Herder, and D. Krause, "Linkage, aggregation, alignment and enrichment of public user profiles with mypes," in *I-SEMANTICS '10: Proceedings of the 6th International Conference on Semantic Systems*. New York, NY, USA: ACM, 2010, pp. 1–8.

[8] J. Euzenat and P. Shvaiko, *Ontology matching*. Heidelberg (DE): Springer-Verlag, 2007.

[9] P. Shvaiko and J. Euzenat, "A survey of schema-based matching approaches," *Journal of Data Semantics IV*, pp. 146–171, 2005.

[10] C. Cattuto, D. Benz, A. Hotho, and G. Stumme, "Semantic grounding of tag relatedness in social bookmarking systems," in *International Semantic Web Conference (ISWC2008)*, October 2008. [Online]. Available: http://data.semanticweb.org/conference/iswc/2008/paper/research/177

[11] F. Duchateau and L. Hardman, "Integrating and Ranking Interests From User Profiles," 2010, pp. 28–39. [Online]. Available: http://CEUR-WS.org/Vol-595/paper3.pdf

[12] R. Yang, P. Kalnis, and A. K. H. Tung, "Similarity evaluation on tree-structured data," in *SIGMOD*. ACM, 2005, pp. 754–765.

[13] F. Duchateau and Z. Bellahsene, "Measuring the quality of an integrated schema," in *ER - International Conference on Conceptual Modeling*, To appear in 2010.

[14] P. Resnik, "Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language," *J. Artif. Intell. Res. (JAIR)*, vol. 11, pp. 95–130, 1999.

[15] M. Stankovic, C. Wagner, J. Jovanovic, and P. Laublet, "Looking for experts? what can linked data do for you?" in *Linked Data on the Web*, 2010.

[16] W. Cohen, P. Ravikumar, and S. Fienberg, "A comparison of string distance metrics for name-matching tasks," in *In Proceedings of the IJCAI-2003*, 2003. [Online]. Available: citeseer.ist.psu.edu/cohen03comparison.html