8-2009

Ssnetviz: A visualization engine for heterogeneous semantic social networks

Ee Peng LIM
Singapore Management University, eplim@smu.edu.sg

Maureen MAUREEN
Nanyang Technological University

Nelman Lubis IBRAHIM

Aixin SUN
Nanyang Technological University

Anwitaman DATTA
Nanyang Technological University

See next page for additional authors

DOI: https://doi.org/10.1145/1593254.1593288

SSnetViz: A Visualization Engine for Heterogeneous Semantic Social Networks

Ee-Peng Lim
School of Information Systems
Singapore Management University
eplim@smu.edu.sg

Maureen
School of Computer Engineering
Nanyang Technological University, Singapore

Nelman Lubis Ibrahim
School of Computer Engineering
Nanyang Technological University, Singapore

Aixin Sun
School of Computer Engineering
Nanyang Technological University, Singapore

Anwitaman Datta
School of Computer Engineering
Nanyang Technological University, Singapore

Kuiyu Chang
School of Computer Engineering
Nanyang Technological University, Singapore

ABSTRACT

SSnetViz is an ongoing research to design and implement a visualization engine for heterogeneous semantic social networks. A semantic social network is a multi-modal network that contains nodes representing different types of people or object entities, and edges representing relationships among them. When multiple heterogeneous semantic social networks are to be visualized together, SSnetViz provides a suite of functions to store heterogeneous semantic social networks, to integrate them for searching and analysis. We will illustrate these functions using social networks related to terrorism research, one crafted by domain experts and another from Wikipedia.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Systems and Software—*information networks*

General Terms

Design

Keywords

Semantic social network, SSnetViz, social network exploration

1. INTRODUCTION

1.1 Motivation

ICEC '09, August 12-15, 2009, Taipei, Taiwan

Understanding how people and organizations are related in social structures is a major task of social network analysis[12]. In the past, most of the traditional social network analysis research has been carried out on manual constructed social networks and each analysis task usually involves one network at a time.

Today, with large number of users and organizations making their presence on Web and Web 2.0 sites, social network analysis increasingly involves network data that can be gathered from the Web either automatically or semi-automatically. These sites maintain different sets of attributes about the user and organization accounts, and different relationships among them. This motivates the study of *multi-modality* in networks. This paper thus focuses on semantic social network which consists of different types of nodes and links.

In addition to the above multi-modality issue, the same users and organizations may appear in different social networks due to their involvement in different Web sites (e.g., social networking sites). To study their relationships, one has to combine these different social networks and conduct analysis on the combined network. This requires the *integration* issues to be addressed in the analysis process.

1.2 Objectives

In this paper, we define a **semantic social network** to be a social network graph with multi-typed nodes and links. Semantic social networks, unlike the traditional social networks, have richer node and link type semantics. Instances of each node or link type are expected to share common set of attributes. For example, in a terrorist network, we may find terrorist and terrorist group node types and member-of(terrorist,terrorist group) and associated-with(terrorist group, terrorist group) link types. Every terrorist instance is expected to have attributes such as name, age, and background, etc. Every terrorist group instance is expected to have attributes such as group name, objectives, leadership, etc. Semantic social network is different from *multi-relational network* and *heterogeneous social network* which refer to social networks with single-typed nodes but multi-typed links[12]. In other words, semantic social network is a more general

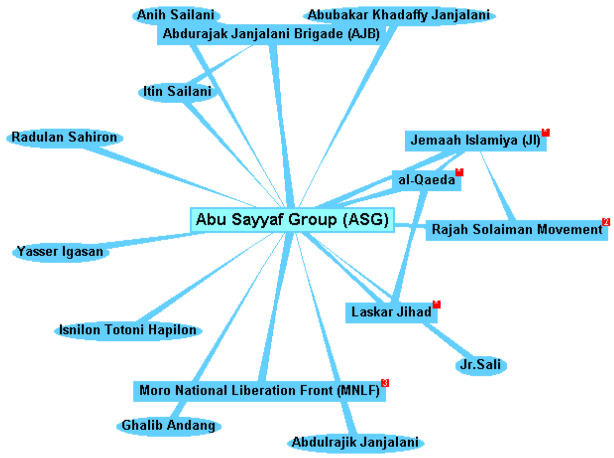


Figure 1: Abu Sayyaf and its neighbors in TKB network

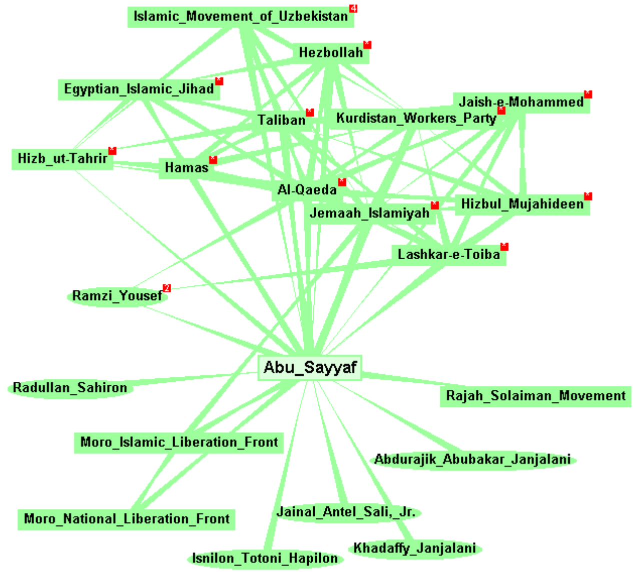


Figure 2: Abu Sayyaf and its neighbors in WikiTerrorism network

kind of network.

SSnetViz is an ongoing research to design and implement a visualization engine for heterogeneous semantic social networks. The main objectives of SSnetViz include the development of a set of general integration, visualization and search techniques to manage and analyze heterogeneous semantic social networks.

1.3 Motivating Example

To illustrate the requirements of SSnetViz, we describe two heterogeneous semantic social networks in the terrorism domain. The first is a network of terrorists and terrorist groups from the MIPT Terrorism Knowledge Base[1] (TKB). The network is constructed by experts funded by the United States Department of Homeland Security. TKB was made available on the Web at www.mipt.org before March 2008 and we crawled a copy of the dataset. Another is a terrorism network extracted from Wikipedia, consisting of also terrorists, terrorist groups and incidents. This network represents knowledge collaboratively edited by the online community. We call these two networks **TKB** and **WikiTerrorism networks** in this paper. The TKB network has 858 terrorist group nodes and 1463 terrorist nodes. For the WikiTerrorism network, we have identified 471 terrorist group nodes, 544 terrorist nodes and 1797 incident nodes, and more nodes (and links) are still in the process of extraction and verification.

We show a glimpse of the two networks in Figure 1 and Figure 2. The figures show the sub-networks containing "Abu Sayyaf" and its related terrorists and terrorist groups. For clarity, we do not show the incident nodes. The two sets of terrorist and terrorist group nodes and links involved are overlapping but not identical. 277 terrorist groups and 112 terrorists can be found in both the two networks. To allow users to perform analysis using the two networks, we would require a tool to *integrate* them and to support *search* and *visualization* on the integrated network.

Integration requirement. The integration of two networks essentially involves matching the nodes (i.e., terrorists and terrorist groups) that refer to the same real entities. This task is also known as **entity resolution**[8, 2]. Although nodes in a network can be uniquely identified by their node names, nodes representing the same real world entities may not have identical names. This is called the **synonym** problem[10]. For example, in Figures 1 and 2, there are nodes that represent the same real world terrorists and groups as shown in Table 1. There is also the **homonym** problem which refers to different nodes sharing the same names although this is rare.

[1]MIPT is the acronym of Memorial Institute for the Prevention of Terrorism.

Table 1: Matched Node Pairs

	TKB Network	WikiTerrorism Network
	Terrorist Group	**Terrorist Group**
1	Moro National Liberation Front (MNLF)	Moro_National_Liberation_Front
2	Abu Sayyaf Group (ASG)	Abu_Sayyaf
3	Jemaah Islamiya (JI)	Jemaah_Islamiyah
4	al-Qaeda	Al-Qaeda
5	Rajah Solaiman Movement	Rajah_Solaiman_Movement
	Terrorist	**Terrorist**
6	Isnilon Totoni Hapilon	Isnilon_Totoni_Hapilon
7	Radulan Sahiron	Radullan_Sahiron
8	Abubakar Khadaffy Janjalani	Khadaffy_Janjalani
9	Abdulrajik Janjalani	Abdurajik_Abubakar_Janjalani

Search requirement. Search is a basic function to locate the nodes of an integrated network that are interesting to the users. These nodes may also serve as the starting points to explore the network. In some cases, users may like to find associations between nodes so as to determine an interesting subset of network.

Visualization requirement. Network visualization is a powerful tool to analyze the integrated network using the human visual ability. Given an integrated network, users will need to visualize information from different data sources, and to explore them with the help of search functions.

SSnetViz is thus designed based on the above three requirements. For integration, SSnetViz supports both matching rule based and manual entity resolution techniques. We design the search function of SSnetViz to include both node and path search. The search function can be used together with the visualization capability so as to allow the user to switch between query and navigation approaches to explore the relevant subnetwork for a given network analysis task. In the following sections, we will describe the architecture of the SSnetViz system, its storage, functionality and user interface designs in greater detail.

1.4 Outline of Paper

The remainder of the paper is structured as follows. Section 2 gives a summary of some works closely related to SSnetViz. Section 3 describes the overall design of SSnetViz. The integration and search functions of SSnetViz are mentioned in Sections 4 and 5 respectively. In Section 6, we describe the export and import functions. We conclude the paper in Section 7.

2. RELATED WORK

Visualization of social network is a widely studied research topic. [3] provides a survey of different network visualization techniques for social networks. [7] gives a detailed description of different ways of visualization for ontologies in the form of networks. A survey of network visualization used in criminal network analysis is given in [13]. To view networks with uncluttered layout, [14] proposes a set of different network layout visualization.

Our work is similar to several other social network visualization projects. Vizster provides visualization functions to support exploration, search and analysis of online social networks [5]. OntoVis, a system also designed for visualizing large heterogeneous social networks[11], focuses on semantic and structural abstractions of networks so as to assist users in analyzing interesting sub-networks. C-Group is another visual analysis tool that analyzes focal user pairs and their dynamic group memberships in a social network which has groups formed by timestamped events[6].

Unlike the above works, SSnetViz does not provide abstraction functions. It however supports functions on integrating multiple networks, searching and graph export/import. By combining existing networks together, users will be able to assemble knowledge from different sources and analyze the structures and relationships of entities in combined networks easily.

3. DESIGN OF SSNETVIZ

3.1 Semantic Social Network Representation

We represent a semantic social network by a tuple $\langle T^N, T^L, G\rangle$ where T^N and T^L represent the set of node types and set of link types respectively. $G = (V, E)$ represents a network with V and E denoting the set of nodes and the set of links respectively. The type of a node v_i is denoted by $v_i.type$ ($\in T^N$) and the type of a link (v_i, v_j) is denoted by $(v_i, v_j).type$ ($\in T^L$). A node type $t_k \in T^N$ has attribute set $t_k.AttrSet$. Every node $v_i \in V$ with $v_i.type = t_k$ thus has an attribute $attr_l$ for each $attr_l \in t_k.AttrSet$, and a corresponding attribute value $v_i.attr_l$. Unlike nodes, the links in our semantic social network do not have attributes.

In Table 2, we summarize the main differences between semantic social network and the traditional social networks.

Table 2: Comparison of Semantic Social Network and Traditional Social Networks

Item	Semantic Social Network	Social Network
Type	Nodes/links are typed	Nodes/links are not typed
Attribute	Nodes have attributes	No attributes at all
Network source	Multiple	Single

To accommodate heterogeneous set of node attributes, SSnetViz devises the following two relational tables to register the set of attributes for each node type, and the data type of each attribute. The ATTRIBUTE table stores for each node type (belonging to some network source) the set of attribute names, their orders, and ids of their domains. The data type information of attributes, consisting of domain name, data type, minimum and maximum values, is stored in the DOMAIN table.

ATTRIBUTE(sourceid, nodetypeid, attributename, attributeorder, ismultivalued, domainid)

DOMAIN(domainid, domainname, datatype, minvalue, maxvalue)

3.2 System Architecture

The system architecture of SSnetViz is shown in Figure 3. It consists of several important system components including **network storage manager**, **network integrator**, **network visualizer**, **network search**, and **network export/import** modules. The last three modules exist within a single user interface application known as the **network explorer**.

SSnetViz is designed to manage multiple semantic social networks as shown in Figure 3. The network storage manager is responsible for storing the networks with different node and link types in a relational database. To support the heterogeneous network representations in network integration, visualization, search and other functions, the storage manager maintains a data dictionary of each semantic social network and allows other SSnetViz modules to access the data dictionary to determine the node types and attributes and link types of any given network. For example, the ATTRIBUTE and DOMAIN tables mentioned in Section 3.1 are part of this data dictionary.

Network integrator assists users to merge heterogeneous semantic social networks with overlapping network elements so as to derive a more complete view of the network data. The resultant integrated network, matching rules and other mapping data will also be managed by the network storage manager. In Figure 3, we use dash lines to represent the input and output of the integrator module while the solid lines represent the communication between the modules (since all network data are accessed via the network storage manager). We shall elaborate the network integration process further in Section 4.

The network visualization module provides a range of network browsing functions to explore and view the network interactively. Coupled with network search that supports both node and path searching, the two network exploration modules help users to identify the sub-networks relevant to the query or analysis tasks at hand through. Once a query-relevant sub-network is found, the network export/import module can further store it in GraphML format allowing

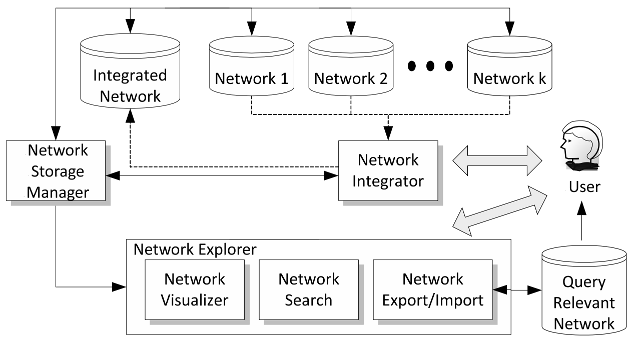


Figure 3: System Architecture of SSnetViz

the sub-network to be stored in non-proprietary standard format making it available to other GraphML compliant network visualizers. One can also import the same sub-network in GraphML format into SSnetViz in the future.

3.3 SSnetViz User Interface Design

SSnetViz has a user interface design that comprises a large central area for visualizing a semantic social network, a right panel to display information of a selected node and such information is divided into multiple tabs one for each component network in which the node exists, and a legend panel at the bottom (see Figure 4).

We use a distinct color for each component network or combination of networks. For example, while TKB and WikiTerrorism networks are shown in blue and green respectively, pink is used for nodes (and links) that are found in both TKB and WikiTerrorism. The color scheme can be modified by the user for better aesthetics. We also use distinct shapes for different node types. For example, terrorists and terrorist groups are displayed as ovals and rectangles respectively. The visualization functions that can be performed on the network include zooming, rotating and hyperbolizing. The link type is displayed as a popped up tooltip when the mouse pointer hovers over a link.

Once a semantic social network is selected, the user can select any node from the network for exploration. A node with links not displayed will be affixed with a red box at the top right corner showing the number of non-displayed outgoing links as shown in Figure 5. By expanding the node, these links and other nodes connected to them will be displayed. Right after a node is expanded, the user can examine the relevance of the newly added nodes and links. If they are found not relevant, the expanded node can be collapsed using the node collapse function. Unwanted nodes and links can also be removed by the node and link hiding functions. Some of the features described are provided by TouchGraph API[2].

4. NETWORK INTEGRATION

4.1 Integration Framework

The integration framework of SSnetViz consists of two main steps as shown in Figure 6, i.e., **node type matching** and **node matching**. The two steps can be performed

[2]http://touchgraph.sourceforge.net

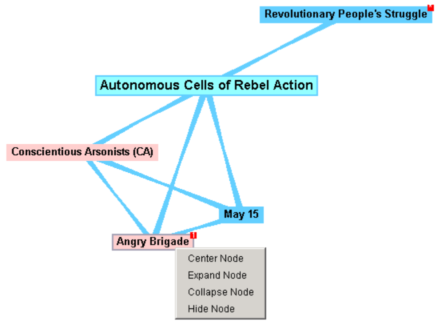


Figure 5: Node and Link Exploration

iteratively so as to merge multiple networks into one by integrating two networks each time. An input network can thus be from some data source or an integrated network.

Two node types from different data sources are considered *matched* if there exists node instances of the node types that may refer to the same real world entities. For example, the `terrorist group` node types in TKB and WikiTerrorism networks should be matched before establishing equivalence among their node instances. The `terrorist` nodes types should be matched similarly. Node matching refers to finding equivalent node instances from matched node types. In SSnetViz, this is performed by semi-automated *rule-based matching* and *manual matching*. We will elaborate both matching functions in the following subsections.

4.2 Rule-based and Manual Node Matching

Rule-based node matching is useful when there are matching rules that can help determining the matched node pairs. Each matching rule is represented by a conjunctive set of necessary but possibly not sufficient matching conditions that a pair of nodes have to meet in order for them to be considered as matched. Usually, each matching condition involves comparing the identifying attribute values (e.g., name) of nodes although other attributes can be used (e.g., date of birth). A node pair satisfying some matching rule is known as a *candidate node pair* which will be subsequently judged by the user. Only candidate node pairs confirmed by the user as matched will be stored in the integrated network. In SSnetViz, a matching rule is a conjunction of conditions and each condition involves a *matching function* and a pair of attributes, one from each node to be matched. Since different matched node pairs may be derived by different matching rules, SSnetViz maintains a list

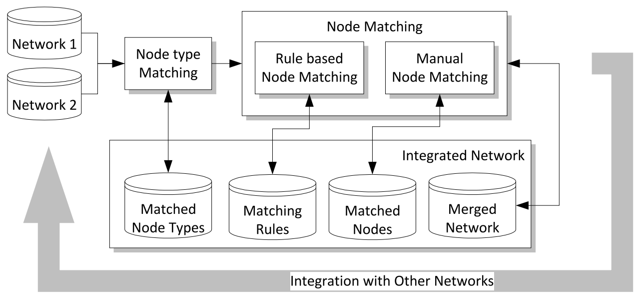


Figure 6: Integration Framework of SSnetViz

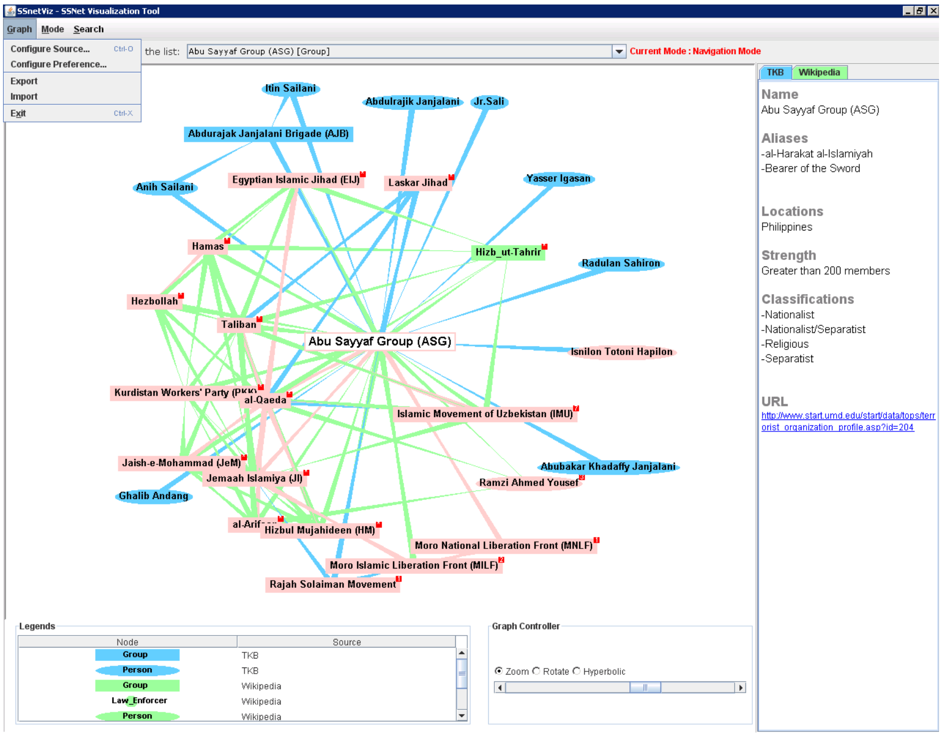


Figure 4: User Interface of SSnetViz

of matching rules defined by the user.

Figure 7 shows the network integrator's user interface allowing the user to define a matching rule using a fuzzy name match function on the name attributes. The matching rule says "if a TKB group node and a WikiTerrorism group node have their names satisfying a fuzzy string match, the two nodes are considered as matched." The candidate matched node pairs generated by the matching rule are shown in the middle. The user is required to manually judge whether each pair is matched or not by selecting the appropriate radio button (i.e., "yes","no", and "unconfirmed"). The unconfirmed option is reserved for unjudged and difficult-to-judge pairs. The numbers of already matched node pairs and not-yet-matched node pairs are shown in the two cylindrical bars on the right of Figure 7. The filled level of each bar indicates the proportion of nodes from the corresponding network that have been matched. This is updated whenever there is a change to the number of matched node pairs.

Manual node matching is conducted by directly selecting matched nodes from the networks. Figure 8 shows at the top two lists of terrorist group nodes from TKB and WikiTerrorism networks that have not yet been matched. The user can select matched node pair and add them to the matched set below.

5. NETWORK SEARCH

Network search provides functionality to help user analyze nodes and their links in a semantic social network. In SSnetViz, two search functions are provided, namely **node search** (to find relevant nodes that satisfy a query) and **path search** (to find paths connecting two given nodes.) Unlike the usual network search, SSnetViz is designed to support **metasearch** as each search query on an integrated semantic social network is evaluated on the multiple data indexes, one for each underlying component network. The evaluation of a metasearch (or simply search) requests in SSnetViz therefore involves (a) performing search requests on the component networks, (c) retrieving the network-specific search results, and (c) combining them into one single search result. In the following, we will elaborate the different types of search evaluation.

5.1 Node Search

Node search allows a user to query nodes using search conditions on the node attributes. In the current version of SSnetViz, we only support full text node search only. In other words, the search engine supports keyword search on all attribute values of nodes. Figure 9 shows the SSnetViz interface for node search. The figure depicts a search for "Abu Sayyaf" related nodes and the returned results consist of the Abu Sayyaf terrorist group and other incidents related to the group.

To search and retrieve result from multiple networks and node types, we build an index for each node type per component network. The nodes satisfying the node query are retrieved from these indexes and matched nodes (if exists) are merged into integrated nodes in the search result. The relevance score of each resultant node (r_i), a component network's node denoted by v_{ik} with a single k or a group of matched nodes denoted by v_{ik}'s as they come from different networks, is thus defined as:

$$combined\_score(r_i) = \sum_k w_k \times rel\_score(v_{ik})$$

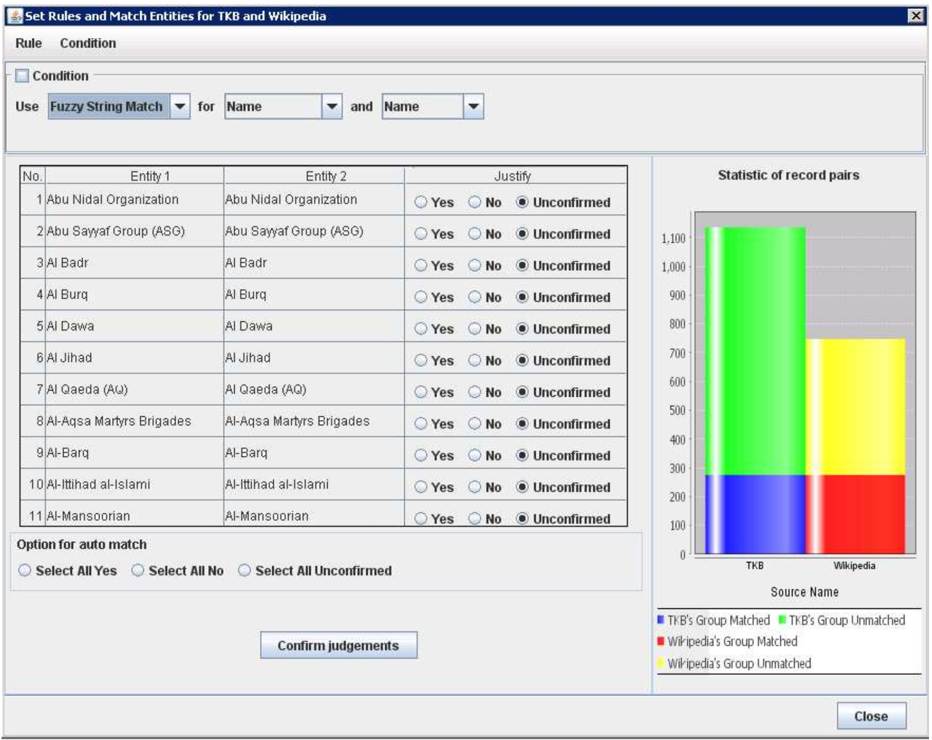


Figure 7: Matching Rule in SSnetViz

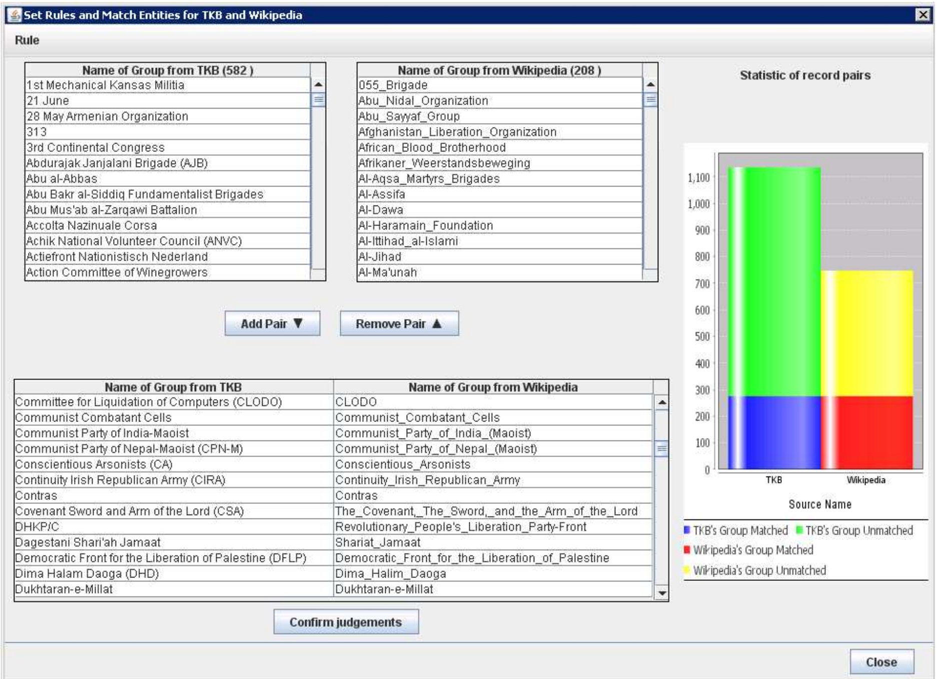


Figure 8: Manual Node Matching in SSnetViz

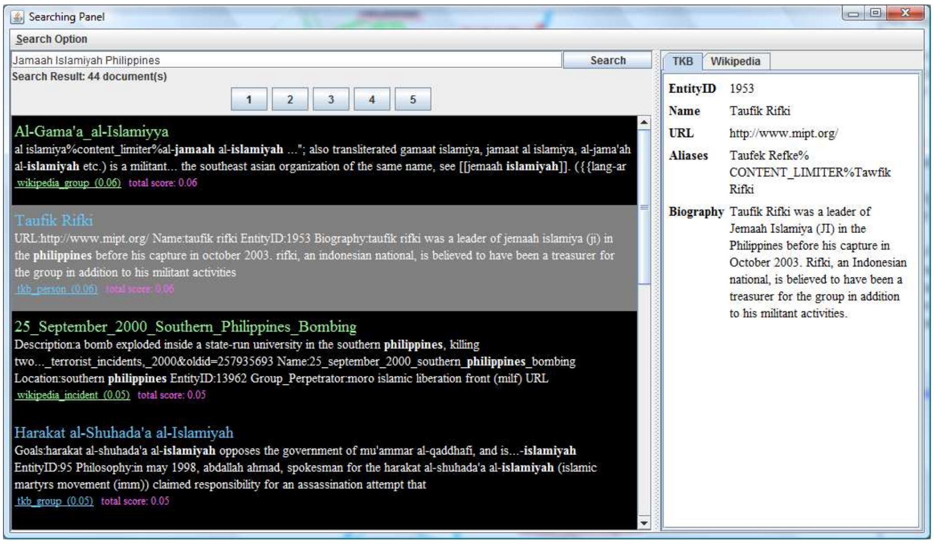


Figure 9: Node Search in SSnetViz

where w_k is the *weight of kth component network* of the integrated node and $rel\_score(v_{ik})$ denotes the relevance score of the matched node from the kth component network. We assume that there are no two matched nodes of the same integrated node coming from the same component network. The w_k value, tunable by the user, ranges between 0 and 1 representing least important to most important respectively, and $\sum_k w_k = 1$. When $w_k = 0$, search results from the kth component network will not be used at all.

The search result consists of a list of nodes sorted by decreasing *combined_score* values. As shown in Figure 9, a search query on "Jamaah Islamiyah Philippines" is performed and several resultant nodes are returned together with their network sources, relevance scores in their component networks and total (or combined) scores. The user can select a resultant node from any component network for viewing. When placing mouse pointer over a node, a tooltip box will appear summarizing the content of the selected node.

To add one or more resultant node into the Network Visualizer, the selected resultant node(s) can be dragged from search result panel directly to the network display panel in the Network Visualizer. The new resultant nodes will be assigned #'s at their top right corner distinguishing it the component network(s) from which the node is derived from. The newly added node is automatically connected with existing nodes in the visualizer if there are links among them. Figure 10 shows how the "Taufik Rikfi" node is added from search result.

An full-text index module and the search result ranking module have been implemented using Lucene[4], an open-source Java API for text indexing and searching.

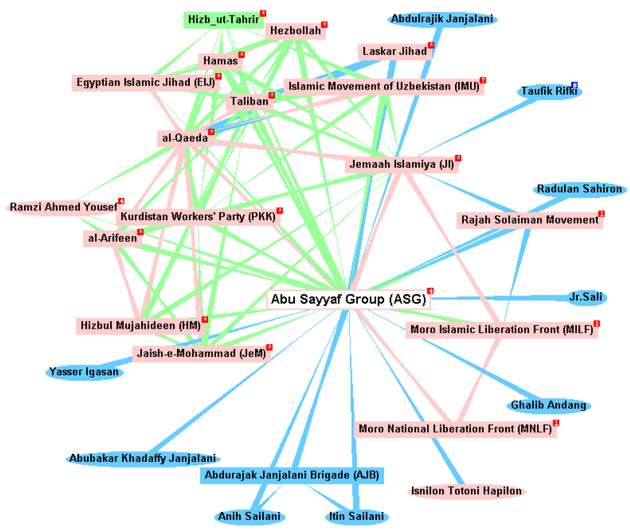


Figure 10: Adding a Resultant Node to Network

5.2 Path Search

Path search is used to analyze relationships between two given nodes in the network. Since the connection between the two nodes may be via multiple paths, path search attempts to find them so as to enable the user to discover useful connecting nodes between the two given nodes. Fig-

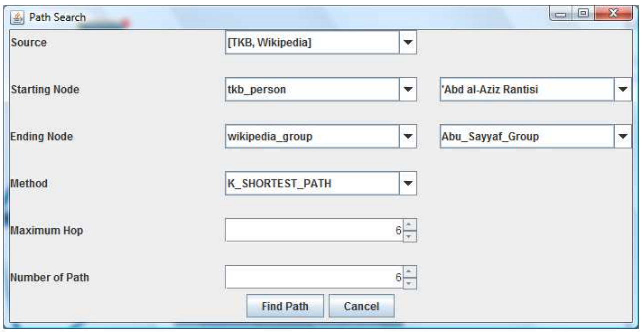


Figure 11: Path Search in SSnetViz

ure 11 shows the user interface for path search.

Given that our semantic social networks are derived from heterogeneous networks, the user needs to specify the (a) target integrated network, (b) the source and destination nodes, (c) the path scoring method, (d) the path length (denoted by k) constraint, and (e) the maximum number of paths to be returned. The source and destination nodes can be selected from integrated network. The path length constraint ensures that overly long paths will be filtered away.

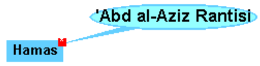
The path scoring method selection allows different measures to be used to determine the relevance of a path so that resultant paths can be ordered by decreasing relevance. There are currently three methods supported by SSnetViz:

- **Path length:** This uses path length as the relevance criteria. Shorter paths are given higher relevance.
- **Path betweenness:** We define the betweenness of a path to be the sum of the betweenness of nodes in the path. The betweeness of a node is defined by the number of shortest paths between any two nodes in the network that pass through the node. In the terrorist network, a node with high betweenness may act as key gatekeeper or broker for communication or flow of goods[9].
- **Path rarity:** The rarity of a path is defined by the average rarity of nodes along the path. The rarity of a node v with respect to a given pair of source and destination nodes, v_s and v_d, is defined by $rarity(v) = 1 - \frac{\#\text{ paths from } v_s \text{ to } v_d \text{ via } v}{\#\text{ paths from } v_s \text{ to } v_d}$[1].

To visualize the resultant paths in Network Visualizer, the user can select the required paths by highlighting their checkboxes. These paths will then appear in the Network Visualizer with nodes assigned with #'s. Multiple selected resultant paths with shared nodes will appear as a subgraph as shown in Figure 12.

6. NETWORK EXPORT AND IMPORT FUNCTIONS

SSnetViz also provides the export and import functions for a user to save, load, import or export networks. Once an interesting network is derived by searching or navigating nodes and paths, the user can export it so as to save a copy of the network or to allow the network to be visualized using other network visualization software. To ensure the exported networks are readable by other systems, we have chosen to represent the exported networks in GraphML[3], a widely used XML representation of graphs. The exported network can be later imported into SSnetViz again for viewing.




```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<graphml>
      <key attr.type="string" id="label"/>
      <key attr.type="int" id="x"/>
      <key attr.type="int" id="y"/>
      <key attr.type="double" id="srcWght"/>
      <key attr.type="double" id="dstWght"/>
      <key attr.type="string" id="entity_type_id"/>
      <key attr.type="string" id="sourceid"/>
      <key attr.type="string" id="entity_id"/>

<graph edgedefault="directed">
      <node id="3">
      <data id="label">Hamas</data>
      <data id="x">-113</data>
      <data id="y">159</data>
      <data id="entity_type_id">1</data>
      <data id="sourceid">1</data>
      <data id="entity_id">1215</data>
      </node>

      <node id="2">
      <data id="label">'Abd al-Aziz Rantisi</data>
      <data id="x">113</data>
      <data id="y">-159</data>
      <data id="entity_type_id">5</data>
      <data id="sourceid">1</data>
      <data id="entity_id">1928</data>
      </node>

      <edge source="2" target="3">
      <data id="srcWght">1.0</data>
      <data id="dstWght">1.0</data>
      </edge>
</graph>
</graphml>
```

Figure 13: GraphML Representation of Network

In GraphML, each node is defined as an XML element. Each node element has an identifier. Every link in the network is defined as edge element. Each edge element must specify its two endpoints with source and target node identifiers. Figure 13 shows how a small network of Abd al-Aziz Rantisi is represented in GraphML. The GraphML file can be created by selecting the export function in the graph menu item in Figure 4.

7. CONCLUSION

In this paper, we present the design framework of SSnetViz, a visualization engine for heterogeneous semantic social networks. SSnetViz is unique in its capability to handle integration of multiple networks to improve network search and analysis. Its network integrator module supports both matching rule and manual based node integration. Network search and navigation can be performed across networks. SSnetViz also provides network export and import for interoperability with other network analysis software.

SSnetViz is still in the midst of development as more data are still being gathered and the visualization engine is still being constructed. There are much room for further research as listed below:

- *Attribute conflict resolution:* While SSnetViz addresses network integration by finding matched node pairs, it does not resolve attribute value differences for matched node pairs. SSnetViz currently assumes that every node type has its unique set of attributes. In practice, this assumption may not hold as matched node

[3]http://graphml.graphdrawing.org/

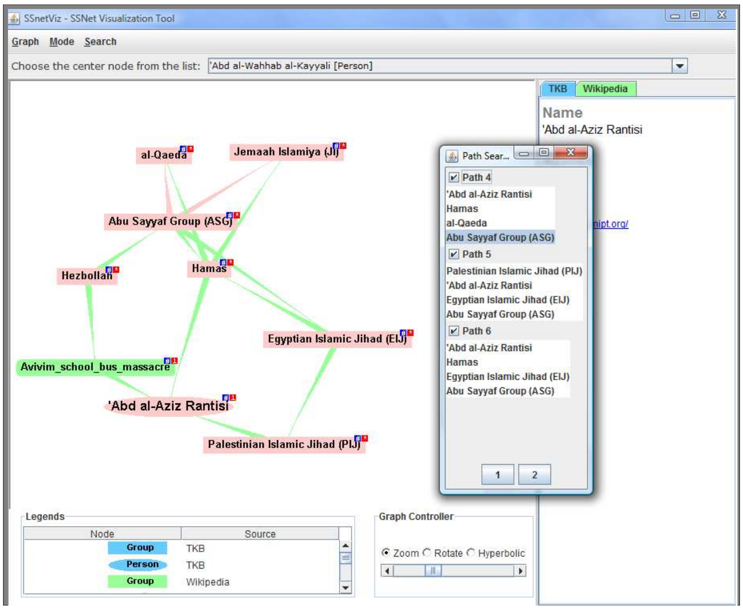


Figure 12: Resultant Path Visualization

types may have overlapping attributes. For example, terrorist nodes from two networks may share common attributes such as birthdate, education level, and religion. Hence, it is necessary to examine how attribute value differences for two or more records of the same terrorist can be resolved.

- *Community finding:* Community finding is an important task in social network analysis. SSnetViz currently does not support the task. Path search can only establish connections between two nodes but not their communities. We therefore plan to develop a community finding function using some graph partitioning method.

8. ACKNOWLEDGMENTS

We acknowledge the funding support of Agency for Science and Technology Research (A*Star) under Project 062 101 0031.

9. REFERENCES

[1] B. Aleman-Meza, C. Halaschek-Wiener, I. B. Arpinar, C. Ramakrishnan, and A. P. Sheth. Ranking complex relationships on the semantic web. *IEEE Internet Computing*, 9(3):37–44, 2005.
[2] I. Bhattacharya and L. Getoor. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data*, 1(1):1–36, 2007.
[3] L. Freeman. Visualizing social networks. *Journal of Social Structure*, 1(1), 2000.
[4] E. Hatcher, O. Gospodnetic, and M. McCandless. Lucene in action, 2009.
[5] J. Heer and D. Boyd. Vizster: Visualizing online social networks. *IEEE Symposium on Information Visualization*, 2005.
[6] H. Kang, L. Getoor, and L. Singh. Visual analysis of dynamic group membership in temporal social networks. *SIGKDD Exploration Newsletter*, 9(2):13–21, 2007.
[7] A. Katifori, C. Halatsis, G. Lepouras, C. Vassilakis, and E. Giannopoulou. Ontology visualization methods—a survey. *ACM Computing Survey*, 39(4):10, 2007.
[8] E.-P. Lim, J. Srivastava, S. Prabhakar, and J. Richardson. Entity identification in database integration. In *International Conference on Data Engineering*, pages 294–301, 1993.
[9] J. Qin, J. J. Xu, D. Hu, M. Sageman, and H. Chen. Analyzing terrorist networks: A case study of the global salafi jihad network. In *IEEE International Conference on Intelligence and Security Informatics*, pages 287–304, 2005.
[10] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 10(4):334–350, 2001.
[11] Z. Shen, K.-L. Ma, and T. Eliassi-Rad. Visual analysis of large heterogeneous social networks by semantic and structural abstraction. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1427–1439, 2006.
[12] S. Wasserman and F. Katherine. *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.
[13] J. J. Xu and H. Chen. Criminal network analysis and visualization. *Communications of the ACM*, 48(6):100–107, 2005.
[14] C. C. Yang, N. Liu, and M. Sageman. Analyzing the terrorist social networks with visualization tools. In *IEEE International Conference on Intelligence and Security Informatics*, pages 331–342, 2006.