User profile matching in social networks

Elie Raad, Richard Chbeir, Albert Dipanda

User Profile Matching in Social Networks

Elie Raad
LE2I, Bourgogne University
Dijon, France
elie.raad@u-bourgogne.fr

Richard Chbeir
LE2I, Bourgogne University
Dijon, France
richard.chbeir@u-bourgogne.fr

Albert Dipanda
LE2I, Bourgogne University
Dijon, France
albert.dipanda@u-bourgogne.fr

***Abstract*—Inter-social networks operations and functionalities are required in several scenarios (data integration, data enrichment, information retrieval, etc.). To achieve this, matching user profiles is required. Current methods are so restrictive and do not consider all the related problems. Particularly, they assume that two profiles describe the same physical person only if the values of their Inverse Functional Property or IFP (e.g. the email address, homepage, etc.) are the same. However, the observed trend in social networks is not fully compatible with this assumption since users tend to create more than one social network account (for personal use, for work, etc.) while using same or different email addresses. In this work, we address the problem of matching user profiles in its globality by providing a suitable matching framework able to consider all the profile's attributes. Our framework allows users to give more importance to some attributes and assign each attribute a different similarity measure. The set of experiments conducted with our default/recommended `attribute/similarity` measures shows the superiority of our proposal in comparison with current ones.**

***Index Terms*—Social networks; Profile matching; FOAF;**

I. Introduction

Since its creation, the web was not only used to read information, make business, connect pages, but also it was meant to be a social tool for users. Nowadays, social networking has become an important part of the online activities on the web. Social sites gain popularity thanks to the diverse services provided ranging from collaborative tagging (e.g., Flicker [1]), blogging sites (e.g., Livejournal [2]), and mainly to social networking (e.g., Facebook [3], LinkedIn [4], MySpace [5]) with a nonstop growing number of active users.

In essence, each social network offers particular services and functionalities that target a well defined community in the real world. To make use of the provided services/functionalities and to keep being tuned with its related members, users create several accounts on various sites. This has participated in the emergence of new users' related needs to perform some inter-networks' operations and functionalities. To illustrate this, let us consider the following scenario. Bob, a software developer, is very active on social networks. As illustrated in Figure 1, he mainly uses two social sites: the first is Facebook (SN1) to stay connected with his friends, and the second is LinkedIn (SN2) to maintain professional contact with a group of software developers. For different purposes, Bob needs to identify:

1) **Intersection between SN1 and SN2**: to allow him invite related friends (Nel and Rosy) to technically test his new Facebook add-ons
2) **Union between SN1 and SN2**: to help him send a gift (containing his company promotional package and Facebook add-ons) only once (so to reduce costs) to people that might be interested in his add-ons (such as James, Deborah, Peter, Richard, Lorie, Yi, Dupond, Rosy, and Nel)
3) **Difference between SN1 and SN2**: to allow him to enrich his friends' profiles with complimentary information found in both sites (particularly Nel and Rosy here).

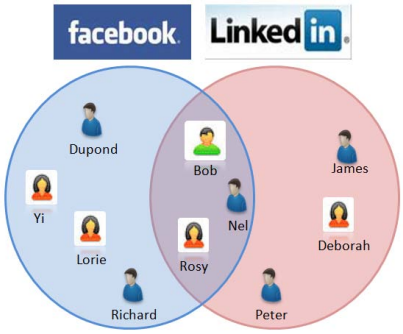

Fig. 1. Social Network of Bob within Facebook and LinkedIn.

Performing this kind of operations requires in one way or another the matching of users' profiles. In fact, the user profile matching[6] consists of accurately linking records corresponding to the same entity in the same or different data sources. However, matching user profiles on social networks suffers currently of three main problems:

- **Social Network Representations:** Social networks offer to users interesting means and ways to connect, communicate, and share information with other members within their platforms. However, those sites have currently different structures/schemas and they represent users' profiles differently. Thus, they prohibit the exchange of information and communication with other social networks

[1]http://www.flickr.com/
[2]http://www.livejournal.com/
[3]http://www.facebook.com/
[4]http://www.linkedin.com
[5]http://www.myspace.com/

[6]Also known as *record linkage, entity resolution, record matching, object identification, and reference reconciliation*

(such as sharing pictures, tags, and comments) making them functioning as "Data Isolated Islands" [1].

- **User Profile Domains:** Even when sites share the same representation, user profile attribute domains are not always common. For instance, the domain values of `interests` attribute in Facebook do not necessarily meet the domain values of the same attribute in LinkedIn.
- **Site/User Objectives:** Depending on the site and on the user objectives, the same attribute can be filled up with two different values. For instance, the `email` attribute in Facebook is commonly filled with a personal email while LinkedIn one is assigned to the professional email of the same user.

In this study, we address the problem of providing inter-social networks' operations and functionalities and particularly focus on the user profile matching. Our contribution in this paper is a matching framework able to consider all the profile's attributes. Using our proposal, users can give more importance to some attributes and assign each attribute a different similarity measure. The set of experiments and tests conducted with our default/recommended `attribute/similarity` measures shows the superiority of our proposal in comparison with current ones.

The remainder of the paper is organized as follows. In Section II, we present some related works. In Section III, we introduce our approach to find social network profiles by using the set of functional properties. In Section IV, we discuss the results of the conducted experiments. Finally we conclude and describe future works in Section V.

II. Related Works

Recently, technologies dealing with the issue of resource integration between profiles are getting a growing attention. In this section we present a number of approaches and techniques that were used to tackle this problem.

A. Approaches depending only on IFP:

In [2], Ding et al. proposed a heuristic approach to identify and to discover FOAF documents from the Web and to extract information about people from these FOAF documents. The authors consider that the FOAF unique identifier such as *foaf:mbox_sha1sum*, and *foaf:homepage*, are the ideal clues for information fusion. Some other identifiers such as *foaf:name* may also be useful in giving some clues. The author urged for caution when merging information from many FOAF documents since some of the facts may be wrong thus resulting into contradictory information. Flink, a system developed in [3], is able to determine the identity of individuals across multiple information sources by reasoning on IFP comparison or on name matching implemented within its code. Name matching computes the similarity between two names, but the dissimilarity between last names is not allowed. However, performing the matching by considering only the name is not enough, accuracy needs the use of more attributes for better matching results. In [4], Golbeck et al. showed by reasoning on FOAF profiles, that thousands of users have accounts on multiple social networks, linking their subgraphs in the unified social network. In their presented study, their reasoning is only based on the *foaf:mbox_sha1sum IFP* to infer that two profiles are the same or not. However, to detect profiles that refer to the same users but created with different *IFP*, other approaches and methods must be proposed resulting into a bigger intersection of users between different social network sites. Some other works defined their own *IFP* attribute [5] or new relationship types [1] that suit their needs.

B. Approaches going beyond only the IFP:

In [6], the authors consider that the single use of an IFP, such as the *foaf:mbox_sha1sum* in FOAF, is not suitable. They provided some explanation showing that it is very common for a user to have two social network accounts with different email address. In their work, they cited the following reasons: 1) People change email address, 2) People use more than one email address depending on the context of use, 3) Email addresses can act as proxies for more than one person. Then they presented an extended service called Foaf-O-Matic for the creation of FOAF profiles. This service is based on issuing a globally unique identifier for users and storing it in an infrastructure. In this work, the primary user has a manual task of adding and identifying each friend as well as determining by himself duplicated friends profiles. In their proposed application, they seek to propose a user-friendly way to include the identifier to the FOAF profile. In [7], the authors propose to disambiguate the identity of a user by using the social circles of the users and some social data tagged with the name of the user. Social circles represent a group of people linked to a central individual by some identifiable common relation. It is then up to the user to decide which identity features are best suited to minimally distinguish their identity from others. However, user based feature identity selection is a potential drawback when performing the user disambiguation process. In [8], the authors studied separately and compared two approaches that can identify the co-occurrence of the same person across different communities. The first approach is based on the *IFP* and the second approach is based on heuristics particularly for comparing entity labels by using a simple but strict string comparison technique. However, since dealing with identity reasoning is not a trivial task and in order to obtain good results it is crucial to implement both of the methods that exploit IFP and the Information Retrieval techniques for string and semantic similarity.

III. Proposed Framework

Our goal is to discover the biggest possible number of social profiles that refer to the same person between two social networks. To do that, we investigate three main areas: social network profile heterogeneity, similarity measuring between attribute values, and decision making about whether two profiles refer to the same person or not. Here, we propose a framework composed of 4 main components as illustrated in Figure 2, each detailed in the following subsections.

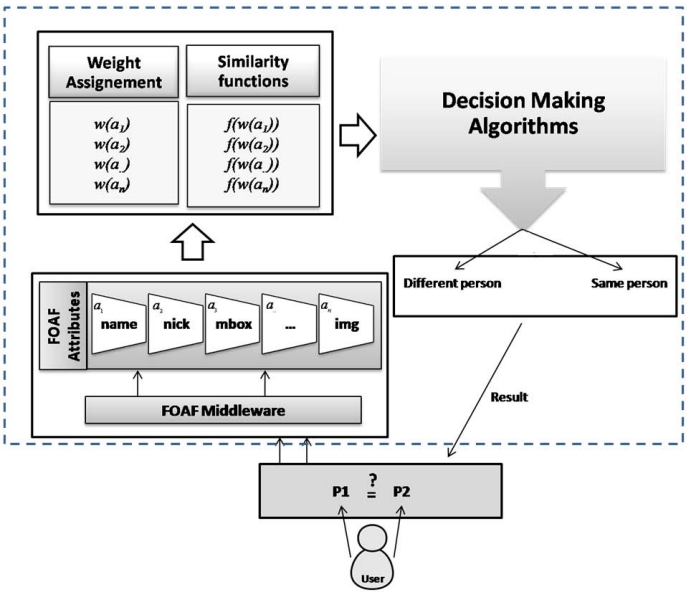

Fig. 2. Main components of the proposed approach

Fig. 3. Two sample FOAF user profiles

A. FOAF Middleware

As mentioned previously, current social networks do not adopt the same user profile representation. This has been pinpointed by the W3C workshop [7] and concluded that most of the technologies needed to create decentralized social networks exist, such as: RDFa[8], Microformats[9], XHTML Friends Network (XFN)[10], Friend Of A Friend (FOAF)[11].

Nowadays, FOAF is admitted to be one of the real success story of the semantic web [9] and is becoming a *de facto* standard with more and more social networks and tools that allow to create/generate FOAF profiles [4] . In reality, it is a machine-readable semantic vocabulary describing people, their relationships, and activities. It is written in XML syntax and adopts the conventions of the Resource Description Framework (RDF) to define a set of attributes. A simple FOAF example is provided in Figure 3. We opted to FOAF as a common representation of social profiles and dedicated this component to transform the input profiles into FOAF.

B. Similarity Function Assignment

Comparing two profiles comes down to compare (a set of) their attributes. In order to obtain appropriate results, adapted similarity function(s) must be associated to each attribute (e.g. comparing `emails` must be computed in a different way than comparing `interests`). Various techniques can be used to measure the similarity score between two textual/string values and can be grouped into 2 main categories:

- **Syntactic-based similarity approaches**: provide exact or approximate lexicographical matching of two values. Using exact similarity techniques can lead to poor similarity results since frequent variations of a word exist and typing errors are common. Thus, approximate string matching techniques can be used to compute the distance between two values that have a limited number of different characters.
- **Semantic-based similarity approaches**: are used to measure how two values, lexicographically different, are semantically similar. They can be:
  - Knowledge-based [10]: computing similarity between values with the usage of predefined (or external) knowledge resources (taxonomies, ontologies, etc.) such as WordNet, Wikipedia, etc. The similarity can be edge-based (computed following the distance separating values to be compared in the external knowledge) or node-based (computed following the amount of information that a concept contains).
  - Corpus-based [11]: computing the similarity between two concepts using large corpora only (and without external knowledge resources). The similarity can be based on vector-space model, statistical such as Pointwise Mutual Information Information Retrieval, or Latent Semantic Analysis.

Consequently, assigning default similarity functions to FOAF attributes must be done carefully with respect to the mentioned categories and the domain values. Figure 4 summarizes our default similarity measure assignments to FOAF attributes.

1) **Senseless One-term attributes:** As stated in [12], [13], Jaro metric [14] is considered as one of the optimal measures to be primarily intended for short string comparison. It is based on the number and order of the common characters between two strings. The definition of common characters is that the agreeing characters must be within half of the length of the shorter string. The Jaro distance similarity between two strings s and t can be computed as follows:

$$sim_{Jaro}(s,t) = \frac{1}{3}\left(\frac{|s'|}{|s|} + \frac{|t'|}{|t|} + \frac{|s'|-0.5\times T_{s',t'}}{s'}\right),$$

where:

- $|s|$ and $|t|$ are the length of each string,
- $|s'|$ and $|t'|$ are the number of common characters,
- T is the number of transposed characters.

[7] http://www.w3.org/2008/09/msnws/
[8] http://www.w3.org/MarkUp/2009/rdfa-for-html-authors
[9] http://microformats.org/wiki/Main_Page
[10] http://gmpg.org/xfn/
[11] http://xmlns.com/foaf/spec/

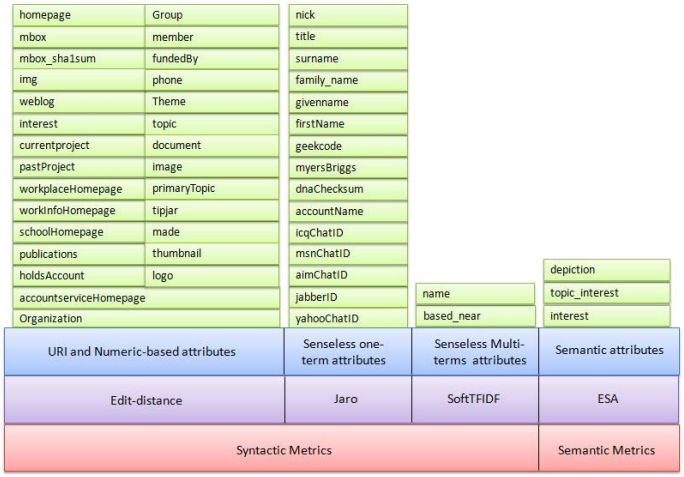

Fig. 4. Default metrics to compute similarity of each FOAF attribute

2) **Senseless Multi-terms attributes:** The SoftTFIDF metric [12] is one of the best techniques [15] that combines the token-based (or words) and string-based methods to compute similarity between sentences. It is based on the cosine similarity that doesn't automatically discard words which are not strictly identical. This metric has two main advantages: 1) the token order is not important, 2) common uninformative words don't greatly affect similarity [15], [16]. The SoftTFIDF similarity measure can be computed between s and t as follows:

$$Sim_{SoftTFIDF}(s,t) = \sum_{w \in close(\phi,s,t)} V(w,s) \times V(w,t) \times D(w,t),$$

where $close(\phi, s, t)$ is the set of words $w \in s$ such that there is some $v \in t$ and $dist'(w,v) \geq \phi$, and for $w \in close(\phi, s, t)$, $D(w,t) = max_{v \in t} dist(w,v)$.

3) **Semantic-based attributes:** The Explicit Semantic Analysis (ESA) is a technique that uses Wikipedia to compute semantic relatedness and considered one of the best existing methods [17]. Each concept is represented by a weighted vector that contains a text describing the concept with a weight computed using the TFIDF measure. A semantic interpreter is formed by all the concepts and their weighted terms. It tries to match each word to the most relevant concepts based on a defined threshold. For a more efficient search, a constructed inverse interpreter index maps each word to all the concepts that are part of them. A weighted vector that represents the relevance of the concepts to a vector with a weight is calculated for each text snippet. At the end, the cosine measure is applied to the two vectors so to compute the relatedness between the two text snippets.
4) **URI and Numeric-based attributes:** The Edit Distance (ED) metric [18] is the most suited technique to compute similarity for this kind of attributes. It measures the distance between two strings, *s* and *t*, by calculating the cost of the minimum number of editing operations (insertions, deletions, and substitutions), commonly called edit script, that convert *s* to *t*. The edit distance similarity between two values s and t can be computed as follows:

$$sim_{EditDistance}(s,t) = 1 - \frac{d}{max(ls,lt)},$$

where:

- *s* and *t*: the two values to compare,
- *d*: the distance (cost) between *s* and *t*,
- *ls* and *lt*: the length of *s* and *t* respectively,
- *max(ls,lt)*: the maximum length between *s* and *t*.

To illustrate all this, we applied the default metrics given above to compute the similarity between the attributes of two sample profiles provided in Figure 3 belonging to the same person. Table I shows the obtained similarity scores. One can see that default metrics provide the best similarity scores.

TABLE I
SIMILARITY SCORES USING DEFAULT SIMILARITY METRICS

Attributes/Similarity Metrics	Jaro	ED	SoftTFIDF	ESA
$< foaf:name >$	0.72	0.12	**0.99**	0
$< foaf:firstname >$	**0.85**	0.6	0.85	0
$< foaf:img >$	0.77	**0.8**	0.66	0
$< foaf:interest >$	0.52	0.22	0	**0.75**

C. Attribute Weight Assignment

This component mainly aims to assign a weight to each attribute in the FOAF vocabulary. This allows to represent the attribute importance within a defined context. In our framework, the weight can be assigned manually or computed automatically. Manual assignment allows users to include their preferences and inputs in the matching process (e.g. `mbox` attribute may be the most important for a user) while automatic assignment is provided in order to allow considering related social network characteristics (e.g. `homepage` attribute is more important on LinkedIn than on Facebook). Of course, the user can use both (he can start with automatic assignment and tune it manually after having received the results). In the Automatic assignment, the user gives the framework as input either the list of related social networks or the list of his/her accounts on each social network with the list of **IFP** attributes.

In both cases, the following steps are processed as described in Algorithm 1. The default IFP is the *foaf:mbox_sha1sum* as it is defined in the FOAF vocabulary. Once the input parameters are given, the component attempts, if it is not done previously, to compute the importance of each attribute by crawling the related social networks and storing the concerned profiles (user and friends) locally. The retrieved profiles are transformed into FOAF representation via FOAF Middleware component. Data analysis is then performed by computing the similarity score between the attribute's values of each pair of profiles having the same IFP. To do that, profiles having the same IFP value are firstly extracted from the dataset obtained. Then, the similarity between their attribute values is computed. This is done for each pair of extracted profiles. At the end, each attribute will be associated with a set of similarity scores to be used to compute its final weight.

Algorithm 1: Assigning weights to attributes

Input:
IFP : List of Inverse Functional Property,
P: Set of profiles having the same IFP values,
A: Set of all attributes used to describe profiles,
f_{fusion}: Fusion function
Data:
pc: Number of pair of profiles having the same IFP,
Output: w: Vector of weights assigned to attributes

```
begin
    foreach P_i in P do
        foreach P_j in P \ P_i do
            if (P_i.IFP == P_j.IFP) then
                foreach a_i in (P_i ∩ P_j) do
                    v[pc][a_i]=sim(P_i.a_i, P_j.a_i)
                end
                pc++
            end
        end
    end
    foreach a_i in A do
        for p=1 to pc do
            r[a_i]=v[p][a_i]
        end
        w[a_i] = f(r)
    end
    return w
end
```

At this point, data fusion/aggregation techniques are needed to combine information from different sources and to obtain one result for a more accurate decision. Several approaches are commonly used for data fusion such as the probabilistic methods: Bayesian Networks (BN), the evidence theories: DempsterShafer (DS), the fuzzy set theories: Fuzzy Decision Trees (Fuzzy DT), and other classical functions: Average (Avg), Minimum (Min), Maximum (Max), etc.) [19] studied and extensively used in many fields (e.g. database, multimedia, security, etc.). In our study, we adopt and recommend the Avg and/or the Fuzzy DT as default fusion/aggregation functions. Our choice is related to the following reasons: 1) Granting the lowest or the highest weight to an attribute using the Min or Max functions is inappropriate as it represents a special case. 2) Applying the BN or DS methods is not relevant since they require respectively the use of the probability values of different sources and the mass function calculated from probabilities that deals with uncertainty. 3) Aggregating multiple belief structures on the same variable (attribute) is considered an issue in the theory of DS as stated in [20]. 4) Values computed for one attribute are similarity values and not probability ones as it is the case in BN.

D. Profile Matcher

This component aims to provide a decision whether two input profiles refer to the same physical person or not. Here, two profiles are considered as representing the same user if their profile similarity score is higher than a threshold called the *profile matching threshold*.

Computing those two scores using a set of valued attributes can be a complex process due to the incompleteness and the uncertainty of the used information. For decision making, several methods can be used as detailed in [21]. In this work, we chose the DS function, as the default method, while leaving for the users the option to modify the default settings. Our choice is based on the following reasons: 1) One of the main complications in BN is when a new evidence is added, the probabilities at each node are recomputed to propagate the evidence through the nodes. 2) Another drawback for BN is that it cannot distinguish the lack of evidence for a proposition from the evidence against the proposition meanwhile DS theory can make the difference [19]. 3) In fact, this advantage of DS is the result of the non existence of a causal relationship between a hypothesis and its negation, so the lack of belief does not imply disbelief. 4) The DS theory is able to represent both imprecision and uncertainty, flexibility, and its ability to consider more than one class for decision making. In the following, we explain how to compute the profile matching threshold and the similarity score between two profiles.

1) Computing the profile threshold matching: It is the minimal similarity value required for matching two profiles. We propose to compute this threshold using the weights assigned to each attribute. The assumption here is that those weights are the result of an attribute based aggregation of values coming from profiles that refer to same physical users. Based on this, the weights form reliable measures and can be considered as reference values for computing a profile matching threshold. This threshold is computed as follows:

$$th = f_{decision}\left(w\left(a_0\right), w\left(a_1\right), .., w\left(a_n\right)\right)$$

where:

- th: the profile matching threshold to compute,
- $f_{decision}$: the decision making algorithm used,
- a: the attributes used to describe a user profile,
- n: the number of available attributes,
- w: the weight assigned to each attribute.

2) Computing similarity scores between two profiles: For the similarity score, the values of common attributes in both profiles are extracted and their similarity scores are computed. Then, the obtained similarity scores are tuned in order to have more realistic scores that take into consideration the importance assigned to each attribute. By doing so, the new similarity value will tend to increase or decrease depending on the importance of each attribute. This tuning is an attribute-based operation that outputs a new similarity score to each attribute by applying a weight to the computed similarity scores. The new similarity score is computed as follows:

$$sim'\left(P1.a_i, P2.a_i\right) = \frac{2 \times sim(P1.a_i, P2.a_i) \times w(a_i)}{1 + (sim(P1.a_i, P2.a_i) \times w(a_i))} \in [0, 1]$$

where:

- a_i an attribute used to describe a profile,
- $P1.a_i and P2.a_i$ are two values of an attribute a_i in Profile P1 and Profile P2,
- $w\left(a_i\right)$ the computed/assigned weight of an attribute $\in [0, 1]$,

- $sim\,(P1.a_i, P2.a_i)$ the similarity score computed between the values of an attribute in P1 and P2 $\in [0, 1]$,
- $sim'\,(P1.a_i, P2.a_i)$ the new similarity score computed between the values of an attribute in P1 and P2 $\in [0, 1]$,

The new similarity scores of all attributes are sent to a decision making algorithm. The task of this algorithm is to return a value, v, that represents the similarity score between two profiles. This is shown in Algorithm 2.

Algorithm 2: Deciding whether two profiles refer to the same user or not

Input: P_1, P_2 : Profile of user 1 and user 2,
$P1.a_i and P2.a_i$ are two values of an attribute a_i in P1 and P2,
$f_{decision}$: Decision making function,
Output: result: Matching

```
1  begin
2      foreach a_i in (P_1 ∩ P_2) do
3          k[a_i] = sim'(P1.a_i, P2.a_i)
4      end
5      D = f(k)
6      if D ≥ th then
7          result = true
8      end
9      else
10         result = false
11     end
12     return result
13 end
```

IV. Implementation and Experimentations

In this section, we present the prototype that we implemented to validate our approach. We also explain the results of a set of experiments conducted to test and prove the relevance of our proposal.

A. Implementation

Implemented using C#, our prototype is composed of 4 components as shown in Figure 5:

1) **Profile generator**: is used to generate random social network profiles with different or similar attributes' values using the FOAF vocabulary. To simplify this process, a "word generator" is used to generate from a small set of words, random words with a similarity measure higher than a chosen threshold. When generating a dataset of profiles, it is possible to define the percentage of the:
   - Profiles created with the same IFP value
   - Similar profiles referring to the same user but having different IFPs
   - Number of common attributes between two similar profiles
2) **Profile retriever**: is used to extract profiles having the same IFP value from the initial set of profiles. This can be done using a *smusher*[12] or by accessing a dataset of profile provided locally. It is important to note that crawling profiles from social network is a difficult task due to social site protection policy.
3) **Weight assignment**: is used to assign manually or automatically each attribute in the user profile to a weight as indicated in the Section III-C.
4) **Profile matcher**: returns the decision whether the two compared profiles are the same or not. This decision, done via a decision making algorithm, is computed using the weighted similarity scores.

[12]http://lists.w3.org/Archives/Public/www-rdf-interest/2000Dec/0191.html

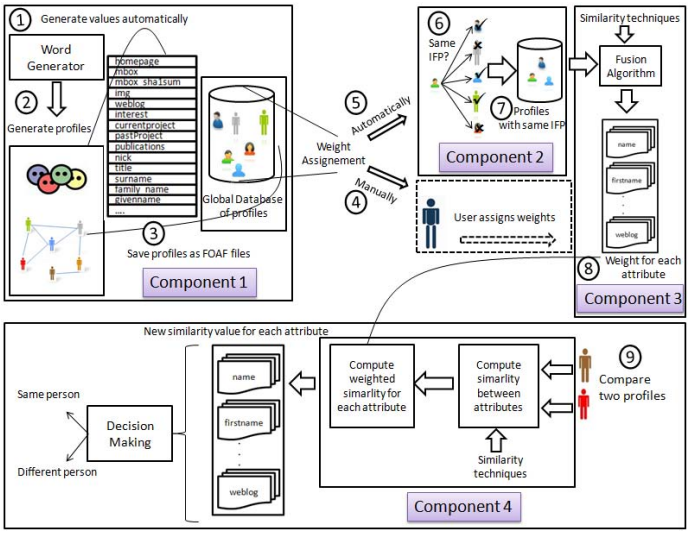

Fig. 5. Prototype Architecture

B. Experiments

1) Context: To conduct our experiments, we created 3 datasets of user profiles with FOAF attributes. The values of the attributes have been generated automatically by the "word generator". From a predefined set of words further words have been generated. The similarity threshold was set at 0.8 to obtain sets of similar words. To obtain profiles representing different persons, the set of predefined words was different for each of the 3 datasets. We created a set of 50 profiles that represent different user-profiles related to 3 different physical persons. Those profiles were divided into 3 sets (Set 1: 25 profiles, Set 2: 15 profiles, and Set 3: 10 profiles). Each set was generated randomly and contained profiles that:

- have the same *foaf:mbox_sha1sum* value,
- have different *foaf:mbox_sha1sum* values but represent the same real person.

In the following experiments, 20% of the generated profiles, called *set R*, represent the same physical person but with different IFP values. All the conducted tests have been performed on 2.8 GHz Intel Centrino machine, 4GB RAM.

*2) **Relevance of the Proposed Approach**:* The aim of this experiment is to show that our proposed method can find profiles that refer to the same physical person more than the existing methods. The existing methods are based on the IFP to do the matching. We measured the efficiency of our approach

by varying from 0 to 100% the percentage of attributes having similar values between two profiles. We compared the results between the IFP based method and our approach and we obtained the following results summarized in Table II.

TABLE II
RESULTS SHOWING THE DIFFERENCE IN THE NUMBER OF MATCHED PROFILES THAT REFER TO THE SAME USER USING THE IFP BASED METHOD AND THEN OUR PROPOSED METHOD. THE PERCENTAGE OF ATTRIBUTES HAVING DIFFERENT VALUES IN SET R IS BEING VARIED.

% of attributes having different values	Total number of combinations	Number of detected profiles using			
		IFP	Proposed		
			All	$\in$ R	$\notin$ R
0	46	32	53	46	7
10	45	31	44	44	1
20	48	34	44	44	0
30	45	31	43	40	3
40	45	31	36	33	3
50	49	35	38	37	1
60	46	32	36	32	4
70	48	34	46	34	5
80	45	31	32	32	0
90	47	33	33	33	1
100	44	30	32	31	1

In this test, the percentage of attributes having different values between two profiles of the set R is known as *% of attributes having different value*. The *total number of possible combinations* is the result of different possible combinations inside a set of profiles. In our case, this set represents the number of combinations within the generated profiles that refer to the same physical person. For example, within a set of 4 profiles that refer to the same user, the total number of combinations that can be found is 6. Using the IFP method and then our proposed approach, we searched for the total number of possible combinations that refer to the same physical person. Then we calculated the number of combinations of found profiles by our method that also exist in the initial set R. We also calculated the number of profiles combinations that were detected by our approach as being the same physical person. Those profiles, part of the randomly generated profiles, are not part of the set R. The obtained results show that:

- We were able to detect a bigger number of profiles that refer to the same user by using our approach.
- We were able to detect some of the profiles that we generated that they refer to users with different IFPs. Here, we note that we were not able to detect all the profiles all the time (comparing the second and the fifth column of the table), and also that we detected in some cases false positive results (comparing the fourth column with the sum of the last two column). This can be explained by the fact that some users may have similar profiles but in reality they are different physical persons.
- The highest number of correctly detected profiles corresponds to profiles with a low percentage of attributes with different values. The lowest number of detected profiles correspond to profiles that have a high percentage of attributes having different values. Here, the results are very similar to the ones yielded by the existing methods.
- As the number of detected profiles decreases, the percentage of attribute with different values increases.

*3) **Impact of assigning weights to attributes**:* We conducted the following experiment in order to have a clearer estimation of the benefits of assigning weights to attributes. For that, we divided this experiment into two parts:

1) Applying our approach by granting all the attributes a weight of 1 which is called *attributes with same weight* (granting a weight of 1 to all attributes is dealing with all the attributes as having a similar importance)
2) Granting each attribute a different weight

We conducted this experiment and we obtained the following results shown in Table III. The obtained results show that:

TABLE III
RESULTS SHOWING THE POTENTIAL BENEFIT OF GIVING WEIGHTS TO EACH ATTRIBUTE. COMPARISON IN THE NUMBER OF FOUND PROFILES IS PRESENTED. THE PERCENTAGE OF ATTRIBUTES HAVING DIFFERENT VALUES IN SET R IS BEING INCREASED FOR EACH TEST.

Attributes with different %per profile	Total number of combinations	Number of detected profiles with			
		$\neq$ weight	Same weight		
			All	$\in$ R	$\notin$ R
0	46	46	394	46	380
10	45	44	347	45	327
20	48	44	378	45	363
30	45	43	540	45	510
40	45	36	413	45	397
50	49	38	471	45	448
60	46	36	387	46	371
70	46	46	413	46	390
80	45	32	421	46	401
90	47	33	373	47	359
100	44	32	411	46	395

- When all the attributes were granted a weight of 1, the number of the detected profiles combinations was a lot greater than the real total number of profile combinations.
- Without measuring the importance of each attribute we were able to detect the profiles that represent same users with different IFP. In this case, the main drawback was that the result also included a big number of false positive matched profiles (see the fourth and the last column).
- When the weight assignment was used the obtained results were adequate (see the third column).

We can conclude that when weights were assigned to each attribute, the detection of the profiles that correspond to the same physical users was more efficient and more reliable.

*4) **Different decision making algorithms**:* To more formally evaluate the benefit and the effectiveness of the decision making algorithms, we undertook a series of experiments to measure the potential benefits and reliability of each algorithm. We varied the number of attributes having different values and we computed the precision and the recall measures as follows:

$$Precision = \frac{Number\ of\ Found\ and\ Correct\ profiles\ matches}{Total\ Number\ of\ profiles\ found}$$
$$Recall = \frac{Number\ of\ Found\ and\ Correct\ profiles\ matches}{Total\ Number\ of\ correct\ profiles\ matches}$$

Five methods, mentioned in Section III, participated in this experiments: DS, BN, Avg, Min, and Max. In this test,

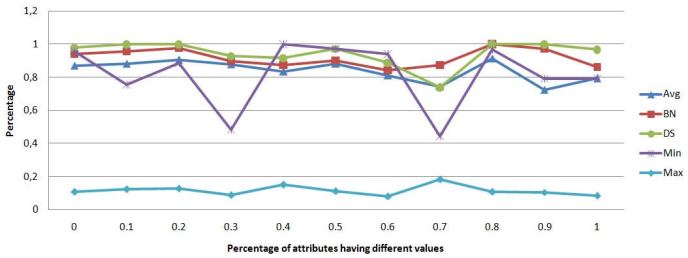

Fig. 6. Precision percentage while varying the number of attributes having different values

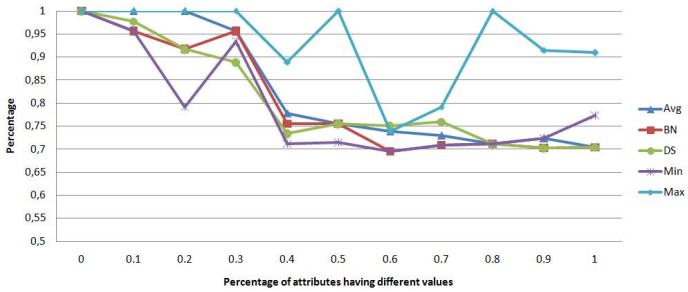

Fig. 7. Recall percentage while varying the number of attributes having different values

our interest is to measure the performance achieved by each algorithm on the decision making level (this doesn't concern any performance on the fusion level). As we can see in Figure 6, the precision of *DS* was the best followed by the one of *BN*. The precision of the *Avg* was acceptable meanwhile the *Min* showed a constant variation. The *Max* had the lowest precision and the highest recall as shown in Figure 7. The DS method had a good recall percentage higher than the remaining methods (*BN*, *Min*, and *Avg*). Finally, *DS* method was chosen for two main reasons: 1) Reliability: Most of the detected profiles were relevant 2) Completeness: Most of the previously generated relevant profiles were detected.

V. Conclusion

In this paper, we addressed the issue of providing inter-social network operations and functionalities. In this work, we proposed a framework for user profile matching in social networks. This framework is able to discover the biggest possible number of profiles that refer to the same physical user that existing approaches are unable to detect. In our work, attributes describing social network profiles were assigned weights manually or automatically, string and semantic similarity metrics were used to compare attribute values. Aggregation functions were used for data fusion and for decision making. We have also developed a prototype that was used to conduct the experimentations. The results of the experimentations showed improvements compared to other classical methods. As a future work, we are planning to further explore and propose more interesting inter-social operations and functionalities.

References

[1] C. Zhou, H. Chen, and T. Yu, "Learning a probabilistic semantic model from heterogeneous social networks for relationship identification," in *Tools with Artificial Intelligence, IEEE International Conference on*, vol. 1. Los Alamitos, CA, USA: IEEE Computer Society, 2008, pp. 343–350.

[2] L. Ding, L. Zhou, T. Finin, and A. Joshi, "How the semantic web is being used: An analysis of FOAF documents," in *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, Big Island, HI, USA, 2005, pp. 113c–113c.

[3] P. Mika, "Flink: Semantic web technology for the extraction and analysis of social networks," *JOURNAL OF WEB SEMANTICS*, vol. 3, pp. 211—223, 2005.

[4] J. Golbeck and M. Rothstein, "Linking social networks on the web with FOAF: a semantic web case study," in *Proceedings of the 23rd national conference on Artificial intelligence - Volume 2*. Chicago, Illinois: AAAI Press, 2008, pp. 1138–1143.

[5] A. Hogan, "The ExpertFinder corpus 2007 for the benchmarking and development of Expert-Finding systems," in *First International ExpertFinder Workshop*, 2007.

[6] S. Bortoli, H. Stoermer, P. Bouquet, and H. Wache, "Foaf-o-matic - solving the identity problem in the foaf network," in *In Proceedings of the Fourth Italian Semantic Web Workshop (SWAP2007)*, 2007.

[7] M. Rowe and F. Ciravegna, "Disambiguating identity through social circles and social data," in *Collective Intelligence Workshop ESWC 2008*, Tenerife, Spain, 2008.

[8] L. Shi, D. Berrueta, S. Fernndez, L. Polo, S. Fernndez, and A. Asturias, "Smushing RDF instances: are alice and bob the same open source developer?" in *ISWC2008 workshop on Personal Identification and Collaborations: Knowledge Mediation and Extraction (PICKME 2008)*, 2009.

[9] P. Bouquet, H. Stoermer, and B. Bazzanella, "An entity name system (ENS) for the semantic web," in *The Semantic Web: Research and Applications*, 2008, pp. 258–272.

[10] J. J. Jiang and D. W. Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy," *International Conference Research on Computational Linguistics (ROCLING X)*, Sep. 1997.

[11] A. ElSayed, H. Hacid, and D.-A. Zighed, "A multisource context-dependent semantic distance between concepts," in *18th International Conference on Database and Expert Systems Applications (DEXA 07), Regensburg, Germany*, ser. Lecture Notes in Computer Science, R. Wagner, N. Revell, and G. Pernul, Eds., vol. 4653. Springer, September 2007, pp. 54–63.

[12] W. Cohen, P. Ravikumar, and S. Fienberg, "A comparison of string distance metrics for name-matching tasks," 2003.

[13] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg, "Adaptive name matching in information integration," *IEEE Intelligent Systems*, vol. 18, no. 5, pp. 16–23, 2003.

[14] M. A. Jaro, "Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida," *Journal of the American Statistical Association*, vol. 84, no. 406, pp. 414–420, Jun. 1989.

[15] W. W. Cohen and E. Minkov, "A graph-search framework for associating gene identifiers with documents," *BMC Bioinformatics*, vol. 7, p. 440, 2006.

[16] A. Bilke and F. Naumann, "Schema matching using duplicates," in *21st International Conference on Data Engineering (ICDE'05)*, Tokyo, Japan, 2005, pp. 69–80.

[17] E. Gabrilovich and S. Markovitch, "Computing semantic relatedness using wikipedia-based explicit semantic analysis," in *Proceedings of the 20th international joint conference on Artifical intelligence*. Hyderabad, India: Morgan Kaufmann Publishers Inc., 2007, pp. 1606–1611.

[18] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Soviet Physics Doklady*, vol. 10, p. 707, Feb. 1966.

[19] S. L. Hegarat-Mascle, I. Bloch, and D. Vidal-Madjar, "Application of dempster-shafer evidence theory to unsupervised classification in multisource remote sensing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 35, pp. 1018–1031, 1997.

[20] R. R. Yager, "Human behavioral modeling using fuzzy and DempsterShafer theory," in *Social Computing, Behavioral Modeling, and Prediction*, 2008, pp. 89–99.

[21] E. Raad, B. A. Bouna, and R. Chbeir, "Bridging sensing and decision making in ambient intelligence environments," in *Multimedia Techniques for Device and Ambient Intelligence*, 2009, pp. 135–164.