COMPLEX RELATION DISCOVERY FROM THE SEMANTIC WEB

A Paper
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Mousumi Tanha

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science

April 2010

Fargo, North Dakota

North Dakota State University
Graduate School

Title

COMPLEX RELATION DISCOVERY

FROM SEMANTIC WEB

By

MOUSUMI TANHA

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

ABSTRACT

Mousumi Tanha, M.S., Department of Computer Science, College of Science and Mathematics, North Dakota State University, April 2010. Complex Relation Discovery from the Semantic Web. Major Professor: Dr. Juan Jen Li.

The vision of the Semantic Web undertakes an extension of the current Web, in which machines can understand all the data. The nature of Semantic Web data is relationship-centric and is very complex. In this study we aimed to discover those complex but meaningful and concealed relationships between resource entities from the Semantic Web data. We utilized the notion of semantic relation discovery approach which aims to capture meaningful and probable complex relationships between entities in a dataset based on graph search model. We considered three fictitious datasets for the experiment. The outcome showed sequences and connections among the nodes and how the nodes are semantically inter-related.

ACKNOWLEDGEMENTS

I am grateful for the help and inspiration I got from the faculty of the Computer Science Department. I am especially grateful to Dr. Juan Jen Li, who is an excellent mentor, for her assistance in every aspect of this research, her support, her guidance and her motivation. I would also like to thank my supervisory committee members for their excellent support. Without their help, it would have been impossible for me to come up with this paper.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

The Semantic Web is a Web of meaning. It is a vision of information that is understandable by computer applications, so that applications can perform more of the monotonous work involved in finding, combining and acting upon information on the Web [1]. On the current Web, statements are built with syntax rules. The syntax defines the rules for building language statements. But syntax cannot be semantic. This is why we need the Semantic Web to describe things in a way that computer applications can understand.

The Semantic Web is not about links between Web pages. The Semantic Web describes the relationships between things (such as - A is a part of B and Y is a member of Z) and the properties of things (such as - size, weight, age and price) [2]. As the current Web evolves into the next generation termed the Semantic Web, the emphasis will shift from finding documents to finding facts, actionable information and insights [5]. Improving ability to extract facts, mainly in the form of entities, embedded within documents leads to the underlying challenge of discovering relevant and interesting relationships amongst the entities [1]. The Semantic Web allows us to take better advantage of information on the Web. Most of the time people go to a number of sites and/or download information. The process is labor demanding; it requires opening a new browser session for each site. The Semantic Web approach greatly simplifies this process. In general, if we need data from ten sites, we need to go to all ten sites and cut and paste the data to get an integrated view [17]. A Semantic Web browser can be configured to go to multiple sites, find the specific information required, retrieve this information and display it in a single Semantic Web browser. Such capabilities make the Semantic Web very

interesting and popular these days. The beginning of the Semantic Web is providing the standards and tools needed to build integrative informatics systems. One of the most powerful features of the Semantic Web is the ability to write and carry out complex rules with very little programming effort. In the Semantic Web search it is possible to drag and drop views from pathway network to relationship views. With such features, appreciation is growing that the Semantic Web offers much more value than simple data aggregation technology does [1].

The Semantic Web discovers relevant and interesting relationships amongst entities that any document can describe. These relationships are the basis of analysis and underpin the semantics of the data. There are two types of relationship that we can search for in the Semantic Web - simple relationships and complex relationships [17]. These relationships may be based only on what is contained in or directly derived from data (direct content based relationships), or they may be based on information extraction, external and prior knowledge and user defined computations (content descriptive relationships). It is also possible to discover indirect and virtual (user-defined) yet meaningful (i.e., contextually relevant) relationships based on a set of patterns and paths between entities of interest [1].

The Semantic Web complex relation discovery has both theoretical and practical application. Discovery of complex relationships is an important component in the Semantic Web analytics. Computing complex relationships require new forms of processing data, relevant knowledge and associated techniques of creating and maintaining a variety of relationships. Instead of relying only on data, it depends on a broad variety of domain knowledge context, which enables scalability by ignoring irrelevant information [1] [2].

Many applications in analytical domains such as national security and business intelligence require a more complex notion of relationships than the simple direct relationships between the entities [5]. Semantic relations in the most basic sense involve evaluating a set of contextually relevant paths of relations from one entity to another. By evaluating such paths it may identify relations based on connectivity or similarity of paths. This allows us to analyze sequences of complex relationships, instead of simple and single binary relationships and manipulate these sequences to find similar entities as well as entities that may be connected directly or indirectly [1].

1.1. Motivation

Purpose of this research is to discuss the common research challenges of Complex Relation Discovery from the Semantic Web. The success of this visualization can be measured by contribution to the increasing use of the Semantic Web applications [2]. Complex relation discovery is an essential key for the Semantic Web knowledge discovery. The Semantic Web seeks to correlate annotations to discover the knowledge primarily based on concepts from ontologies and vocabularies with all Web-accessible resources that programs can associate with data. This process also enables scalability improvements [1].

1.2. Problem Statement

This research will present a scheme that can be used to discover complex but significant relationships from the Semantic Web. The research will take an experimental look into the complex relationships of the Semantic Web and will explore the development of the Semantic Web data capture and representation. The study will be focused on the relationships based only on the contents that are directly derived from data, or based on information extraction, external and prior knowledge and user defined computations [1]

[2]. Meaningful and functional data is well defined by the Semantic Web which is specifically understandable by computers. As a result of extracting Semantic metadata from document headers the internet information systems can query and retrieve more relevant information efficiently from documents [2]. However, it is possible for the Semantic Web data to expose much more complex relationships than simple information. In the Semantic Web the databases increasingly store metadata, so the number of relationships between entities and descriptions of data will need to be further defined. Basic ontological structures are being developed to constrain that information in schema form [3]. Instances of semantic structures still reside in text/XML format on the current Web. As the number of semantic relationships explodes on the Internet, efficient search algorithms will need to be created to absorb the data and discover the complex semantic relationships [2].

1.3. Proposed Solution

This study aims to discover complex, meaningful and concealed relationships between resource entities from the growing Semantic Web data. It will facilitate users to uncover previously unknown and potentially interesting correlations between two or more entities, or associations between two or more people. The assumption is that any two Web pages that have some kind of relationship must be connected by a link or a path.

In this paper graph search method will be explored to address this problem .We will implement iterative deepening depth first search (IDDFS) algorithm. The algorithm will focus to investigate the automatic generation of meta-data for the Semantic Web. IDDFS will minimize space and time complexity to provide complete and optimal complex paths for the relation discovery. Complex relations are usually multi-hop relations, with goals probably being found at very low levels in the tree. Therefore iterative deepening depth

first search returns deeper paths much faster and provides more significant results. The proposed graph search algorithm has limitations. Such as, if a relation is not close enough and if it is too far from the start node, then the results will be returned with limited amount of path discovery. The limit will be basically a depth limit for the IDDFS. The more complex and more distant the paths are, the deeper in the tree the goal node will be found.

1.4. Paper Organization

The paper is organized as follows:

- Chapter 1 includes an introduction of the topic, motivation and goals of the study, problem statement and proposed solution.
- Chapter 2 provides background of the topic and also provides discussion about related work; it also includes some comparison of the Semantic Web and data mining relation discovery schema.
- Chapter 3 contains methods, which provides a description of the algorithm used for the experiment, a comparison with other algorithm and the process of the experiment.
- Chapter 4 provides the results and analysis for the three fictitious examples used in the experiment.
- Chapter 5 includes the conclusion, summary of the results, limitations and recommendations for future work.

CHAPTER 2. BACKGROUND AND RELATED WORK

2.1. Background

2.1.1. The Semantic Web

The Semantic Web is a globally linked mesh of ontological information to be understood by computers. The Semantic Web is not an application; it is an infrastructure on which many different applications will develop. This is a subset of the World Wide Web and the World Wide Web is a globally linked mesh of information. The Semantic Web makes use of ontological schemas to define data in Web documents and creates instances of those objects defined in the schemas as metadata added to Web documents. This metadata can then be polled by artificially intelligent agents to produce meaningful search results or provide more detailed information about Web resources [13].

According to the US national science foundation's research the goal of the Semantic Web is to be "a Web talking to machines", in which machines can provide a better help to people because they can take advantage of the content of the Web. The information on the Web should thus be expressed in a meaningful way accessible to computers. This definition is easily related to what already exists on the Web. The Semantic Web can be used for extracting data from regularly structured pages, natural language analysis for extracting Web page contents, indexing schemes, syndication facilities for broadcasting identified Web resources [8]. The Semantic Web can also be thought of as an infrastructure for supplying the Web with formalized knowledge in addition to its actual informal content. No consensus exists on how far the formalization

should go. It ranges from metadata schemes to full-fledged logical representation languages.

2.1.2. Resource Description Framework (RDF)

RDF is a standard model for data interchange on the Semantic Web. RDF has features that facilitate data merging even if the underlying schemas differ. It specifically supports the evolution of schemas over time without requiring all the data to be changed. RDF extends the linking structure of the Web to use URIs (Uniform Resource Identifiers) to name the relationship between things as well as the two ends of the link [17]. RDF allows structured and semi-structured data to be mixed, exposed and shared across different applications. The linking structure forms of RDF directed and labeled graphs, where the edges represent the named link between two resources, is represented by the graph nodes. This graph view is the easiest possible mental model for RDF and is often used in easy-to-understand visual explanations [9]. RDF provides an ontological description of a resource.

Benefits of RDF is that it can draft a language in the information maps directly and unambiguously as a model, a model which is decentralized and for which there are many generic parsers already available. That means when we have an RDF application, we will know which bits of data are the semantics of the application and which bits are just syntactic fluffs. RDF data is becoming a part of the Semantic Web, so the benefits of drafting data in RDF now draws parallels with drafting information in HTML [9].

2.1.3. Ontology

Ontology is a set of representational primitives. The representational primitives are typically classes (or sets), attributes (or properties) and relationships (or relations among class members) [5]. The definitions of the representational primitives include information

about their meaning and constraints on their logically consistent application. In the context of database systems, ontology can be viewed as a level of abstraction of data models. Ontology is analogous to hierarchical and relational models. It is intended for modeling knowledge about individuals, their attributes and their relationships to other individuals [3]. Ontology typically specified in languages that allow abstraction away from data structures and implementation strategies. In practice, the languages of ontologies are closer in expressive power to first-order logic than languages used to model databases. For this reason, ontologies are said to be at the "Semantic" level, whereas database schemas are models of data at the "logical" or "physical" level [7]. Due to their independence from lower level data models, ontologies are used for integrating heterogeneous databases, enabling interoperability among disparate systems. Ontologies specify interfaces to independent, knowledge-based services. In the technology stack of the Semantic Web standards, ontologies are called out as an explicit layer. There are now standard languages and a variety of commercial and open source tools for creating and working with ontology for the Semantic Web [3].

A semantic system needs to provide and mange metadata about resources to locate resources of interest. In that case ontology can represent resource metadata; because ontology is able to express the meaning of resource information. An ontological representation defines concepts and relationships. It sets the vocabulary, properties and relationships for concepts [7]. The elements accumulate more meaning by the relationships they hold and the potential inferences that can be made by those relationships. Web Ontology Language (OWL) is an ontology language for the Web. It can process the content instead of just presenting information. OWL can be used to explicitly represent the

meaning of terms in vocabularies and the relationships between those terms of ontology for the Semantic Web [6].

2.2. Related Work

2.2.1. Complex Relations Discovery

The Semantic Associations are meaningful complex relationships between entities, events and concepts. That meaningful information is universally understandable; it can be extended to discover new relationships in the Semantic Web. Various studies have been carried out over the years to find the complex relationships between the entities using the Semantic Web [18]. The complex semantic relationships between the entities have a directed path or a sequence to get from one entity to another [6].

The use of complex relationships can be particularly useful in the National Security Applications; in particular the aviation security. An aviation security application of significance to national security dealing with a system called PISTA for Passenger Identification, Screening and Threat Analysis has been studied as a prototype of the Semantic Association for semantic metadata [11]. It extracts relevant metadata from different information resources including government watch-lists, flight databases and historical passenger data and using some semantic-based knowledge discovery techniques. It can identify suspicious patterns and categorize passengers into high-risk groups, low-risk groups, no-risk groups and positive groups using the metadata that has been extracted [7].

Ranking or prioritizing of documents is a critical component of today's search engines. With the development of the Semantic Web, the retrieval of information and the ranking of the complex relationships obtained from the Semantic Web is becoming very important [10]. Searching and ranking documents can be done with some relevance

measures such as relevance weights that are assigned to the relationships by the humans and a relevance threshold. The relationships of named entities can be analyzed with respect to a query. The results can be obtained within this threshold [8].

2.2.2. Association Discovery Rules and Data Mining

An association discovery is at the heart of data mining. It detects hidden linkages of otherwise seemingly unrelated data. These linkages are the rules, those that exceed a certain threshold and are deemed interesting. Interesting rules allow actions to be taken based upon data pattern. Association discovery is generally applied to database transactions where each transaction consists of a set of items. In such a framework the problem is to discover all associations and correlations among data items, where the presence of one set of items in a transaction implies the presence of other items. In the context of Web mining, this problem amounts to discovering the correlations among references to various files available on the server by a given client. Each transaction is comprised of a set of URLs accessed by a client in one visit to the server [14].

Most of the time association discovery rules describe frequent co-occurrences in sets, such as - an item set is a subset A of all possible items. The general form of association discovery – $A_1, A_2, ..., An \rightarrow B_1, B_2, ..., B_m$

Interpretation: When items A_1 appear, items B_1 also appear with a certain probability

Example Problems: Which products are frequently bought together by customers?

DataTable = Receipts x Products

Results could be used to change the placements of products in the market

Examples: Bread, Cheese → RedWine.

Customers that buy bread and cheese, also tend to buy red wine.

On the other hand, we can explore data mining which is a great aspect of Database Query Processing. Data mining queries are on the ad hoc or unstructured end of the query spectrum rather than standard report generation or "retrieve all records matching a criteria [4].Data mining tools sweep through databases and identify previously hidden patterns in one step. When data mining tools are implemented on high performance parallel processing systems, they can analyze massive databases in minutes. Faster processing means that users can automatically experiment with more models to understand complex data. High speed makes it practical for users to analyze huge quantities of data.

2.2.3. Data Mining VS. The Semantic Web

The complex semantic relation discovery is different from data mining that uses statistical techniques to find co-occurrence relationships between predicates based on patterns in data. There are more differences between data mining and the Semantic Web, the main difference is the placement of complexity [1]. The essence of data mining is the data and knowledge represented with simple mechanisms (typically, HTML) without metadata (data about data). In data mining consequently, relatively complex algorithms have to be used (complexity migrated into the retrieval request time).

On the other hand in the Semantic Web data and knowledge represented with complex mechanisms and with plenty of metadata. In the Semantic Web consequently, relatively simple algorithms can be used. For computing relationships, what we need is very different from data mining, at least as it has been traditionally understood in terms of grouping or market basket type analysis through the discovery of association rules. Data mining techniques are typically based on statistics and look for patterns that are already present in the data [7]. The patterns are sought at a syntactic level and do not take into

consideration the meaning of the data. They are not easily extendable to look for the types of relationships that are meaningful to humans or to the software agent. [1].

CHAPTER 3. METHOD

3.1. The Complex Semantic Web Architecture

The Semantic Web architecture works by converting text files to XML and by analyzing the files with a Semantic processor. This process understands the meaning of the words and structure of the sentence and also the Semantic relationships of the context. These meanings and relationships are then stored in the Semantic Web database. It contains all of the logical content and context of the original source and it links each word and concept back to the original document. Figure 1 shows the Semantic Web architecture:

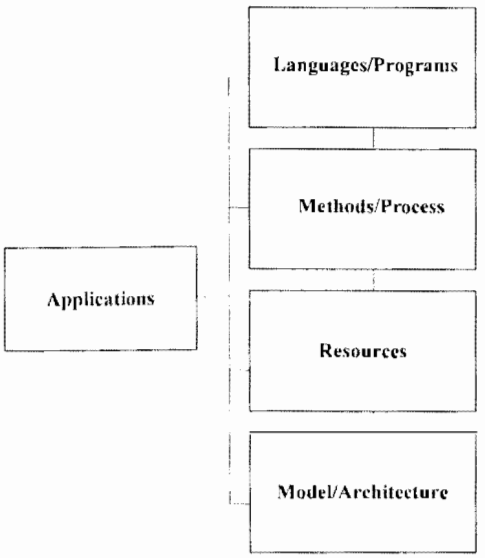


Figure 1. The Semantic Web Architecture

The Semantic Web applications can efficiently and accurately search, retrieve, summarize, analyze and report discrete concepts or entire documents from huge databases. The architecture is a process of the Semantic Web complex relation discovery which can automatically extract and process the concept and context in the database. As the number complex relationships explodes on the Semantic Web, efficient graph and search

algorithms is needed to be created in order to digest the data and discover complex Semantic relationships.

3.2. Design of the Study

To discover the complex relationships we will be using an iterative depth-first search to traverse the large data sets efficiently. The algorithm used is an Iterative Depth First Search (IDFS) to minimize space and time complexity and to provide complete and optimal complex paths. Complex relations are usually multi-hop relations, with goals probably being found at very low levels in the tree. Therefore IDFS returns deeper paths much faster, providing more meaningful results. Because there are no costs or heuristic functions for the search, A* search or cost-based searches are irrelevant. Graph search and tree traversal searches require the entire graph to be loaded into memory, which is not possible with large datasets in Semantic meta-bases.

A complex Semantic relation can be defined as the path between two entities where a path is defined as: $e_1, P_1, e_2, P_2, \ldots, e_{n-1}, P_{n-1}, e_n$. Where e_i is an entity and P_i is a predicate or property that defines a relationship between two entities, e_i and e_{i+1}. Therefore this denotes a complex Semantic relation between entities e_1 and e_n. An important measurement for these complex relations is the number of hops it takes to reach en from e_1. Hop count is the number of traversals down a directed a cyclical graph tree created by these relationships. Alternatively, the number of hops is the number of triples required to generate the path [15]. Figure 2 shows the model of the study:

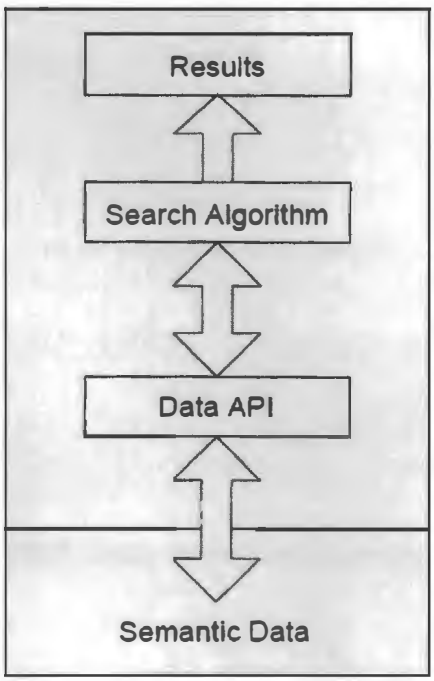


Figure 2: Work Process Model of the Study

We will create a tree to link nodes to each other and then we will make graphs to see the complexity of the relations between the nodes. Then with the Semantics and the complexities of the graph, we will run the data sets to our iterative deepening algorithm using Java programming language to obtain the result.

3.2.1. Graph Search Based Relation Discovery

A graph search (or graph traversal) algorithm is just an algorithm to systematically go through all the nodes in a graph, often with the goal of finding a particular node, or one with a given property. Searching a linear structure such as a list is easy: we can just start at the beginning and work through to the end. Searching a graph is obviously more complex [15]. There are two main ways to traverse a graph: depth first and breadth first. If we start at a particular node (n_1), then in breadth first search, all nodes that are path length M away from n_1 are searched before all nodes that are path length M_{+1} away. In depth first search, if

a node n_2 is searched, all nodes connected to n_2 are searched before any other nodes. We can also describe it in terms of family tree terminology: in depth first the node's descendants are searched before its (unvisited) siblings; in breadth first, the siblings are searched before its descendants.

For searching a general graph it is necessary to keep track of which nodes have already been searched. If we don't do this and if there are cycles in the graph, then the loop might never terminate [15]. Even if there are no cycles, redundant work is done, re-visiting old nodes. Avoiding revisiting of previously visited nodes leads to the following modified algorithm, which keeps track of nodes visited (using an array visited, which would be initialized appropriately).

3.2.2. Depth-First Based Relation Discovery

DFS is the general search algorithm where the insert function is "enqueue-at-front". This means that newly generated nodes are added to the fringe at the beginning, so they are expanded immediately. In this case, the queue acts like a stack and it is easy to implement with a list. DFS goes down a path until it reaches a node that has no children. Then DFS "backtracks" and expands a sibling of the node that had no children. If this node has no siblings, then DFS looks for a sibling of the grandparent and so on. Depth of current node is not a factor [15]. DFS will always go deeper if it has a child. The major weakness of DFS is that it will fail to terminate if there is an infinite path "to the left of" the path to the first solution. In other words, for many problems DFS is not complete: a solution exists but DFS cannot find it [15]. The major advantage of DFS is that it only uses O (bm) space if the branching factor (number of children returned by the expand function) is b and the maximum depth is m.

3.2.3. Breadth-First Based Relation Discovery

The breadth-first strategy always expands all the nodes at one level of the tree, before expanding any of their children. This strategy is guaranteed to find the shortest solution path first. In most circumstances, the shortest path is also the *best* solution available. BFS must represent all paths simultaneously. The memory cost thus increases exponentially with the exploration depth and can be calculated in the usual way, using the branching factor raised to the relevant depth. If there are few solutions, however, the strategy may be more effective than depth-first search. A depth-first search may waste time exploring deep into the tree. BFS is guaranteed not to do this.

3.2.4. Iterative Deepening Based Relation Discovery

Iterative deepening is a very simple, very good, but counter-intuitive idea that was not discovered until the mid 1970s. The idea is to perform depth-limited DFS repeatedly, with an increasing depth limit, until a solution is found. Intuitively, this is a dubious idea because each repetition of depth-limited DFS will duplicate uselessly all the work done by previous repetitions. But, this useless duplication is not significant because a branching factor $b > 1$ implies that the number of nodes at depth k is much greater than the total number of nodes at all depths k-1 and less. For a problem with branching factor b where the first solution is at depth k, the time complexity of iterative deepening is O (bk) and its space complexity is O (bk). This means that iterative deepening simulates breadth-first search, but with only linear space complexity.

IDDFS is arguably the best, general-purpose search strategy since it offers the low memory costs of depth-first search together with the optimality and completeness of breadth-first search. Intuition suggests that IDS will do a lot of unnecessary work since it

will repeatedly explore the upper levels of the search tree. However, in general, most of the nodes in a search space are in the lower levels. By avoiding unnecessary exploration of these levels, the iterative-deepening strategy manages to achieve a respectable time complexity.

3.2.5. Comparison of the Algorithms and Methods

Depth-first search achieves its efficiency by generating the next node to explore only when this needed [20]. The breadth-first search algorithm has to grow all the search paths available until a solution is found and this takes up memory. Iterative deepening achieves its memory saving in the same way that depth-first search does, but at the expense of redoing some computations again and again (costs time rather than memory). Figure 3, Figure 4, Figure 5 and Figure 6 show the different levels of work process of the algorithms and a comparison of the algorithms:

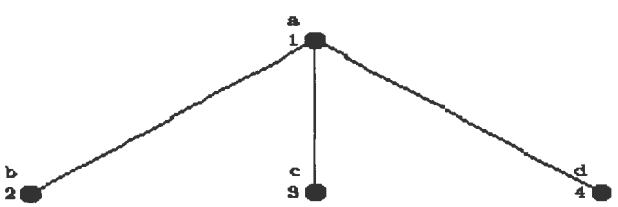


Figure 3. Iterative Deepening to Level 1

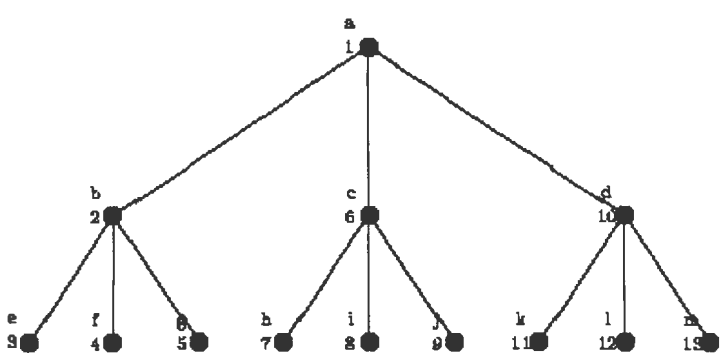


Figure 4. Iterative Deepening to Level 2

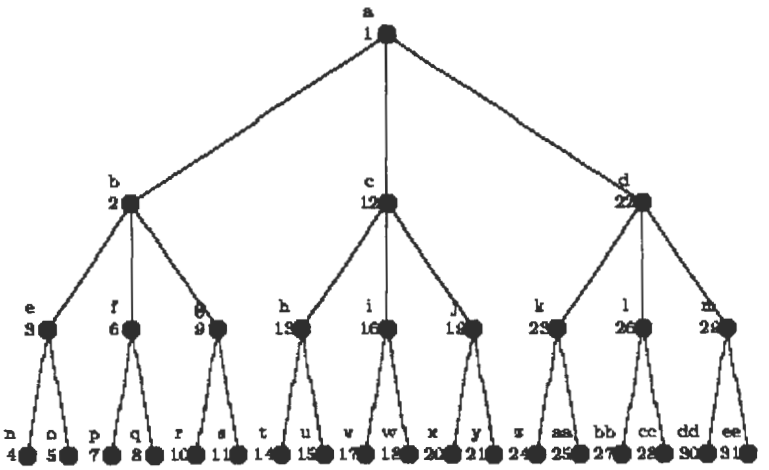
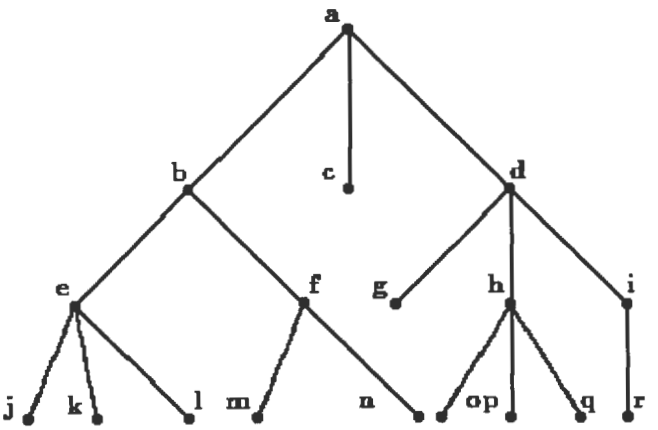


Figure 5. Iterative Deepening to Level 3



A 'Best' Case:	**goal state is j**	
Depth-First:	**4 nodes examined**	
Breadth-First:	**10 nodes examined**	
Iterative Deepening:	**23 nodes examined**	**(perhaps 10)**
A 'Worst' Case:	**goal state is d**	
Depth-First:	**11 nodes examined**	
Breadth-First:	**4 nodes examined**	
Iterative Deepening:	**4 nodes examined**	

Figure 6. Depth-First, Breadth-first and Iterative Deepening Compared

3.2.6. Work Process of Iterative Deepening

The algorithm uses an iterative deepening depth First Search to minimize space and time complexity and to provide complete and optimal complex paths. Complex relations are usually multi-hop relations, with goals probably being found at very low levels in the tree. Therefore IDFS returns deeper paths much faster- providing for more meaningful results [13]. There are no costs or heuristic functions for the search, so A* search or cost-based searches are irrelevant. Figure 7 shows the pseudo code [19] that we have used for our study:

```
                                        pseudocode_aima.txt
function IterativeDeepeningSearch(problem);
begin
        for depth = 0 to infinity  do
                begin
                        result <- depthsearch(problem, depth);
                                if result != cutoff then
                                        return result;
                end
        return solution;
end

function depthsearch (problem, limit);
begin
        recursive_depthsearch (MAKE_NODE(problem.INITIALSTATE), problem, limit);
        return solution;
end

function recursive_depthsearch (node, problem, limit);
begin
        if problem.GOAL_TEST(node.STATE) then
                return SOLUTION(node);
        else if limit = 0 then
                return cutoff;
        else
                cutoff_occurred? <- false
                for each action in problem.ACTIONS(node.STATE) do
                begin
                        child <- CHILD-NODE(problem, node, action)
                        result <- recursive_depthsearch (child, problem, limit - 1)
                        if result = cutoff then
                                cutoff_occurred? <- true
                        else if result != failure then
                                return result;
                        if cutoff_occurred? then
                                return cutoff;
                        else
                                return failure;
                end
        return solution;
end
```

Figure 7. Pseudo Code

Graph search and tree traversal searches require the entire graph to be loaded into memory, which is difficult with the large datasets in Semantic Meta bases. IDDFS can find an optimal solution path and it also obeys the same asymptotic branching factor as A*. If the number of newly expanded nodes grows exponentially with the search depth growth rate, then the heuristic branching factor depends on the average number of applicable operators per node [15].

3.2.7. Importance of Iterative Deepening Search

Iterative deepening is a popular method of search. Depth-first search can be implemented to be much cheaper than breadth-first search in terms of memory usage. But it is not guaranteed to find a solution even where one is guaranteed [10]. On the other hand, breadth-first search can be guaranteed to terminate if there is a winning state to be found and will always find the quickest solution. It is, however, a very expensive method in terms of memory usage. Iterative deepening is preferred because it is an effective compromise between the two other methods of search. It is a form of depth-first search with a lower bound on how deep the search can go. Iterative deepening can produce the same solution that breadth-first search would produce but does not require the same memory usage (as for breadth-first search).

CHAPTER 4. RESULT AND ANALYSIS

4.1. Result

We used iterative deepening depth first search algorithm to find a better way for complex relation discovery in the Semantic Web. We gathered some random hypothetical data to experiment and analyze the results. Our algorithm was limited by options including K-hop limit (i.e. a complex relation is not closely enough related if it is too far from the start) [6]. The K-hop limit is essentially our depth limit for the iterative depth first searches because the deeper in the tree a goal is found, the more complex and more distantly related the path is. Every time a goal is found, the path back to the root; and then the search continues until the iterative depth first search is exhausted. In our result we noticed that in the search algorithm total cost of a node is made up of the cost already spent in reaching that node, plus a lower bound on the estimated cost of the path to a goal state. At the beginning, the cost bound is set to the heuristic estimate of the initial state. The bound is increased for all iteration to the minimum value that exceeded the previous bound. The outcome showed sequences and connections among the nodes and the results also showed how the nodes are semantically interrelated.

This study used three different fictitious cases as examples to discover the complex relation from the Semantic Web:

- Friend of A Friend method (FOAF)
- Impact and Effects of Global Warming
- Terrorist Relation discovery

4.1.1. Case Study 1: Friend of A Friend (FOAF)

In the first example, the research considered a hypothetical Friend of A Friend (FOAF) scenario. In this example we have tried to show how the Iterative Deepening Search finds the connection of friends from a root node to a goal node. In this fictitious scenario M. Tanha is the root node and Tracy Lee is the goal node. Figure 8 and Figure 9 show the path of connection of the friends, it shows who knows who by starting from the root node and ending to the goal node.

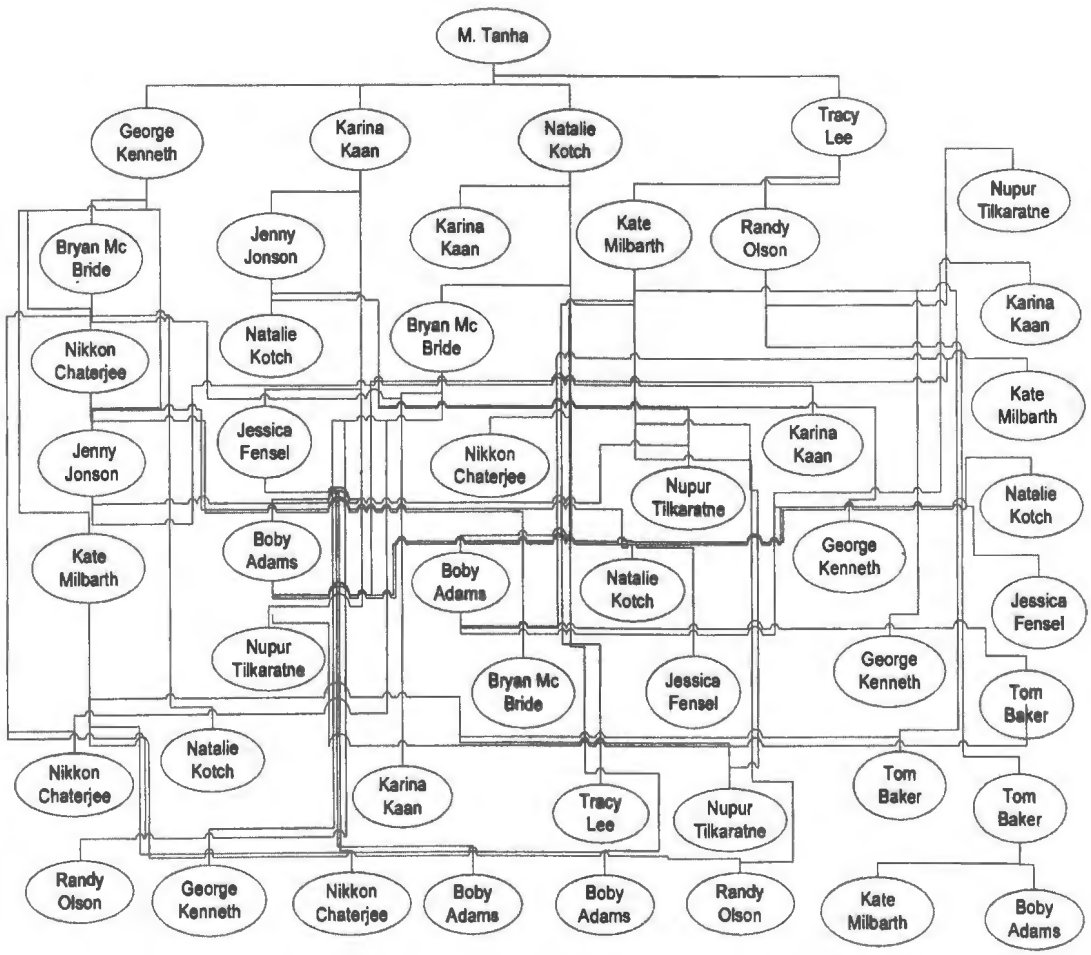


Figure 8. A Fictitious Tree of Friend of A Friend

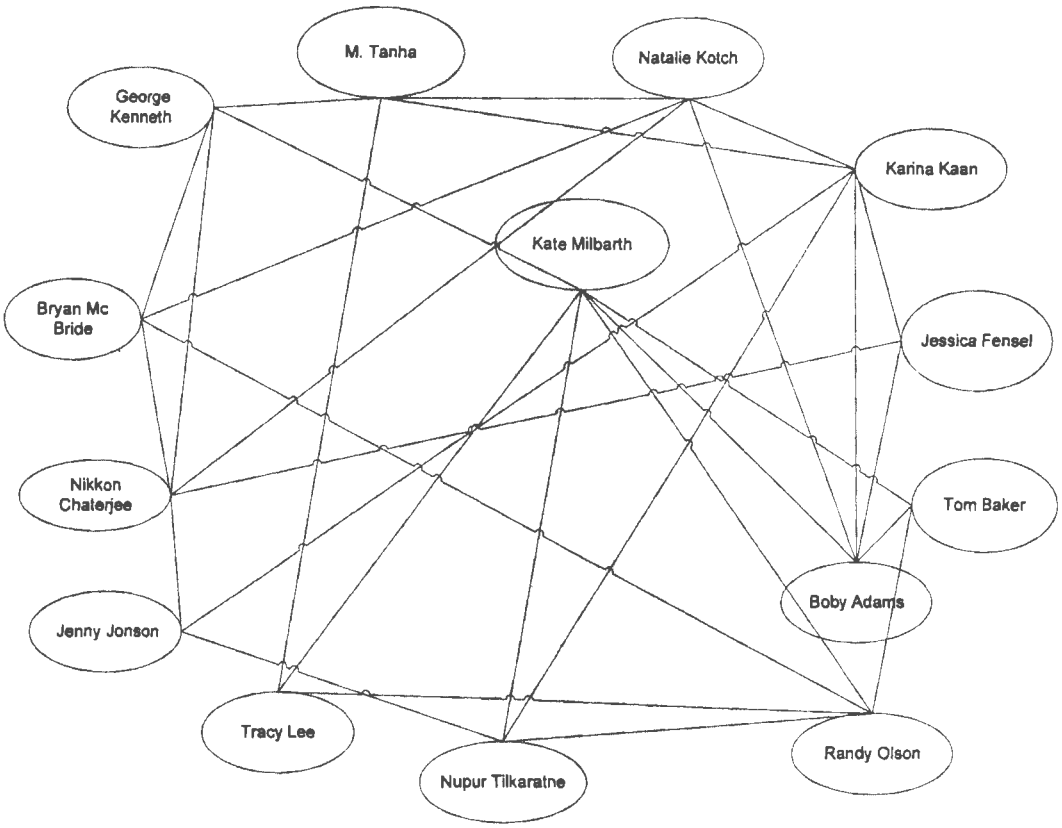


Figure 9. A Fictitious Graph of Friend of A Friend

4.1.2. Case Study 2: The Impacts and Effects of Global Warming

In the hypothetical Global Warming impacts and effects tree, we have several nodes like Flood, Drought, Earth Quake, Tsunami, Hurricane, Cyclone, Temperature Rise, Sea Level Rise, Glacier Retreat, Increased Evaporation and Extreme Weather. In this example we tried to maintain the hypothetical correlations for the impacts and effects of the global warming. The Iterative Deepening Search has found the relation among these nodes by going through Climate Change tree. In this scenario Climate Change is the root node in the tree and Temperature Rise is the final node and goal node of this search. Figure 10 and Figure 11 show the tree and graph for the Relationship of the Impacts and Effects of the hypothetical Global Warming:

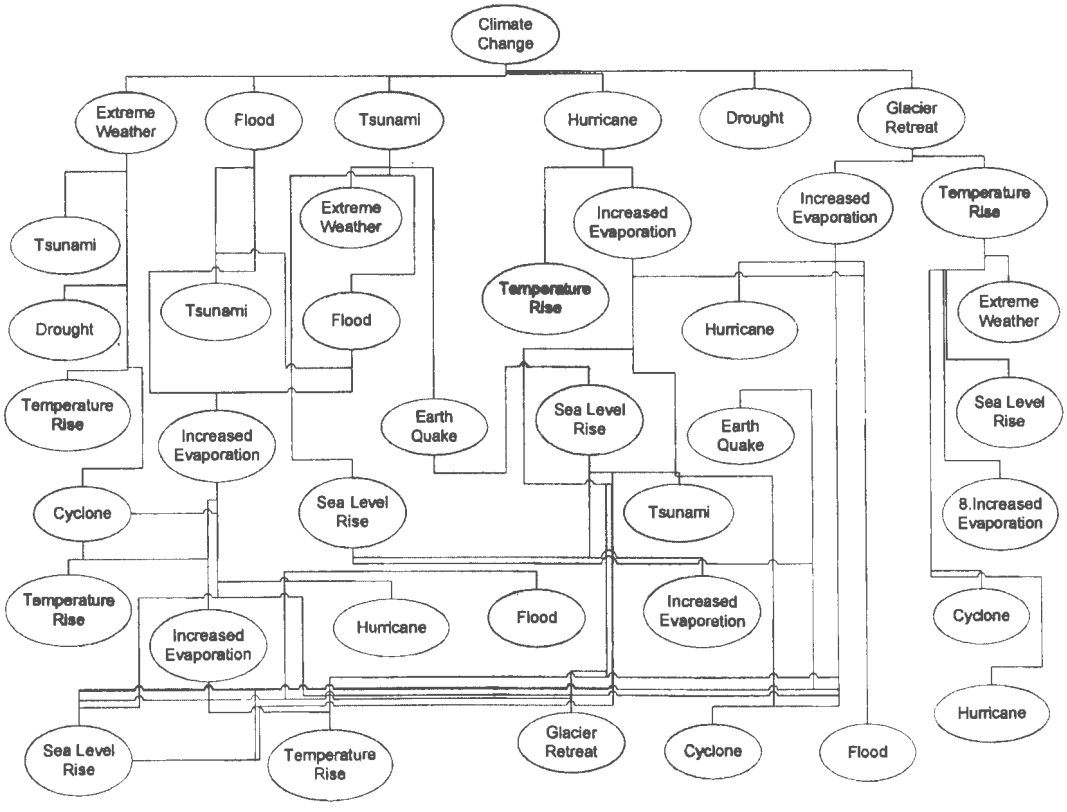


Figure 10. A Fictitious Tree of Impacts and Effects of Global Warming

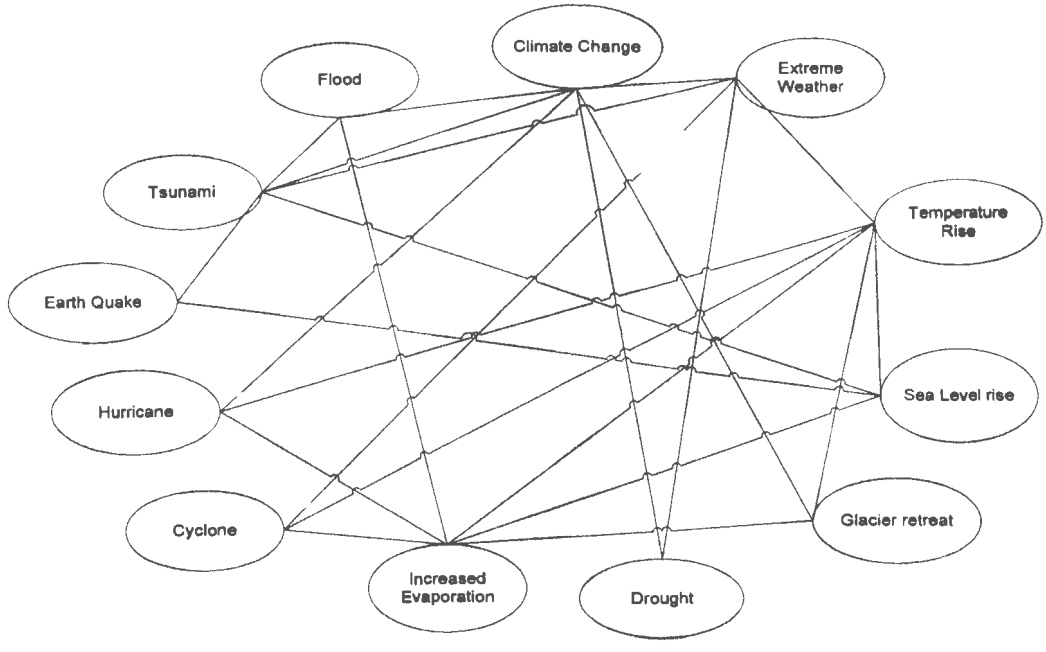


Figure 11. A Fictitious Graph of Impacts and Effects of Global Warming

4.1.3. Case Study 3: Terrorist Relation Discovery

In our last example, we have considered the example of a hypothetical terrorist group. All names of people, terrorist group, names of banks and country are fictitious. If we find a terrorist, then try to find who are the connected people with him or with him he is connected. In this hypothetical scenario, Hasanat Al Jamim is the first man. He is the root node; now our goal is to find with whom else he is also connected. Figure 12 and Figure 13 show how the Iterative Deepening Search has helped to find the connection among the terrorists.

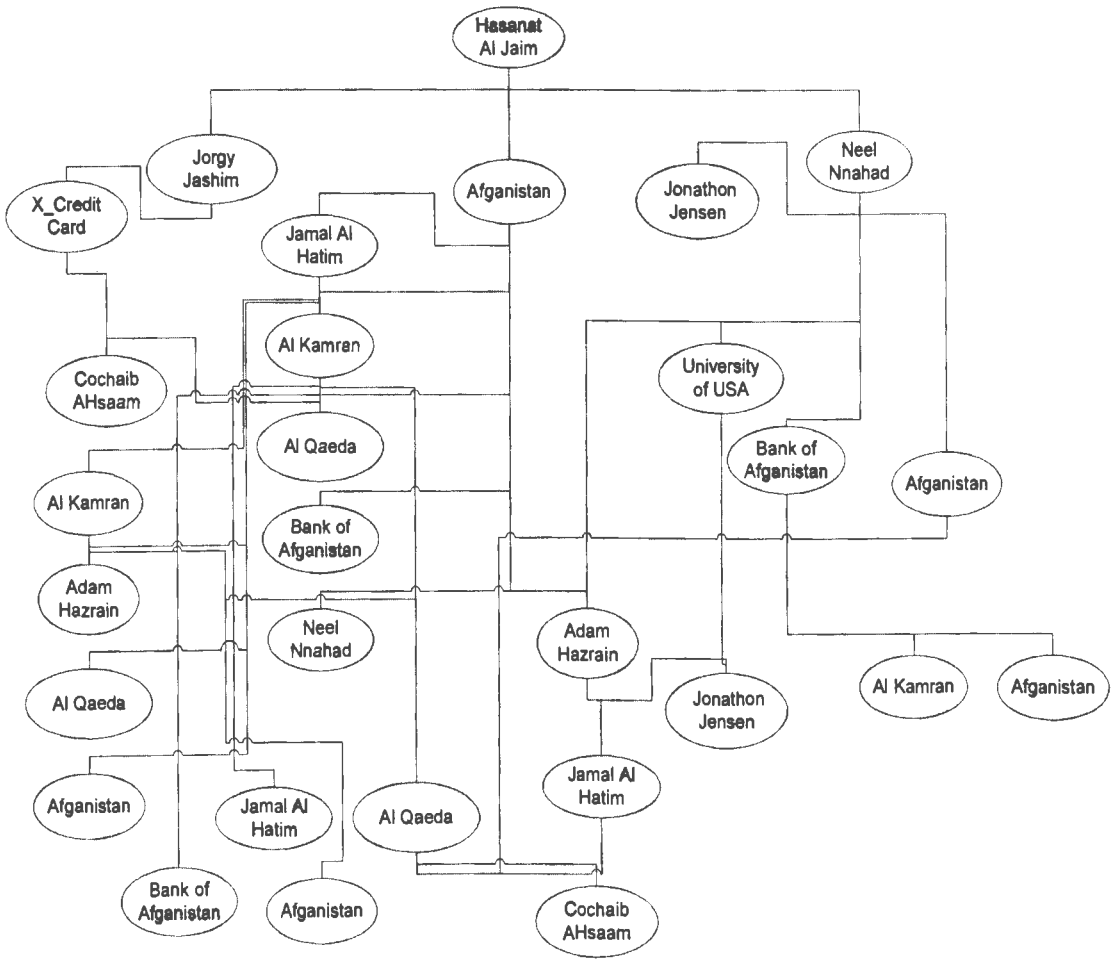


Figure 12. A Fictitious Tree for Terrorism Relationships

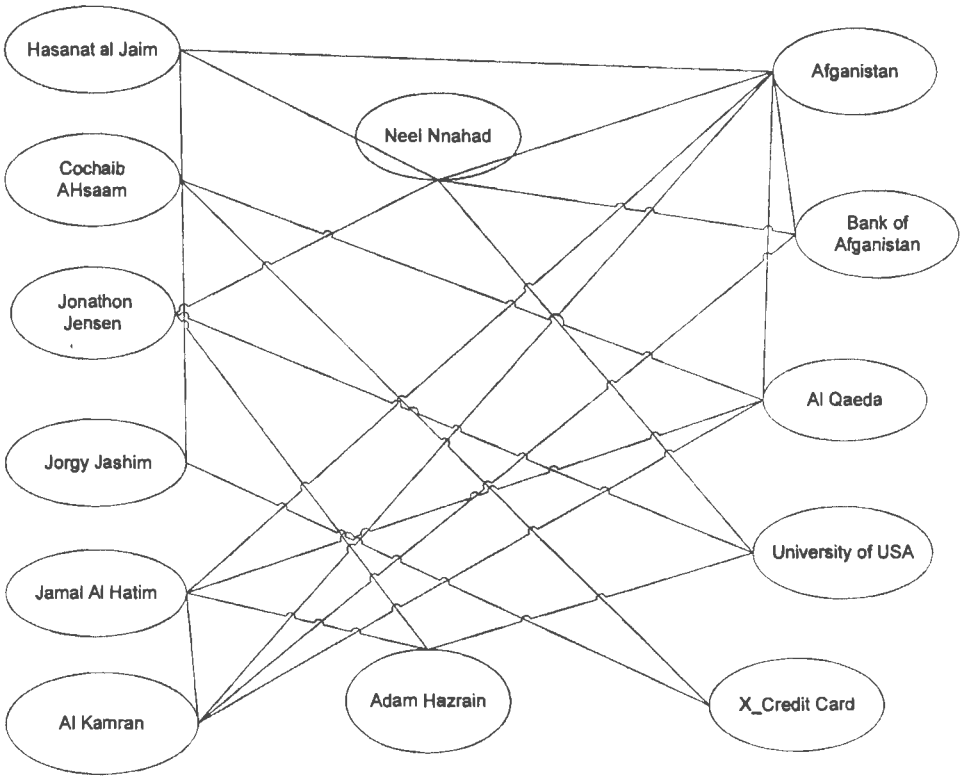


Figure 13. A Fictitious Graph of Terrorism Relationships

4.2. Analysis

4.2.1. Friend of a Friend (FOAF)

FOAF has become a widely accepted standard vocabulary for representing Semantic relations and many large social networking Websites use it to produce the Semantic Web profiles for their users [16]. There are millions of FOAF profiles hosted at a wide range of Websites. FOAF is frequently used as an example of the Semantic Web relation discovery schema, because it is very successful in terms of use [12].

FOAF satisfies the goal of using ontology to represent considerable amounts of distributed data in a standard form. However, FOAF truly serves as an example of the Semantic Web's full potential [18]. That means merging profiles of the same person from multiple social networking Websites and creating a large and unified social network from

sub networks that evolved independently. It is common for people to have accounts on several networks [12]. If the Semantic Web applications are built that use social networks, automated aggregation of a user's distributed social connections will give a fuller picture of their profile and improve the functioning of the applications [16].

4.2.2. Impacts and Effects of Global Warming

In the Semantic Web search, users are able to pose questions that involve exploring complex hypothetical relationships amongst many domains, in order to gain a better understanding of their domains of study and the interactions between them. Such relationships across domains, e.g., causal relationships, may not necessarily be hierarchical in nature. Such questions may involve complex information requests involving user defined functions and fuzzy or approximate match of objects, therefore requiring richer environment in terms of expressiveness and computation [2].

For example, a user may want to know "What are the effects of our environment that are causing global warming?" Answering such a question requires correlation of data from sources of the domain Climate Change and Disasters [1]. That type of correlation is only possible if, the user's notion of "cause" is clearly understood and exploited. This involves the use of ontologies of the involved domains for shared understanding of the terms and their relationships. Furthermore, the user should be allowed to express their meaning (or definition) of the causal relationship [1]. In this case it could be based on the proximity in time and distance between the two events (Climate change and disasters) and this meaning should be exploited when correlating data from the different sources. Subsequent investigation of the relationship by refining and posing other questions based

on the results presented, may lead the user to a better understanding of the nature of the interaction between the two events [2].

4.2.3. Terrorist Relations Discovery

Our experiment contains entities as well as relationships connecting the entities. An entity has a name and a classification (type). A relationship has a name and a vector of entity classifications, specifying the types of entities that are allowed participating in the relationship. Both entity classifications and relationships will be organized into their respective hierarchies [7]. The entity classification hierarchy represents the similarities among the entity classifications. In our third example, a general entity class "terrorist" may have subtypes of "planner", "assassin", or "liaison". The relationship hierarchy is intended to represent the similarities among the existing relationships. For example, "supports" is a relationship linking people and terrorist organizations. It is the parent of several other relationships, including "funds", "bank transactions", "citizenship status", etc [1]. We can look into the importance of the relationships between two people that went to the same university. A new relationship may emerge because of complex transitive relations connecting these two persons. Furthermore, the notion of importance depends primarily on the context, which assesses the risk of possible case of terrorist attacks [17]. In this example, it is not possible to encode all the relevant relationships as rules, because they are not usually known; yet they can be discovered through an analytical process. In general, the relevant relationships emerge as a set of connections or various interesting patterns of connections between the entities [11].

Different domains may have different notions of relationships. It is useful to use domain-specific ontology to guide the search for Semantic relations. As an example,

consider some of the people in the list who are the nationals of the same country and purchased something using the same credit card. In this scenario they may not have a known family relationship and one of them is on the FBI watch-list [1].

CHAPTER 5. CONCLUSION

Semantic relations involve by evaluating a set of contextually relevant paths of relations from one entity to another entity. By evaluating such paths we may identify relations based on connectivity or relations based on similarity of paths. This allows us to analyze sequences of binary relationships instead of just single relationships and manipulates these sequences to find similar entities as well as entities that may be connected or may not be directly connected [1] [2]. This technique is different from data mining that uses statistical techniques to find co-occurrence relationships between predicates based on patterns in data. [1].

The study attempted to present an appropriate graph search model to discover the complex Semantic Associations. This study focused on the problem by utilizing the notion of the Semantic Associations which aim to capture meaningful and probable complex relationships between entities in a large dataset of metadata based on a graph model. The proposed method is iterative deepening depth first search algorithm to find the complex Semantics on the Web. The study have proposed iterative deepening depth first search algorithm which discovers those complex Semantic relations. This proposal can be easily extended by using an efficient data management system in order to maintain a high performance.

5.1. Limitation

There were some limitations of this study. Ranking ontologies at the document level has been widely studied in the Semantic Web literature. But we were not able to do ranking the data, which was a limitation of this study. We collected random datasets from

the Web for our experiments. We created our trees and graphs and we tested our algorithm using these data. We did not have any real and solid datasets for our experiments, which constrained our results.

5.2. Recommendations for Future Work

This study came up with a prototype to discover complex relations in the Semantic Web. For future work, attention needs to shift from documents (e.g., searching for relevant documents) to integrated approach of exploiting data (content, documents) with knowledge. Relationships, their modeling, specification or representation, identification, validation or their use in query or information request evaluation are then the fundamental aspects of future study. Ontology from heterogeneous sources and means to support inspection of the explicit relations that makes a document relevant to the context is of further research. In future it will be better to include further extensive evaluation of the prototype that we developed, which will lead to better understanding of quality and scalability issues. For future experiments, creation of a user friendly graphical user interface is recommended to get better results.

REFERENCES

1. Sheth, A., Arpinar, I. and Kashyap, V. (2001) *Relationships at the Heart of Semantic Web: Modeling, Discovering and Exploiting Complex Semantic Relationships*. FLINT Book, LSDIS Lab, Computer Science Department, University of Georgia, Athens, GA
2. Sheth, A., Arpinar, I. and Kashyap, V. (2002) *Relationships at the Heart of Semantic Web: Modeling, Discovering and Exploiting Complex Semantic Relationships*. Technical Report, LSDIS Lab, Computer Science Department, University of Georgia, Athens, GA
3. Liu, L and Özsu, T. (2009) *Ontology to appear in the Encyclopedia of Database Systems*, Springer-Verlag, New York, NY
4. Nikravesh, M., Azvin,B., Yager, R., Zadeh, L. (2007) *Enhancing the Power of the Internet: Studies in Fuzziness and Soft Computing*, Springer-Verlag, New York, NY
5. Sheth, A. et al. (2003) *Semantic Association Identification and Knowledge Discovery for National Security Application*, Proceedings of the 15th international conference on World Wide Web, pages 407 - 416 Edinburgh, Scotland
6. Anyanwu, K. and Sheth, A. (2003) *ρ-Queries: Enabling Querying for Semantic Associations on the Semantic Web*, The Twelfth International World Wide Web Conference, Budapest, Hungary

7. Burns, P. et al. (2005) *An Ontological Approach to the Document Access Problem of Insider Threat*, IEEE Intl. Conference on Intelligence and Security Informatics Atlanta, GA

8. Aleman-Meza, B., Halaschek, C., Arpinar, B. and Sheth, A. (2005) *Context-Aware Semantic Association Ranking*, First International Workshop on Semantic Web and Databases, Berlin, Germany

9. Halaschek, C., Aleman-Meza, B., Arpinar, I. B. and Sheth, A. (2004) *Discovering and Ranking Semantic Associations over a Large RDF Metabase*, 30th International Conference on Very Large Databases, Toronto, Canada

10. Ding, L. et al. (2005) *Finding and Ranking Knowledge on the Semantic Web*, Proceedings of the 4th International Semantic Web Conference, 2005 Galway, Ireland

11. Golbeck, J., Mannes, A. and Hendler, J. (2005) *Semantic Web Technologies for Terrorist Network Analysis*, Emergent Technologies and Enabling Policies for Counter Terrorism, IEEE Press, Washington, DC

12. Golbeck, J. and Rothstein, M.(2008) *Linking Social Networks on the Web with FOAF*. Proceedings of the 17th international conference on World Wide Web (WWW2008), Beijing, China

13. Li, Y., Wang, Y. and Huang, X. (2007) *A Relation-Based Search Engine in Semantic Web*, IEEE Transactions On Knowledge and Data Engineering, Volume 19, number 2

14. Paliwal, A. , Adam, R., Xiong, H., Bornhovd, C. (2006) *Web Service Discovery via Semantic Association Ranking and Hyperclique Pattern Discovery*, Proceedings of

the 2006 IEEE/WIC/ACM International Conference on Web Intelligence, p.649-652, Hong Kong, China

15. Wolfe, J. and Russell, S. (2007). *Exploiting Belief State Structure in Graph Search.* Proceedings of the ICAPS 2007 Workshop on Planning in Games, Providence, RI
16. Finin, T., Ding, L., Zhou, L., Joshi, A.(2005) *Social Networking on the Semantic Web,* The Learning Organization, Volume 12, Issue: 5, Page: 418 - 435
17. Han, L., Sun. L., Chen, G. and Xie, L. (2006) *ADSS: An approach to determining Semantic similarity*, Advances in Engineering Software, Volume 37, Issue 2, Pages 129-132
18. Perry, M., Janik, M., Ramakrishnan, C., Ibanez, C., Arpinar, B. and Sheth, A. (2005) *Peer-to-peer discovery of Semantic Associations*, 2nd International Workshop on Peer-to-Peer Knowledge Management (P2PKM), La Jolla, CA
19. Russell, S., Norvig, P. (1995) *Artificial Intelligence: A Modern Approach*, (1st ed.), Prentice Hall, Upper Saddle River, NJ