ASP–SSN: AN EFFECTIVE APPROACH FOR LINKING SEMANTIC SOCIAL NETWORKS

Sanaa Kaddoura[1] and Islam Elkabani[2]

[1] Department of Mathematics and Computer Science, Faculty of Science, Beirut Arab University, Lebanon
sanakadoura@hotmail.com
[2] Department of Mathematics and Computer Science, Faculty of Science, Alexandria University, Egypt
elkabani@gmail.com

ABSTRACT

The dramatic increase of social networking sites forced web users to duplicate their identity on many of them. But, the lack of interoperability and linkage between these social networks allowed users' information to be disseminated within walled garden data islands. Achieving interoperability will contribute to the creation of rich knowledge base that can be used for querying social networks and discovering some facts about social connections. This paper presents a new approach for linking semantic social networks (SSN). This approach is based on the Answer Set Programming (ASP) Paradigm and Fuzzy Logic. An ASP-SNN reasoner is developed using the DLV answer set solver and tested on data sets exported from seven different semantic social networks. Fuzzy logic is used to assign a degree of truth to every discovered link. The proposed approach is simple, generic and intuitive.

KEYWORDS

Semantic Web, Social Networks, Answer Set Programming, Fuzzy Logic, Ontologies

1. INTRODUCTION

Semantic web, the extension of the current web, is under development in its layered approach where the OWL [1], Web Ontology Language, layer is the highest mature layer [2]. On the top of OWL layer, logic layer is under development to combine rules with ontologies. The importance of the logic layer is to enable intelligent reasoning on meaningful and interlinked data. Interlinking data is an important issue for semantic web, the web of data; because it is not only about just putting data on the web, but also allowing machines to discover other related knowledge [3].

Social networks can be represented in semantic web formats. The RDF-based Friend of A Friend (FOAF) project is used to represent social data. RDF stands for Resource Description Framework and it is the basic building block for Semantic Web. It is an XML-based language that uses graph

data format to represent data. FOAF allows machine readable web pages and provides a vocabulary to represent personal information and relation between people.

With the increase of social networks, people started to duplicate their identity on many of these networks. The social identity of people was fragmented on many social sites. However, these sites act like walled garden data islands because they lack the interoperability between them. In order to link various social networks, people who have duplicate identities can be exploited. Since people do not always provide their complete personal information in one social media and in the absence of a global unique identifier for each person, the problem of identity ambiguity arises. Hence, a method that identifies that two profiles refer to the same real world person instance is definitely needed.

This paper presents a contribution to the efforts for interlinking data on social networks. We present a new approach to link social data in the absence of complete knowledge. This approach exploits people who duplicated their identity on different social networks and uses them as a bridge between two social islands. If a person X on the first social network was identified to be the same person Y on the second social network, an identity link will connect the two networks. In the absence of a global unique identifier that identifies people on the social media, the proposed method utilizes the personal information provided by the users in order to cluster person instances that refer to the same real world person. Some links can be deterministically established when the available data about the user is enough to identify duplicated profiles. However, in some cases the lack of knowledge about users will cause that only few links might be explored between networks. In order to increase the number of links and interoperability over networks, the new approach suggests non-deterministic links based on fuzzy logic. This method exploits all the information available in the profiles to cluster person instances that refer to the same real world person. Then, it sets a degree of truth to the established link according to the availability of matching personal information in the two FOAF files.

In this work, we have implemented an Answer Set-based reasoner. The reasoner searches for identity links on different degrees of truth. It is tested on social data exported in a semantic web format from different social media. The ASP-SSN reasoner is characterized by being generic. It can be used to interlink semantic data other than the social one. It can be adjusted to solve the identity disambiguation problem outside the social networks too. Given the characteristics of any set of resources on the semantic web, the reasoner is able to discover identity links and interlink these resources according to the suggested reasoning approach. Moreover, the reasoner is based on a simple and intuitive reasoning process. It utilizes all the available personal information provided by the user in order to help in inferring identity links. The high expressive power and simplicity of answer set programming facilitated the knowledge representation during the implementation of the reasoner.

2. RELATED WORK

The identity disambiguation problem directed many researchers to search for the presence of the same person on many online communities [4, 5, 6, 7]. In this section, we relate our work to the previous work in this domain. We discuss the different suggested approaches and compare them with the approach presented in this paper.

In [6], a graph matching paradigm is used to infer identity links: node/edge overlap, node mapping and graph reasoning. In node/edge method, the number of edit operations was counted to perform the transformation from one graph to another, and then normalized by summing the node and edge counts from each graph. While in the node mapping method, a list of all possible node mapping combinations between two graphs along with the similarity measure of the two nodes was created. The graph reasoning method followed the hypothetical assumption that the owner of several social network accounts can never be a friend with two or more different people who share the same name and live in the same place.

The authors discussed in [5] four alternatives of *owl:sameAs*. They considered the *owl:sameAs* just one type of identity links and provided a similarity ontology that includes, in addition to identity links, similarity links between users. The new links that they introduced were: *Identical But Referentially Opaque*, *Match*, *Similar* and *Related*. *Identical But Referentially Opaque* links were used when a person had two names but the properties associated with this person depends on which name was used. *Match* links were used when two people share enough properties to replace each other. *Similar* links indicated that two people share some but not all properties. Finally, the *Related* links were used when two people did not share any property but were nonetheless closely aligned in some fashion.

In [7], the authors aimed to discover as much redundant instances as possible. They described two approaches. First, they exploited the OWL inverse functional properties and restricted this approach to one inverse functional property which is the email address. Second, they relied on comparing personal names and restricted their comparison to exact name match. Both approaches were implemented using SPARQL (a Semantic Web Query Language).

Finally, in [4], the authors used an OWL reasoner that compares inverse functional properties of users. When there was an exact match between two properties, an identity link was inferred between the two users. They also presented results on the impact of these links on the structural properties of the unified network.

In this research, we adopt some of the aforementioned techniques. Nevertheless, we extend these techniques and build our own reasoner, ASP-SSN, which discover links between different social networks. The ASP-SSN reasoner does not use any graph matching or makes any friends comparison as in [6]. Instead it takes advantage of the personal data available in the FOAF files regardless of the friends that a user may have. The ASP-SSN reasoner exploits all the available inverse functional properties in order to infer that two users are identical. In addition, it compares FOAF links that are provided in the users' profiles. These links indicate whether the user has another FOAF file hosted in another network or not. Moreover, our reasoner infers identity links by applying the transitive property. The reasoner will not only infer identity links; it adds another fuzzy links with certain degree of membership. Instead of assigning a name to these new links as in [5], it assigns a degree of membership to the links indicating that two people are identical to a certain degree of truth. Some membership degrees are based on comparing the names of the users. In [7], they compared names but they were restricted to exact name matching. In our approach, we consider the names that are similar to a certain extent and reinforce this comparison by comparing the date of birth and the country of a user too.

2. PRELIMINARIES

2.1. Semantic Social Networks

Social networks refer to the latest generation of interactive online services like weblogs, wikis, forums and instant messaging [8]. Social networks are not a tool for wasting time anymore; they have become a tool for business. The number of users of social networking sites has dramatically increased recently. This fact is supported by statistical results from some ranking sites. For example, according to EmarketinGuide [9], Twitter is adding nearly 500,000 users per day. Statistics showed that the number of users of Twitter increased from 50 million in 2008 to 200 million in 2011. In addition, Facebook users reached 750 million by the end of 2011; however, this number was 200 million in 2008. According to Alexa [10], Facebook is ranked the 2nd on the list of top sites globally, whereas Twitter is ranked the 10th and Linkedin is ranked the 16th. Moreover, EmarketinGuide claims that Google+ has more than 25 million users and it was the fastest website to reach 10 million users within 16 days only. Twitter spent 780 days to reach 10 million users and Facebook reached this number after 852 days.

Semantic Social Networks (SSN) are those social networks that use semantic web technologies. Integrating social networks and semantic web facilitates the retrieval of users' profiles from different social networks. Semantic social networks publish FOAF files of the registered users. These FOAF files hold the information of the users that they provided on the network in an XML format. The next section will discuss FOAF files and provides a sample file.

2.2. Friend Of A Friend Ontology (FOAF)

FOAF project [11], initiated by Dan Brickley and Libby Miller in 2000, is the first formal and machine processable representation of user profiles and friendship. A FOAF file is written in RDF format. RDF, stands for Resource Description Framework, is the basic building block for Semantic Web. It is an XML-based language that uses graph data and provides machine readable syntax for the web pages. FOAF is an agreed-upon semantic web format that describes people and the connections that bind them. The *foaf:Person* class is the core of the FOAF vocabulary. It describes the person's identity and lists all the attributes related to him like name, gender and mailbox. Each friend that a person adds to his profile is also an instance of *foaf:Person* class and can be related to that person by *foaf:knows* relation. Using people as a bridge, FOAF can be used to link many social data on the web and study online communities and the emergence of new ones. Inverse Functional Properties (IFPs) are special properties of the FOAF ontology. They act as unique identifiers for the user. The IFPs that are available in the FOAF ontology are shown in Table 1.

Table 1. FOAF IFPs

IFP	Description
foaf:mbox	It is used to specify the email address of the user in a plain text.
foaf:mboxsha1sum	It publishes a hash for the user's email address without revealing the address to spammers.
foaf:homepage	It is used to specify the homepage of the user.
foaf:weblog	It indicates the blogs of the users.
foaf:openid	It is used to specify the openid of the user.
foaf:icqChatID foaf:msnChatID foaf:yahooChatID foaf:aimChatID foaf:jabberID.	These properties provide the chat ids of the users in a FOAF file.

The following is a snippet of a FOAF file that was exported from a social network and Figure 1 shows its graphical representation:

```
<foaf:Person rdf:ID="#me">
              <foaf:name>Sanaa Kaddoura</foaf:name>
              <foaf:mbox_sha1sum>c7145a9f68570490517eb90a0d11361b39f53c82
              </foaf:mbox_sha1sum>
            <foaf:knows>
                <foaf:Person>
                    <foaf:name>veiled_angel</foaf:name>
                </foaf:Person>
            </foaf:knows>
</foaf:Person>
```

Figure 1. Graphical Representation of FOAF File

2.3. Answer Set Programming Paradigm

Answer set programming (ASP) is a logic programming paradigm whose underlying theoretical foundation is the answer set semantics (a.k.a stable model semantics) introduced by Gelfond and Lifschitz (1988) [12]. In ASP, problems are encoded as logic programs; the corresponding answer sets of the logic programs provide solutions to the original problems. The ASP paradigm is most widely used with the formalism of logic programming without function symbols, with programs interpreted by the stable model semantics [13]. Many applications like non-monotonic reasoning, reasoning about actions, declarative problem solving, can be expressed by logic programs, and then solved by an ASP system. Unlike Prolog-like logic programming, ASP is fully declarative. Neither the order of rules in a program nor the order of literals in rules has any effect on the semantics and only negligible (if any) effect on the computation [13].

In ASP, each solution to a problem is represented by an Answer Set of a deductive database logic program encoding the problem itself. These deductive databases do not include any function except the aggregate functions. ASP is clearly related to deductive databases and knowledge bases, where the occurrence of several answer sets indicates the presence of uncertain or incomplete knowledge, and each answer set represents a possible plausible instance of the database/knowledge base. The problem will be encoded as set of facts, also called Extensional Databases (EDB), and rules, also called Intensional Databases (IDB) [14]. It represents a given problem by a logic program and use answer set solver, such as DLV, to find answer sets for this program. These answer sets represent solutions to the given problem. Figure 2 shows the strategy of processing of ASP programs:

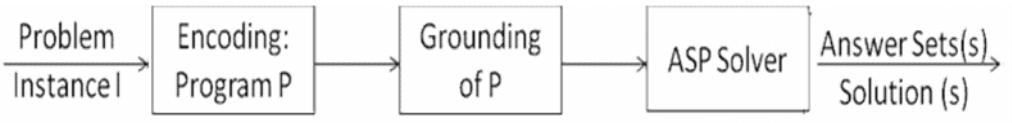

Figure 2. Problem Solving in ASP

After encoding the problem, processing of programs in ASP is most often done in two steps. The first step consists of grounding the program to its equivalent propositional version. In the second step, this propositional program is solved by a backtracking search algorithm that finds one or more of its answer sets (they represent solutions) or determines that no answer sets (solutions) exist [13].

The current software tools employed in each step, commonly referred to as grounders and solvers, respectively, have already reached the level of performance that makes it possible to use them successfully with programs arising from problems of practical importance. One of these software tools which includes a grounder and a solver is the *DLV*.

2.3.1. DLV

DLV stands for Datalog with Disjunction [14]. It is a deductive database system, based on disjunctive logic programming, which offers front-ends to several advanced knowledge representation formalisms. DLV can also be described as a solver for ASP. One of the main strengths of the system is its wide applicability, from more “database oriented” deductive

database applications (where larger input data have to be dealt with), to hard search and optimization problems. For more details about DLV, [14] discusses in details the DLV system and its features.

2.4. Fuzzy Logic

Fuzzy logic is a superset of Boolean logic. It has been introduced in 1960's by Dr. Lotfe Zadeh, a professor at the University of California at Berkley [15]. It has been extended to handle the concept of partial truth (i.e. a truth value between absolute truth and absolute falsity).

A fuzzy subset *F* of a set *S* can be defined as a set of ordered pairs, each with the first element from *S*, and the second element from the interval *[0,1]*, with exactly one ordered pair present for each element of *S* [15]. This defines a mapping between elements of the set *S* and values in the interval *[0, 1]*. The value *0* is used to represent complete non-membership (absolute falsity), the value *1* is used to represent complete membership (absolute truth), and values in between are used to represent intermediate Degrees of Membership or Degrees of Truth. The mapping is described as a function called the Membership Function of *F*.

3. ASP–SSN REASONER

3.1. Reasoning Methodology

The ASP–SSN system is an ASP based system whose purpose is to link identical people with duplicate identities on different or on the same semantic social networks. The input of the system is a set of *foaf:Person* instances collected by the crawler discussed later in section 4. The system will produce links between the networks. Figure 3 represents the system in terms of input and output. The detailed architecture will be discussed later in Section 3.2.

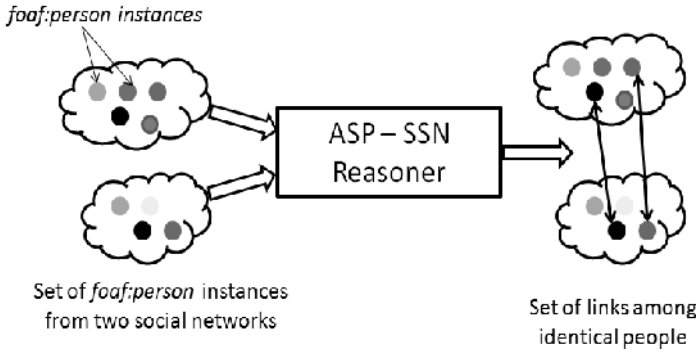

Figure 3. Overview of the System

The data collected by the crawler represents all the available information about a user on some social network. The reasoner exploits these data to cluster instances that refer to the same person in real world. Two types of links are produced: Deterministic Links and links inferred based on the fuzzy measure. These two types of links will be discussed in details in the following subsections.

3.1.1. Absolute Truth – Deterministic Links

The absolute truth is the highest degree of truth. This link is called a deterministic link because the method used by the reasoner succeeds in determining that two persons are identical. First, a comparison occurs between the inverse functional properties of each pair of *foaf:Person* instances. Second, a comparison occurs between other FOAF links that a person may have on other social communities. Finally, other links are inferred via transitivity and symmetry. A flow chart that shows the different stages of inferring a deterministic link is shown in Figure 4.

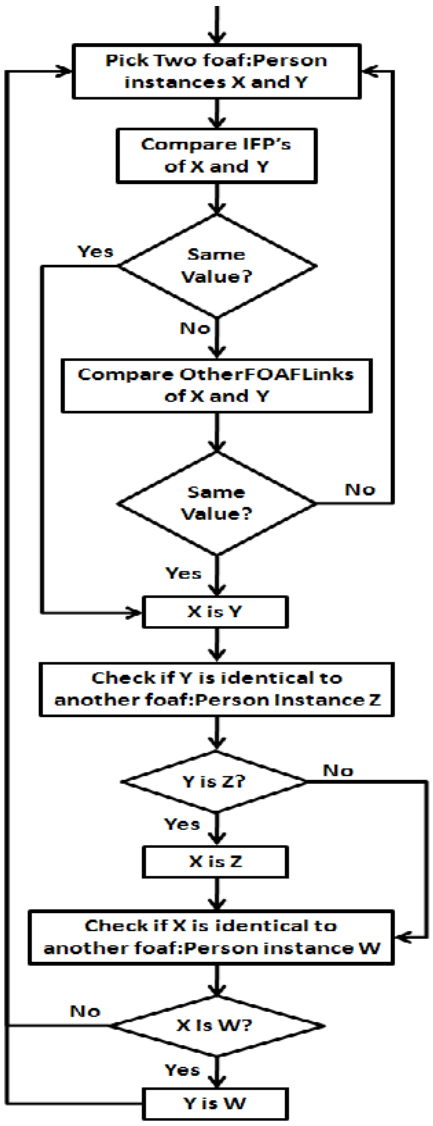

Figure 4. Absolute Truth Linking Process

A) Deterministic Linkage by Inverse Functional Properties

When two *foaf:Person* instances have the same value of any of the inverse functional properties listed in Table 1, they will be inferred to be the same person. The order that the IFPs will be compared in does not matter. For simplicity, since the email address was the most relevant property, it was chosen to be the first to compare by. Jabber ID is rare to be used, so it was chosen to be the last one to be compared among the chat IDs. Figure 5 shows a graphical representation of two person instances that have the same yahoo ID. The two instances have two different IDs but they represent the same person because they have the same value of an IFP.

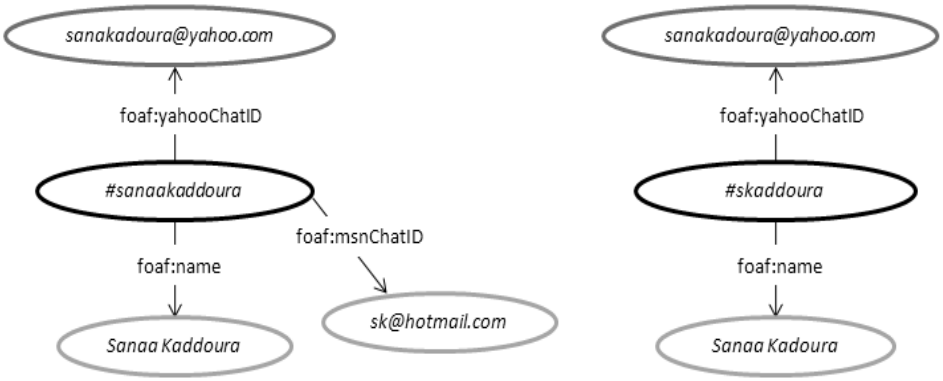

Figure 5. People instances have same yahooChatID

B) Deterministic Linkage using rdfs:seeAlso Tag

FOAF provides *rdfs:seeAlso* property that is used to tell that additional information about the resource is available in another FOAF file. This tag holds the URI of the other FOAF file. This method has its power in inferring deterministic identity links in the absence of any IFP value for privacy issues. Consider, for example, this person predicate from Ecademy:
person("Julian","e42b34b637b3c06d872e5b8938aad918b3bf345a,"http://www.ecademy.com /module.php?mod=network&op=foafrdf&uid=18670","http://www.livejournal.com/users/aldon/data/foaf ").

The data in this predicate are arranged as follows: nickname, encrypted email address, FOAF file link on the Ecademy network, *rdfs:seeAlso* link. This user states that he has a FOAF file with this link: http://www.livejournal.com/users/aldon/data/foaf. Upon reasoning between Ecademy and LiveJournal, the reasoner will compare the value of this *rdfs:seeAlso* tag with FOAF files' links on LiveJournal. Thus, when they match, the reasoner directly infers an identity link between the two networks through this user.

C) Deterministic Linkage by Transitive and Euclidean Relations

The reasoner can infer links even in the absence of matching IFPs. It takes advantage of the transitive and the Euclidean relations to infer links based on previously discovered ones. A transitive relation *R* can be defined as follows:

$$\forall a,b,c \in A : (aRb \wedge bRc) \Rightarrow aRc$$

and an Euclidean relation *R* can be defined as:

$$\forall a,b,c \in A : (aRb \wedge aRc) \Rightarrow bRc$$

Consider the following example that explains how the reasoner infers links based on the transitive relations. The following person predicates show three instances in a network with their associated usernames, yahoochatIDs and msnChatIDs.

person("13092", "iain_abg@yahoo.com","iain_1@hotmail.com").
person("5496", "iain_abg@yahoo.com","abi@hotmail.com").
person("23764", "Ia_Ab@yahoo.com"," abi@hotmail.com").

According to yahoochatID comparison, the reasoner will infer that profile id *13092* and profile id *5496* are identical. Based on msnChatID comparison, the reasoner will infer an identity link between profile id *5496* and profile id *23764*. There is no enough information to infer that the profiles *13092* and *23764* are identical since they have different values of IFPs. However, they are identical based on the transitive closure. So, the reasoner will infer an identity link between them.

A similar example can be used to show how the reasoner uses the Euclidean relation to infer links:

person("13092", "iain_abg@yahoo.com","iain_1@hotmail.com").
person("5496", "iain_abg@yahoo.com","abi@hotmail.com").
person("23764", "Ia_Ab@yahoo.com", "iain_1@hotmail.com").

Based on the yahoochatID comparison, the reasoner will link the profiles with ids *13092* and *5496*. Also, based on msnChatID, the reasoner will link *13092* and *23764* profiles. The given data is not enough to tell that the people whose ids are *5496* and *23764* are identical. However, according to the Euclidean relation, an identity link will inferred between these two people. The importance of transitive and Euclidean relations lies in inferring deterministic links when users do not provide enough information about themselves.

3.1.2. Fuzzy Measure

In some cases, the deterministic link cannot be inferred because of the absence of the identifiers for some privacy issues. Sometimes, by looking at two people profiles on a social network, one might instantly think that those profiles belong to the same person, even though there is no unique identifier available to rely on. For example, two people having the same name, same date of birth and lives in the same country, give evidence, to a high degree of truth, that they are the same person. In such case, a deterministic link cannot be inferred; however, we can say that these two people are identical to a certain degree of truth.

In other cases, two people may have same date of birth but their names are not identical, even though they are the same person. Consider two profiles carrying the same date of birth, but one of the names is "*J.C. O'Nazareth*" and the other is "*JC O'Nazareth*". These profiles should not be ignored because they show a level of identity. In this case, a lower degree of truth might be assigned to the link connecting these two people.

Fuzzy measure was created to set a degree of truth to the links between identical people in the absence of deterministic link. This fuzzy measure depends on the names of the users (real names not usernames), date of birth and the country. The date of birth is functional property i.e. the same person cannot have two different values of the date of birth on two different networks. The country, in this case, is a supporting detail in order to increase the degree of truth. A problem that may arise here is how the reasoner can identify similar names. Exact name matching is totally not sufficient because a person may write his name in different ways on different social networks. Thus, an algorithm that measure string similarity distance must be used.

Jaro-Winkler algorithm proved its efficiency in measuring the similarity distance between people names [16, 17]. It is an enhancement to Jaro algorithm that was introduced by Jaro in 1989 [17]. Jaro algorithm produces a similarity value for two strings, *A* and *B*, in the range [0, 1]. Let $A = (a_1 a_2 \ldots a_m)$ and $B = (b_1 b_2 \ldots b_n)$ be the two strings to be compared. The search range distance *d* is defined as follows:

$$d = \frac{max\ (|A|,|B|)}{2} - 1$$

where $|A|$ and $|B|$ represent the lengths of *A* and *B* respectively. So, *d* is less than half the length of the longer string. The character a_i is counted as in-common with character b_j iff $a_i = b_j$, and $i-d \leq j \leq i+d$. The value of the string comparator metric is further determined by a count of transpositions. Transpositions are determined by pairs of in-common characters that are out of order. Let $A' = (a'_1 a'_2 \ldots a'_m)$ and $B' = (b'_1 b'_2 \ldots b'_n)$ be the in-common characters for strings *A* and *B* respectively. A half transposition occurs for *A'* and *B'* at position *i* if $a'_i \neq b'_i$. Thus, the number of transpositions *t* is defined as:

$$t=0.5 * Number\ of\ half\ Transpositons$$

The Jaro comparator is defined as:

$$Jaro(A,B) = \frac{1}{3}\left(\frac{c}{m} + \frac{c}{n} + \frac{c-t}{c}\right)$$

where *c* is the set of in-common characters in strings *A* and *B* of lengths *m* and *n* respectively and *t* is the number of transpositions.

Jaro-Winkler uses Jaro's algorithm and adjusts the similarity value when up to first four prefixes of each string matches in the string pair being compared. Jaro-Winkler starts by computing the Jaro string distance, and then it compares the first four prefix characters of the two strings *p* as follows:

$$p = min\ (4, p')$$

where *p'* is the length of the longest prefix in the string pair that agrees exactly where an exact agreement is defined by $a_i = b_j$. The Jaro-Winkler String Comparator S_w is:

$$S_w = Jaro(A,B) + \frac{p\,(1-Jaro(A,B))}{10}$$

The number produced by this algorithm has no significance unless a threshold is identified. When the similarity is above or equal to a certain threshold, the names will be considered similar. The threshold is statistically estimated by using a sample from the collected datasets and it is 0.8. Thus, when two names has a similarity value of 0.8 or above, they will be classified as similar names.

3.1.2.1. Fuzzy Measure Symbolic Representation

Table 2 shows the symbolic representation of the fuzzy measure and its membership in the identity set. The interval [0, 1] was divided equally into 9 levels. Each level depends on the available properties. The highest level is the deterministic link. The other degrees of truth depend on name of person, date of birth and the country. The names of people either completely match so they take a high level of truth or match by a similarity value (Sim) that does not go beyond the threshold 0.8. The names' similarity values above the threshold were divided into three intervals: [0.8, 0.9[, [0.9, 1[and the exact name match (Sim(N1, N2) =1). When the country is not available or does not match, the degree of truth will decrease.

When a level of truth is inferred for a pair of *foaf:Person* instances, the process quits for this pair, picks another pair and starts to search for a degree of truth that suits it. The pair cannot be inferred twice on different levels of truth. The reasoner will assign levels of truth to the identical pairs as if a person is looking on the two profiles and trying to guess whether they belong to the same person or not. If any unique identifier matched, then, the two *foaf:Person* instances refer to the same person. If no unique identifiers are available, the reasoner will check whether specific personal information matches and assigns a lower level of truth. The reasoner keeps going down in the levels till it reaches the absolute falsity level and the pair is excluded from the identity set.

Table 2. The Membership of Each Fuzzy Level

Number	Symbolic Representation of the Measure	Membership in the Set
1	Nothing Matches	0 (Absolute Falsity)
2	Sim(N1,N2)=1	0.125
3	0.8≤Sim(N1,N2)<0.9, same DOB	0.25
4	0.9≤Sim(N1,N2)<1, same DOB	0.375
5	Sim(N1,N2)=1, same DOB	0.5
6	0.8≤Sim(N1,N2)<0.9, same DOB, same country	0.625
7	0.9≤Sim(N1,N2)<1, same DOB, same country	0.75
8	Sim(N1,N2)=1, same DOB, same country	0.875
9	Common IFPs	1 (Absolute Truth)

The membership function of this fuzzy measure is linear. As the available information decrease, the degree of membership decreases (see Table 2). Each level was given a number to simplify the graphical representation. Figure 6 shows a graphical representation of the membership function. It shows the degree of membership (or degree of truth) as a function of the Number of level according to Table 2.

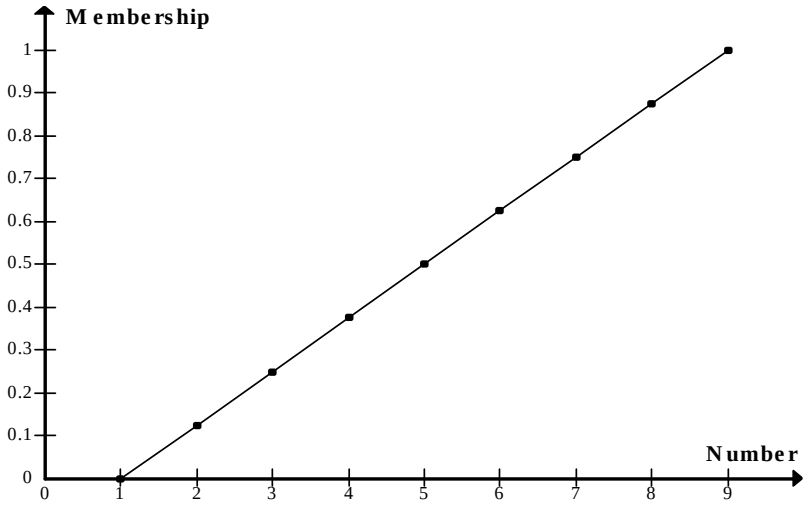

Figure 6. The Membership as a Function of the Number of the Level

3.2. ASP–SSN Reasoner Architecture

In order to cluster *foaf:Person* instances that belong to the same person, an Answer Set Based reasoner for semantic social networks was implemented. This reasoner is generic and can be applied to any two semantic social networks regardless of their context. The usage of ASP added simplicity to its architecture. The availability of off-the-shelf ASP solvers, such as the DLV, to compute the answer sets avoided us from building an ad-hoc solver. The ASP–SSN reasoner is implemented by using Java programming language. Microsoft SQL server is the underlying database management system used for storing the data. Figure 7 represents the overall architecture of the reasoner. The reasoner reads data from the databases that carry the data of the social networks and produces a set of identity pairs associated with their identity level.

As shown in the figure, the inputs of the reasoner are the relational databases of the social networks. The reasoner prompts the user for the names of the networks. After this step, no interference of the user is needed. The reasoner will pass along the stages of Figure 7 and produces text files that contain predicates of identical pairs and the degree of truth of these pairs.

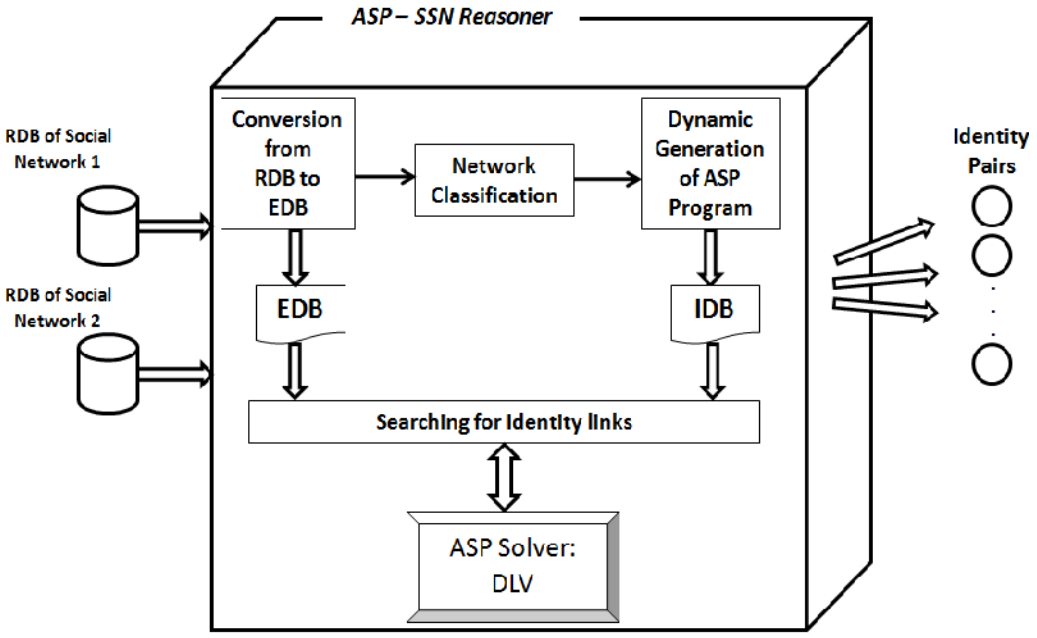

Figure 7. Architecture of the ASP-SSN Reasoner

The following is a description of the processes performed by the reasoner:

A) Conversion from RDB to EDB

This is the first process of the reasoner. It converts the relational database data (RDB) to extensional database (EDB). It produces a document of all *foaf:Person* instances that are present in the database in a form of person predicates. Each person predicate holds all the personal information of the person together with his person id:

Person(Id, Name, DateOfBirth, Country, Email, Hompage, MsnChatid, Other IFPs…).
The Data Object Model (DOM), which is supported by Java, helps in this stage of converting the FOAF files to a format that is accessible by DLV.

B) Classification of Networks

Once the conversion process ends, the classification process starts. The classification process classifies the network based on the available information about its users on the network. This will affect the next process of generating the ASP program. When information is available for at least one of the users, the network will be classified as having this information. In this case, a rule will be generated for it in the next process. If none of the users has the information, the network will be classified as not having it. For example, if a network does not provide the Chat Ids of the users, then no rules will be generated for the Chat Ids. This saves the time of the reasoner and prevents it from reasoning on something that will not produce any link.

C) Dynamic Generation of ASP Program

The ASP program is dynamically generated based on the class of the network. When the network is classified to which class it belongs, the process of generating the ASP program starts. It generates the IDB (or rules) to search among the previously generated person predicates in the EDB file. These rules differ from network to network. They are produced and saved in the file referred as *IDB* in the figure. Other than the reasoning rules, the IDB contains a rule that counts the inferred identity links. The number of links will also be produced by the reasoner.

D) Searching for Links

In this stage, the clustering of FOAF file takes place. To search for links, the reasoner sends the IDB and EDB files to DLV that in turn returns the identity link with its degree of truth.

3.3. Role of ASP in the ASP–SSN Reasoner

One of the important features of any intelligent system is its ability to perform logical inferences, make decisions and handle search problems over a large search space. These actions are simple to be implemented in declarative logic programming paradigms because such formalisms allow the developer of the system to focus on the reasoning that must be performed instead of being plunged in the details of how to implement them.

Another good feature of ASP is that it always terminates, which prevents the reasoner from falling in an infinite loop. In addition, the reasoner also takes benefit from transitivity closure supported by ASP. Many results were produced by transitive relations when no data is enough to infer deterministic link. Moreover, ASP supports recursion that is not supported by other formalisms such as relational databases. Furthermore, aggregates that are supported by ASP have their impact on the reasoner. The aggregate used here is the count. Instead of counting links manually or by another program, which will be implemented for this purpose, aggregates helped in doing this at the runtime of the reasoning stage. Below is a simple part of a program used in reasoning that shows these features:

```
DOT_1(X,Z):-DOT_1(X,Y),DOT_1(Y,Z),X<Z.                    (Line 1)
DOT_1(X,Y):-DOT_1(X,Z),DOT_1(Y,Z),X<Y.                    (Line 2)
countDegree_1 (N):- #count{X, Y: DOT_1(X, Y)} =N.         (Line 3)
```

The predicate called DOT_1 represents people profiles connected by deterministic link. As shown in Line 3, the count aggregate counts the links that were produced by the rules. In Lines 1 and 2, the recursion was clear when the predicate DOT_1, in the head, calls itself in the body of the rule. Line 1 shows the transitive closure for inferring identity link between X and Z through Y that is identical to both of them. The X<Y subgoal is added to avoid inferring a link between the person and itself. X!=Y subgoal does not work here because it will cause symmetric results to be inferred like DOT_1(“A”,”B”) and DOT_1(“B”,”A”).

Negation as failure played a role in the reasoner. Consider this rule from the reasoner:

sameMsn(X,Y) :- person(X,_,H), person(Y,_,H),
H != "NotAvailable", X<Y, not sameEmail(X, Y).

This rule is comparing MSN chat ids of the users (Note that the underscore "_" means any value). The negation as failure operator *not* is used to negate the last subgoal and preventing inferring the same link twice. If the *sameEmail(X,Y)* was not inferred yet, then, *sameMsn(X,Y)* will be added to the knowledge. ASP can handle large datasets and infer links in a reasonable time. This helped the reasoning process due to the huge dataset collected by the crawlers.

The expressive power of ASP, its features and the availability of off- the-shelf solvers such as DLV, made it a good choice for this reasoner.

4. EXPERIMENTS AND DISCUSSION

4.1. Datasources

Data sets are exported from seven different semantic social networks presented in Table 3. Boards, Ecademy, hi5 and Advogato provide a list of registered users, so all the available FOAF files are accessed. However, these FOAF files do not necessarily represent the whole network. In particular, when this study was conducted, Advogato had 12065 registered users but people with FOAF files were only 3607 since Advogato does not produce FOAF files for all users.

Table 3. The Collected Data Sets

Social Network	Collected Users
Advogato	3607
Boards	170,980
DeadJournal	59700
Ecademy	492,113
hi5	974,511
LiveJournal	26009
Twitter	9903

For social networks that do not provide list of registered users, the network was crawled in Breadth First Search fashion. The crawler starts with a list of seed URIs. A seed URI is a link to a FOAF file for a user on the network. As the crawler visits a link, it saves the data into a database and pulls all the friends URIs into a queue. Each time a URI is visited, it is added into a list that holds the visited URIs. The goal of this list is to prevent visiting a link twice.

The Breadth First Search (BFS) algorithm is chosen as the node selection algorithm based on the work described in [18, 19]. In both researches, it was found that BFS is reliable and tends to discover high level degrees nodes. This increases the number of links in the queue and guarantees the continuity of the crawling process.

Some links acts as black holes for crawlers. These black holes refer to the people who do not have any friends linked to them by *foaf:knows* relation. This prevents the crawler from out-leading to other nodes of the network. These links are also not reachable if a list of users was not provided by social networks.

4.2. Discovering Links between Social Networks

The ASP–SSN reasoner was experimented on the collected data sets described earlier in Section 4.1. The experiments were performed on a 3.4 GHz Dual Core equipped with 2 GB of RAM. Summarized results were presented in the tables below. All the presented results are logically correct. On any degree of truth, a link identifies that the two profiles shares something in common.

The results showed the number of links inferred between different social networks based on the different fuzzy levels. The reasons for the absence of some social networks at some fuzzy levels were either the degree of truth was not applicable for that network or there were no links inferred for the given dataset. For example, DeadJournal did not provide the names of the users. Thus, it was missing from the level 0.125 (see Table 10).

The results also showed that some people duplicated their identity in three or four different networks, particularly, 399 users have accounts on three different networks and 4 users have accounts on four different networks.

The small number of inferred links may be referred to the small number of collected users. Advogato, in particular, showed a deterministic link with every other network which indicates the importance of Advogato in such studies. This might imply that if Advogato produced more FOAF files, the number of inferred links might highly increase. This network is used by open source developers. Such people might have accounts on business networks like Ecademy or entertainment networks like LiveJournal. Advogato only provides the email address, weblog, homepage and the username. Thus, the degrees of truth between [0.25, 0.875] were not included in the reasoning process on the Advogato network.

Table 4 shows the results of reasoning at the absolute truth level (DOT=1). The *foaf:homepage* IFP was excluded from the reasoning process at the absolute truth level because it turned out to be misleading. Many users use the website of their work place or the Google page URL as their homepages. Because Twitter provides only the *foaf:homepage* IFP, it was excluded for the reasoning process at this level.

Indeed, the context of the network has a direct impact on the results. People usually prefer to hide their professional identity from their identity on other social networks. This was clear from the results of reasoning between Ecademy, Boards and hi5. Although the number of FOAF files collected from these communities was considerably large, the links between (Ecademy and hi5) and (Boards and hi5) are considered small.

The results of Table 4 showed that none of the Ecademy users had a duplicate identity on it, since each business must be centralized in only one single profile. It also showed that 617 links exists between Ecademy and Boards. It is normal to find links between business network like Ecademy

and advertisement network like Boards. A person who has a business will need an advertising community for promotion.

DeadJournal has an advantage on the absolute truth level because it shows the email, chat Ids, and weblogs of users. The results showed that DeadJournal has links with all other social networks including it.

Most of the used networks do not provide any data about the country of the user. This affected the result in the degrees of truth 0.875, 0.75 and 0.625. These levels showed results in LiveJournal only (see Tables 5 and 6). Boards, hi5 and LiveJournal provide the date of birth; links could be inferred between them on the levels 0.5, 0.375 and 0.25 (see Tables 7, 8 and 9). No links were inferred at DOT=0.75 level.

The results at DOT=0.125 are considered high (see Table 10). This is the level of people who has exact name match. By checking out the records, we discovered that many people have common names. For example, on hi5, 78 people carry the name “John Smith”. On Advogato, only 1 person carries this name. Thus, through this name only, there is a possibility of 78 links between hi5 and Advogato. Also, some people used fake names like “A B”. On hi5, 100 people used “A B” as their name. On Advogato, only 1 person carries this name. This gives a possibility of 100 links between Advogato and hi5. Using fake names increased the number of links at DOT=0.125 level.

Table 4. Linking Networks at DOT=1

	Advogato	**Boards**	**DeadJournal**	**Ecademy**	**hi5**	**LiveJournal**
Advogato	19	82	3	18	3	5
Boards	X	1292	11876	617	96	536
DeadJournal	X	X	10734	18	196	6694
Ecademy	X	X	X	0	258	0
hi5	X	X	X	X	168	1
LiveJournal	X	X	X	X	X	26

Table 5. Linking Networks at DOT=0.875

	LiveJournal
LiveJournal	5

Table 6. Linking Networks at DOT=0.625

	LiveJournal
LiveJournal	1

Table 7. Linking Networks at DOT=0.5

	Boards	**LiveJournal**
Boards	129	0
LiveJournal	X	1

Table 8. Linking Networks at DOT=0.375

	Boards	**LiveJournal**	**hi5**
Boards	38	0	0
LiveJournal	X	0	8
hi5	X	X	0

Table 9. Linking Networks at DOT=0.25

	Boards	**hi5**	**LiveJournal**
Boards	50	0	0
hi5	X	0	23
LiveJournal	X	X	1

Table 10. Linking Networks at DOT=0.125

	Advogato	**Boards**	**Ecademy**	**hi5**	**LiveJournal**	**Twitter**
Advogato	29	37	1475	1610	11	16
Boards	X	37	3494	5227	37504	1124
Ecademy	X	X	135281	176913	498	581
hi5	X	X	X	416443	1029	1614
LiveJournal	X	X	X	X	13203	616
Twitter	X	X	X	X	X	103

5. CONCLUSION

In this paper, we tackle the problem of integrating semantic social networks to facilitate interoperability between them by identifying the presence of the same person on different semantic social communities. Our new approach proposed an ASP–SSN reasoner, which was implemented for the purpose of inferring links with different degrees of truth. The data set is composed of FOAF files published by social networks and collected using ad-hoc crawlers. The reasoning process is intuitive. The reasoner uses the concept of fuzzy logic to set a degree of membership to the discovered identity links based on the available information in the FOAF files of pair of users. ASP-SSN reasoner is generic. The reasoning methodology can also be applied to semantic data other than social data. The high expressive power and simplicity of answer set programming facilitated the knowledge representation during the implementation of the reasoner.

6. FUTURE WORK

In the future work, we will study the relationship between people on social networks. We will exploit the collected datasets and the results produced out of this study to analyze the semantic social network connectivity based on the Homophily principle [20]. This principle states that people with similar characteristics tend to associate. We will study the friendship between users to see to which extent the principle of homophily affected this relationship. Do the friends connected with *foaf:knows* tag share same interests? Is the *foaf:knows* relation on social networks random or obeys homophily principle? Can we cluster people based on the similarity value between them? This project is currently under development.

REFERENCES

[1] (2004, Feb.). P. F. Patel-Schneider et al.: OWL Web Ontology Language: Semantics and Abstract Syntax. [Online]. Available: http://www.w3.org/TR/2004/REC-owl-semantics-20040210/

[2] P. Mika. Social Networks and the Semantic Web. New York: Springer, 2007.

[3] T. Finin et al., "Social networking on the semantic web", The Learning Organization: An International Journal, vol. 12, (5), pp.418 - 435, 2005.

[4] J. Golbeck and M. Rothstein, "Linking social networks on the web with FOAF," in 17th international conference on World Wide, 2008, pp. 1138-1143.

[5] H. Halpin and P.J. Hayes, "When owl:sameAs isn't the same: an analysis of identity links on the semantic web," in WWW Workshop on Linked Data on the Web, 2010, pp. 25-30.

[6] M. Rowe, "Interlinking distributed social graphs," in Linked Data on the Web Workshop, WWW09, 2009.

[7] L. Shi et al., "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), 2008.

[8] (2011, Jun.). K. Hampton et al.: social networking sites and our lives [Online]. Available: http://pewinternet.org/Reports/2011/Technology-and-social-networks.aspx

[9] (2011, Nov.). Ankush Kohli: Amazing Social Media Growth Statistics 2011 [Online]. Available: http://emarketinguide.com/2011/11/amazing-social-media-growth-statistics-2011/

[10] (2012, Jan.). Alexa Internet: Alexa Top 500 Global Sites [Online]. Available: http://www.alexa.com/topsites

[11] (2010, Jan.). D. Brickley and L. Miller: FOAF vocabulary specification [Online]. Available: http://xmlns.com/foaf/spec/20100101.html

[12] M. Gelfond and V. Lifschitz, "The stable model semantics for logic programs," in 5th International Conference on Logic Programming, 1988, pp. 1070-1080.

[13] V.W. Marek et al., "Origins of answer-set programming - Some Background And Two Personal Accounts". CoRR, vol. abs/1108.3281, 2011.

[14] N. Leone et al., "The DLV system for knowledge representation and reasoning," ACM Transactions on Computational Logic, vol. 7, (3), pp. 499-562, 2006.

[15] L. Zadeh, Fuzzy sets, fuzzy logic and fuzzy systems. USA: World Scientific Publishing Co., Inc., 1996.

[16] E. Cohen et al., "A Comparison of string metrics for matching names and Records," in KDD Workshop on Data Cleaning, Record Linkage, and Object Consolidation, 2003, pp. 73-78.

[17] D. Jyotsna and P.L. Choong, "Resolving partial name mentions using string metrics". Defence Science and Technology Organisation, Australia, Research Report. AR-014-065, Dec. 2007.

[18] M. Kurant et al., "On the bias of bfs," CoRR, vol. abs/1004.172, 2010.

[19] C. Wilson et al., "User interactions in social networks and their implications," in 4th ACM European conference on Computer systems, 2009.

[20] M. McPherson et al., "Birds of a feather: homophily in social networks," Annual Review of Sociology, vol. 27, (1): pp. 415-444, 2001.