DISCOVERING INFORMATIVE SUBGRAPHS IN RDF GRAPHS

by

WILLIAM H MILNOR III

(Under the Direction of John A. Miller and Amit P. Sheth)

ABSTRACT

In most contemporary approaches to pattern discovery in graphs, either quantitative anomalies or frequency of substructure is used to measure the relevance of a pattern. In this thesis, we address the issue of discovering informative subgraphs within RDF graphs. In the context of *Semantic Search*, relevance of such subgraphs depends on the amount of useful information conveyed to a user. This in turn depends on the meaning (semantics) of the edges in the subgraph. We introduce heuristics that guide a discovery algorithm away from banal (both low information and low relevance) paths towards more informative and relevant ones. This guidance is based on weighting mechanisms (driven by relationships) for the edges in the RDF graph. We present an analysis of the quality of the generated subgraphs with respect to path ranking metrics. We then conclude by presenting intuitions about which of our weighting schemes and heuristics produce higher quality subgraphs.

DISCOVERING INFORMATIVE SUBGRAPHS IN RDF GRAPHS

By

WILLIAM H MILNOR III

A.B., University of Georgia, 2001

B.S., University of Georgia, 2002

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GA

2005

DISCOVERING INFORMATIVE SUBGRAPHS IN RDF GRAPHS

by

WILLIAM H MILNOR III

Major Advisor:	John A. Miller
	Amit P. Sheth
Committee:	Hamid R. Arabnia
	Krzysztof J. Kochut

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
December 2005

DEDICATION

To my wife Emily and my son Haines for all their love and support…

ACKNOWLEDGEMENTS

Thanks to all SemDIS project members. I especially would like to thank Cartic Ramakrishnan and Matt Perry for their direct participation and assistance, as well as Boanerges Aleman-Meza for his insightful comments and revision suggestions. I would like to thank both Dr. Amit P. Sheth and Dr. John A. Miller for their wisdom and direction, as well as for the opportunities that they have each given me throughout my academic career. I would like to thank Dr. Hamid R. Arabnia and Dr. Krzysztof J. Kochut for being on my committee. I would further like to thank Dr. I Budak Arpinar, as well as Dr. Krzysztof J. Kochut for their suggestions and insight.

Research in this thesis was funded in significant part by the NSF Medium ITR Project "SemDis: Discovering Complex Relationships in Semantic Web, " (Grant number: IIS-0325464, PI: Dr. Amit Sheth, http://lsdis.cs.uga.edu/projects/semdis/) and the NSF Small ITR Project "Semantic Association Identification and Knowledge Discovery for National Security Applications," (Grant Number: 0219649, PI: Dr. Amit Sheth)[1]. We also thank Semagix, Inc. for the donation of the Semagix Freedom Toolkit, http://www.semagix.com/.

I must express the utmost gratitude to my wife Emily for all her support and motivation through my academic career, as well as to my son Haines for making me smile no matter how tough things can get.

[1] Views and conclusions presented here are those of the author and collaborators, and do not represent the views of National Science Foundation.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER 1

INTRODUCTION

As development of ontologies and utilization of standard languages such as RDF and OWL for knowledge representation is approaching widespread, industrial use, more attention is being given to the possibilities of the Semantic Web and technologies thereof. Large-scale applications have already been deployed in industry as well as research. Companies such as [Semagix], [Cerebra], and [Ontoprise] have developed commercially available products which utilize technologies and techniques from Semantic Web research. Developmental tools such as [Protégé] are receiving much attention, as are research based applications, resulting in prototypical applications such as Passenger Threat Assessment [Shethetal05], and commercially deployed applications such as Anti-Money Laundering and Know Your Customers [CIRAS]. Semantic technology has been defined as technology allowing execution time processing of the meaning of and associations between information [Polikoff03]. Such technologies represent meaning through explicit and implicit relationships, rather than focusing on objects and entities as in XML. Semantic Web technologies are those that exploit knowledge representation, ontology, identification and disambiguation, and reasoning over relationships for enhancement of current information systems.

A practical need for such technologies has arisen due to new opportunities in business markets [Sheth05]. Non-technical issues such as market need or readiness to accept new technologies has played an important role in industry adoption. One area in which the significant use of Semantic Web technologies has been proven is that of risk and compliance applications in industry. New governmental provisions, and the effort and resources required to comply, have revealed a need for Semantic Web technologies in risk and compliance. Since the September 11, 2001 attacks in New York, and the numerous corporate and financial scandals in recent years, this is especially the case in the areas of national security and financial crimes. With the passing of the Patriot Act in 2001 and the subsequent Sarbanes Act in 2002, the government and financial sectors have had to increase focus on the daunting tasks of assessing risk and compliance [Sheth05]. The necessity of this focus has been reiterated with the recent bombings in London. Such necessity presents an immediate practical application area for Semantic Web technologies. [Sheth05] discusses several specific applications of relevance in the area of semantic analytics.

In such applications, *Why* and *How* two entities are related are the crucial questions that must be answered; and discovering relevant sequences of relationships between two entities answers these questions. A discovery process therefore requires investigation of relationships between entities. To this end, we envision a system which *supports* its users in discovering ways in which a pair of entities are related. It is very likely that semantic search engines of the future will need to support such a discovery process. This perspective coincides with the Semantic Web vision [Berners-Lee01]. To this end, we investigate techniques that provide users with a chain of relationships between entities in response to queries of the following kind: *"What are the most relevant ways in which entity X is related to entity Y?"* The notion of relevance is critical to the definition of such a query. This becomes clear when one considers the *small-world*

phenomenon [Milgram67][Albert02]: given a knowledgebase and any two entities *X* and *Y* there may be a myriad of relatively short, insignificant chains (*i.e.,* six degrees) of relationships linking the two. Hence, the need for a way of semantically constraining and discovering the possible ways in which *X* and *Y* could be related.

In [Faloutsos04] the authors address this issue of information overload by developing an algorithm to extract relatively small, but most relevant subgraphs. They define the *Connection Subgraph Problem* as follows:

> **Given:** an edge-weighted undirected graph *G*, vertices *s* and *t* from *G* and an integer budget *b*
>
> **Find:** a connected subgraph *H* containing *s* and *t* and at most *b* other vertices that maximizes a "*goodness*" function *g(H)*.

However, the authors do not consider this problem with respect to semantics. We therefore adapt this approach in order to recast the problem of finding and ranking complex relationships between RDF resources (Semantic Associations [Anyanwu03]) into that of finding informative subgraphs composed of the most relevant.

The data set used in [Faloutsos04] is akin to a social network, and their weighting scheme is based on frequency of co-occurrence of names in Web pages. Clearly, this weighting scheme will not work for finding relevant subgraphs in RDF graphs, since the semantics of each property type in RDF is different. Therefore a systematic, way of weighting edges based on the semantics conveyed by the ontology represented using RDF schema [RDFS] is needed. To extend the approach in [Faloutsos04] to the more general case of an RDF graph, we propose heuristics for edge weighting that depend indirectly on the semantics of entity and property types in the ontology and on characteristics of the instance data. More specifically, we define *Class* and *Property Specificity, Instance Participation Selectivity* and a *Span Heuristic*. We evaluate the generated subgraphs using the path ranking schemes suggested in [Anyanwu05],[Aleman-Meza05],[Lin03]. Chapter 2 presents background and motivation for this work, and Chapter 3 presents related work. In Chapters 4 and 5, we discuss our heuristics and algorithms, respectively. This is followed by a discussion of the dataset for the respective experiments in Chapter 6. Chapter 7 presents the results and evaluations thereof. We conclude in Chapter 8 with a look at future research directions.

CHAPTER 2

BACKGROUND AND MOTIVATION

In this chapter, we present a brief discussion on background and motivation for our work in the SemDis project. We start with a discussion of ontologies and standard Semantic Web representation languages, as well as corresponding query languages and storage systems, and finish with a discussion on Semantic Analytics.

2.1 Ontology and Extraction

At the core of Semantic Web technologies is the concept of ontology. Gruber defines ontology as an explicit specification of a conceptualization, an abstract view of the world [Gruber03]. It represents agreement over, concepts and how they are related. Named relationships between concepts are at the heart of semantics [Sheth03]. Traditional search engines and applications have focused on entities, yet identifying meaning through relationships is how actionable information can be represented in the Semantic Web. By using agreed upon ontologies to annotate and interpret such relationships, an abstract view of the relevant domain can be acquired and utilized in practical applications.

In order to populate an ontology representing entities and relationships that exist across heterogeneous data, relational metadata must be extracted into a knowledge base or data store. Knowledge extraction can be done using semi-automatic [Handschuh02] techniques, with such tools as SemTag [Dill03] and CREAM [Handschuh03], and automatic [Hammond02] techniques, using SCORE or Semagix Freedom [Semagix].

Whether extraction is done automatically or semi-automatically, there are two central issues related to extraction: entity identification and entity disambiguation. Automatically identifying entity objects and relationships between them depicted in documents and web sources is necessary for obtaining meaningful information—this is especially the case when extracting from data sources that contain non-domain specific content. Entity disambiguation resolves the correspondence between the possibly many descriptions of the same entity in different sources, and is thus key for determining whether two entities sharing the same name are in fact the same entity. In order to analyze relationships existing between entities, the data must be integrated into a common format that can be utilized by a single application. The approach taken by Semantic Web technologies is to generate semantic metadata annotating heterogeneous data in order to represent a unified abstraction of the data [Sheth05], provided that the same ontology is used.

2.2 RDF and RDFS

Being that an ontology is a collection of concepts and relationships between said concepts, RDF and RDFS provide a natural fit for representing and querying semantic metadata. RDF provides a graph model, in which nodes represent entities and edges represent labeled relationships between entities, as well as a triple model, in which each triple represents a

statement (subject, predicate, and object) indicating an explicit relationship between a pair of entities [RDFSemantics]. Each of these models also allows for the expression of attributes about entities. RDFS (RDF Schema) is a semantic extension of RDF used to model vocabularies schemas. It allows for the description of concepts, groups of related resources, and possible relationships between these concepts as well as possible attributes on them. In our lab, we develop algorithms for reasoning over the RDF/S graph model. Such algorithms are integral to discovering semantic associations (discussed in section 2.5.1), and relevant subgraphs thereof, for semantic analytics (discussed in 2.5).

RDF and RDFS are recommended by the WWW Consortium, and various query languages as well as storage systems have been developed to interface with RDF/S data. Some such languages and storage systems are described below. For a more comprehensive survey on RDF query languages, the reader is referred to [Haase04].

2.3 RDF Query Languages

Population and maintenance of ontologies are not very useful with the capability to access and query the knowledge represented within. Below is a description of two of the most widely used RDF query languages, as well as an RDF language that currently has the status of a working draft. While each of these languages has been implemented in current systems, none of them provide a natural interface for complex relationship-based queries such as those for semantic associations [Anyanwu05]—finding all ways in which two entities are related including associations of arbitrary length.

2.3.1 RQL

RQL, developed by ICS-FORTH research labs, is a declarative language for querying RDF and RDFS resources [Karvounarakis02]. It provides an interface for navigating an RDF graph model, and for retrieving specific nodes and edges. The language relies on a functional approach based on Object Query Language (OQL). RQL supports generalized path expressions using both edge and node labels. Supporting such features is key for Semantic Web applications such as e-Marketplaces and Knowledge Portals [Karvounarakis02].

Basic RQL queries allow access of RDF descriptions irrespective of any corresponding schema. Navigation of class and property hierarchies, as well as properties between classes, in a schema(s) is also possible due to the rich set of operators provided by the language.

RQL queries utilize three basic statements: a SELECT statement indicating what to retrieve, a FROM statement which uses path expressions to bind variable names to specific locations in an RDF model, and an optional WHERE statement for constraining values of the bounded variable names.

2.3.2 RDQL

HP Labs has developed RDQL [Seaborne04] and has submitted it for W3C recommendation. RDQL is an evolution from multiple query languages, and it has been implemented in various RDF data storage systems.

The language accesses RDF resource descriptions using a graph model with an underlying representation as a list of triples. Following a SQL-like syntax, an RDQL query specifies what

to retrieve via a SELECT statement, what triple patterns to match via a WHERE statement, Boolean expressions over values of literals and URIs, and from which documents to retrieve data, via an optional FROM statement. Each clause in a WHERE statement is expressed as labels and variables for nodes and edges, hence the underlying triple-centric representation. By repeating variables for nodes in multiple clauses of a WHERE statement, an RDQL query can essentially allow the use of path expressions. However, to effectively utilize a complex WHERE statement, a user or application must have knowledge of the schema as such knowledge is not integrated into RDQL.

2.3.3 SPARQL

SPARQL is the query language part of a protocol and RDF query language for RDF data stores [Prud'hommeaux05]. The SPARQL description is currently in its third version as a working draft; however, it has already been implemented in the Corese Semantic Web query engine [Corese], as well as in the Redland RDF Application Framework [Redland]. The language is based on matching graph patterns in an RDF graph. It operates over a list of *subject, predicate, object* triples.

SPARQL provides a similar basic syntax as RDQL with SELECT and WHERE statements specifying what to retrieve and which patterns to match, respectively. (Other statements differ from those in RDQL, yet offer similar functionality). The language provides facilities to RDF data as URIs, blank nodes, and literals, as well as the ability to extract subgraphs and create new graphs based on queried graphs [Prud'hommeaux05].

2.3.4 Sample Queries

Table 2.1 gives a sample query for each of the 3 languages discussed above

Table 2.1 Example Query Syntax. Retrieve all researchers who authored a publication

Language	Query Syntax

RQL	`select RESEARCHER, PUBLICATION`
	`from {RESEARCHER} lsdis:authors {PUBLICATION}`
	using namespace `lsdis = http://lsdis.cs.uga.edu/sample.rdf#`

RDQL	`SELECT ?researcher, ?publication`
	`WHERE (?researcher lsdis:authors ?publication)`
	`USING info FOR <http://lsdis.cs.uga.edu/sample.rdf#>`

SPARQL	`PREFIX lsdis: http://lsdis.cs.uga.edu/sample.rdf#`
	`SELECT ?researcher, ?publication`
	`WHERE { ?researcher lsdis:authors ?publication }`

2.4 RDF Data Stores

With the proposal and recommendation of query languages for RDF, a natural successor is the development of storage systems which provide implementations of said languages. Below is an overview of three of the most popular RDF data storages, followed by a brief description of robust storage system, developed in the LSDIS Lab, which provides more suitable capabilities for semantic analytics.

2.4.1 Jena

Jena is a Java API and toolkit [McBride01] for storing and accessing RDF models, which complies with recommendations from the RDF Core Working Group [RDFCore]. The architecture provides main memory storage backed by a persistent triple store. The triples are stored in database tables for statements. For accessing the persistent triple store, Jena provides a SQL-based implementation of RDQL, offering compatibility with Oracle, MySQL, and PostgeSQL.

Jena utilizes the ARP parser for compliant parsing of RDF, and provides an API over the storage which is based on the graph model of RDF. Jena also provides multiple, flexible graph representation over its triple store [Carroll04], allowing high-level interface manipulation of RDF graphs. For reasoning over RDF, and even OWL, Jena provides simple manipulation of the underlying triples. Both forms of manipulation provide ample interfaces for programmers. External reasoning systems can also be integrated with Jena.

2.4.2 Redland

Redland is a set of modular, object based libraries written in C [Redland]. It provides APIs for manipulating RDF graphs and triples stored in main memory, database, and files.

Redland represents all concepts in the RDF model as well as additional concepts. It provides multiple layers in order to satisfy the requirements of various types of applications—the top layer is intended to be the main access point for such applications, but the lower layers can be interfaced as well [Beckett01]. These stacked layers were designed to cover the lower four layers of the "Semantic Web layer cake", and thus applications should interface with appropriate Redland layer.

For traditional querying of RDF data, Redland provides implementations of RDQL and SPARQL. It also offers other querying capabilities such as computing the neighborhood of a node. However, results of experimental evaluation presented in [Janik05] indicated that performance for such graph querying in Redland is quite poor.

2.4.3 Sesame

Sesame, developed by Aduna, is an open source Java-based application framework for RDF which supports inferencing over RDF Schema [Broekstra02]. Features include parsing and serializing various syntaxes for RDF implementations, as well as, implementations of three RDF query languages (RDQL, RQL and SeRQL). Further query integration is supported for Oracle, SQL Server, PostgreSQL, and MySQL. Flexible deployment of Sesame is provided through main-memory storage of RDF graphs, as well as, persistent storage capabilities of files and

databases. Through a flexible access API, Sesame offers support for both local as well as remote access over HTTP, RMI, and SOAP.

Whereas Jena and Redland offer a triple-centric representation of the RDF model, Sesame offers a node-centric representation from which a nodes neighborhood is computable, yet due to its implementation in Java imposing high memory costs, Sesame provides poor quality graph search capabilities [Janik05].

2.4.4 BRAHMS

BRAHMS [Janik05] is a C++ based main-memory storage system for RDF developed under the SemDis project [SemDis] in the LSDIS Lab at the University of Georgia. BRAHMS was developed to provide a high performance main-memory system supporting intensive querying of an RDF graph as required by algorithms for discovering semantic associations; thus, it provides a node-centric view of RDF as does Sesame. However, extensive experiments comparing BRAHMS to the above RDF storage systems illustrated faster graph querying functionality offered by BRAHMS.

Currently, BRAHMS offers read-only capability for storing RDF data, but it is being extended to provide modification capabilities. It will provide a Java-based interface to the underlying graph model implementing the interfaces specified in the SemDis API [SemDisAPI], thus providing a high-quality main-memory storage system for ongoing research in semantic discovery.

2.5 Semantic Analytics

Systems that go beyond basic search and integration capabilities by offering users an interface for performing ontological computation and formulating complex relationship type queries are becoming extremely valuable [ShethDROPS05]. Such systems will incorporate capabilities to automatically perform such computations and queries over semantic metadata. Automatic analysis of semantic metadata requires searching and mining through possibly multiple heterogeneous data sources in order to uncover meaningful complex relationships connecting entities within and across the data. Uncovering such relationships can become very intensive considering the potential size of an ontology. For instance, consider the SWETO ontology [Aleman-Meza04]. SWETO is a real-world ontology containing more than 800,000 entities and over 1,500,000 explicit relationships between them. Sample queries for complex relationships in SWETO have generated results in the thousands. SWETO is a sparsely connected graph, and in a highly connected graph, even smaller data set could result in millions of results. Thus, care must be taken to obtain the information most relevant to a query.

2.5.1 Semantic Associations

Within semantic metadata, a pair of entities can be connected by multiple complex relationships. As RDF captures the meaning of entities by relating them to other entities, it is a natural fit for storing and acquiring such connectivity. Such relationships are a fundamental aspect for the Semantic Web [Thacker03]. We define these complex relationships as “Semantic Associations” [Anyanwu03]. While these associations can be very simple, such as explicitly stated relationships, they can also contain various intermediate entities forming longer, complex

relationships. A Semantic Association is defined between two entities as a semantic path of alternating entities and relationships. There may be a multitude of such associations between any given pair of entities. There are two methods for uncovering what is meaningful (semantic associations) through semantic analytics:

- association identification, and
- association discovery

Further determining which associations are considered meaningful is at the heart of semantic analytics.

2.5.2 Association Identification

To identify meaningful associations, a query processor must be given predefined rules which indicate *a priori* what is considered meaningful. Such rules can be expressed as patterns of relationships in the data ("a person is a resident of a country that harbors a known terrorist group" may be a meaningful association in terrorist threat assessment applications), or as inference rules (if a charity maintains bank accounts in a bank based in a blacklisted country, and the charity is associated with blacklisted people, then that charity is funding terrorist activities). Current commercial solutions, such as Know Your Customer or Anti-Money Laundering [CIRAS] support such capability. Rule-based analytics, however, require explicit knowledge by the user of patterns or rules considered meaningful. Having such knowledge for a diverse, complex domain is infeasible, especially when data is distributed over multiple heterogeneous data sources. To this end, automating association identification will ease the burden paced on the analyst of checking all patterns of interest.

Furthermore, an analytic system may incorporate ranking techniques to determine the relevance of associations matching the specified patterns. For compliance in some of the aforementioned application areas, ranking the results of an analyst's query is required. For instance, the EU Third Money Laundering Directive requires banks to take a "risk sensitive" approach to Customer Identification Program applications. This means that banks must verify the identity of beneficial owners in an order based on which beneficial owners pose a greater risk in comparison with others.

2.5.3 Association Discovery

Discovery of meaningful associations involves mining the relevant data in order to find associations that are meaningful in respect to a given domain. Data mining is the discovery of implicit patterns, rules and complex relationships residing in large data sets. In defining semantic associations, [Anyanwu03] defines a *Property Sequence*, between two nodes *x* and *y*, as a finite sequence of properties (labeled edges) in which the properties express a uniform directionality between *x* and *y*. Thus, in each pair of connected nodes in the sequence, each node is the origin of one property and the terminus of another property[2]. Based on the definition of a *Property Sequence*, [Anayanwu03] describes three types of associations which are relevant to the context of this thesis:

1. Two nodes *x* and *y* are *ρ-pathAssociated* if there is a sequence of properties between them such that all properties in the sequence belong to the set of properties defined in

[2] Either *x* is the origin of the first property in the sequence and *y* is the terminus of the last property, or *x* is the terminus of the first property and *y* is the origin of the last property.

the corresponding schema(s), and either *x* is the origin and *y* is the terminus or visa versa.

2. A pair of nodes *x* and *y* are *ρ-joinAssociated* if there exist two property sequences, ps_1 and ps_2, which are joined at an intermediate node *n*, and either *x* and *y* are the respective origins of ps_1 and ps_2 or *x* and *y* are the respective terminus for ps_1 and ps_2.
3. A pair of nodes *x* and *y* are *ρ-IsoAssociated*, if there exists two property sequences ps_1 and ps_2 representing the same pattern of properties (labeled edges), and *x* and *y* are either the respective origins or the respective terminuses.

These definitions were refined in [Shethetal05] to relax the restriction of directionality on the properties:

1. Two entities are *semantically connected* if there exists an alternating sequence of properties and entities (a *semantic path*) connecting them.
2. Two entities are *semantically similar* if there exists a pair of matching property emanating from them.
3. Two entities are *semantically associated* if they are either *semantically connected* or *semantically similar*.

Corresponding techniques for mining such associations can be extremely useful when no predefined rules are specified. However, data mining is limited by the fact that it cannot reveal the significance of an association. This limitation can be overcome by incorporating ranking schemes and/or heuristic based search methods. Developing such techniques for relevant association discovery in a semantic web is the basis of the SemDis project in our Lab.

2.5.4 Ranking

To aid the analyst in determining what is meaningful in a mountain of information, ranking techniques can be implored. Such ranking can be performed based on user criteria. For instance, an analyst may specify that associations spanning many entities are more important than those spanning few entities. This may be the case in an application for anti-money laundering where money from a certain activity may change hands many times over in order to elude detection. Another way for an analyst to specify what may be meaningful is through the use of a notion of context [Aleman-Meza03]. For instance, an analyst assessing the threat posed by an airline passenger may indicate that associations involving a context of aviation training or simulation are of greater importance than other associations. Although such criteria may be specified prior to query processing, criteria based ranking differs from rule-based techniques in that results are not ordered based on specific patterns or inference rules.

In [Aleman-Meza05], the authors discuss a ranking scheme for semantic associations which offers an analyst multiple criteria for ranking semantic associations. Said criteria are categorized into two types of groups: semantic and statistical. The semantic metrics described are *context*, which are ontological regions defined as groups of concepts and relationship, *subsumption*, which pertains to the hierarchical classification of concepts[3], and *trust*, which is determined by how trusted is a source from which entities are extracted. The statistical metrics discussed are *rarity*, which is defined as how rare entity types and explicit schema properties are in the corresponding dataset, *popularity*, which relates to how many relationships involve a specific node (nodes with high in-/out- degrees), and *association length*. Each of these criteria can be given a weight indicating how important a criterion is with respect to one another.

[3] The authors only discuss the *rdfs:subClassOf* relationship with no consideration to the *rdfs:subPropertyOf* relationship.

Techniques based on non user specified criteria are also in use. These techniques typically involve ranking based on novelty or rarity of an association [Lin03]—looking for anomalies in the data. For instance, an analyst using an application for passenger threat assessment may perform a query involving an architect with citizenship in Saudi Arabia. The analyst should be more concerned about results connecting the pilot to a flight school in the U.S. rather than those connecting him to an urban technical school. However, novelty of associations may not be of interest in applications for anti-money laundering in which malicious individuals would want their relations to one another to seem common.

Other non user specified criteria based ranking techniques incorporate heuristics in determining order of relevance. The authors of [Guha03] discuss a need to determine an order in which to serialize a resulting discovered subgraph. They describe structural based heuristics, in which all property types are considered equally relevant, using thresholds on: number of triples with the same source and edge label, and number of triples with the same source; as well as semantically based heuristics, i.e., considering what types (sub classes) of person for which the person is searching and general knowledge of such types.

[Anyanwu05] provides the user a modulative relevance model for specifying a conventional search mode or a discovery style search mode. This relevance model is then combined with semantic and information theoretic techniques, as well as, heuristics to determine in what order to rank the resulting semantic associations. For instance, obscure and unpredictable results are considered significant, and therefore ranked high, in the discovery mode, yet the same results would be ranked very low in respect to the conventional search mode. In [Anyanwu05], two of the factors which determine the rank of a semantic association are: 1) the probability of occurrence of a labeled edge in the association such that less probable edges indicate greater information gain, and 2) refraction of an association which determined by how much it deviates from possible paths represented in the corresponding schema(s). Aside from specifying a relevance model, a user of SemRank can add keywords to a query increasing the rank of semantic associations which contain a property with a label matching the keyword—this is the final factor used in determining the SemRank value for a semantic association.

2.5.5 Heuristic Based Discovery

Due to the possible enormity of these result sets, processing analytic queries can be quite intensive with respect to technical resources requiring excessive time or memory. In [Alesso04], the authors discuss two groups of search algorithms: uninformed and informed. Traditional uninformed (or exhaustive) search techniques such as *depth-first search* and *breadth-first search* are inefficient due to high costs of processing time (*depth-first search*) and memory allocation (*breadth-first search*)—both algorithms are in EXSPACE. Thus, informed (or heuristic based) search algorithms can reduce the risk of such intense resource allocation by using *a priori* knowledge to guide the search. For an example algorithm, see the description of [Eliassi-Rad05] in chapter 3. Heuristic based approaches are similar to rule-based approaches in that certain associations can be ignored during processing. However, where rule-based applications will ignore an association as soon as it deviates from a given rule, heuristic-based applications will use other factors in determining when to discontinue analysis of an association.

While such algorithms are more efficient in terms of resource allocation, the down-side is incompleteness of search results. While pruning a search space, results that are actually important to the analyst may be determined irrelevant by the heuristics such that what is returned

to the user is not contextually relevant. Thus, care must be taken in how the heuristics determine relevance while performing a search. In this paper, we discuss an algorithm which incorporates multiple heuristics; and we developed a system based on this algorithm in which an analyst can configure which heuristics are used and the importance thereof.

2.6 Semantic Visualization

The ability to browse and visualize an ontology is crucial to semantic analytics [Sheth05]. Current tools such as Protégé and Semagix Freedom provide interfaces for schema creation which allow a user to graphically visualize and manipulate the hierarchical structure of classes and properties in a schema. Figure 2.1 [ShethandAvant04] illustrates a graphical user interface for Semagix Freedom in which a user can define the structure of schema classes (in the left-hand pane) as well as specify possible relationships and attributes of a class (in top and bottom right-hand panes, respectively). Such capabilities assist a user to gain a cognitive understanding of schema structure.

A more crucial capability for semantic analytics is the ability to visualize search results. While automatic analysis can reveal meaningful associations in information, it will be useful for an analyst to have the ability to further browse the ontology based on the results. Furthermore, an ability to actually navigate the individual links is crucial to gain further insight. Figure 7.12 illustrates a TouchGraph [TG] representation of the results of an analytic query. Such technology provides an analyst with an interface to visualize multiple semantic associations that exist between two given entities. As any two given entities can be connected by an enormous number of semantic associations, the ability to present the analyst with a subset containing the most relevant associations is the motivation behind the work presented herein.

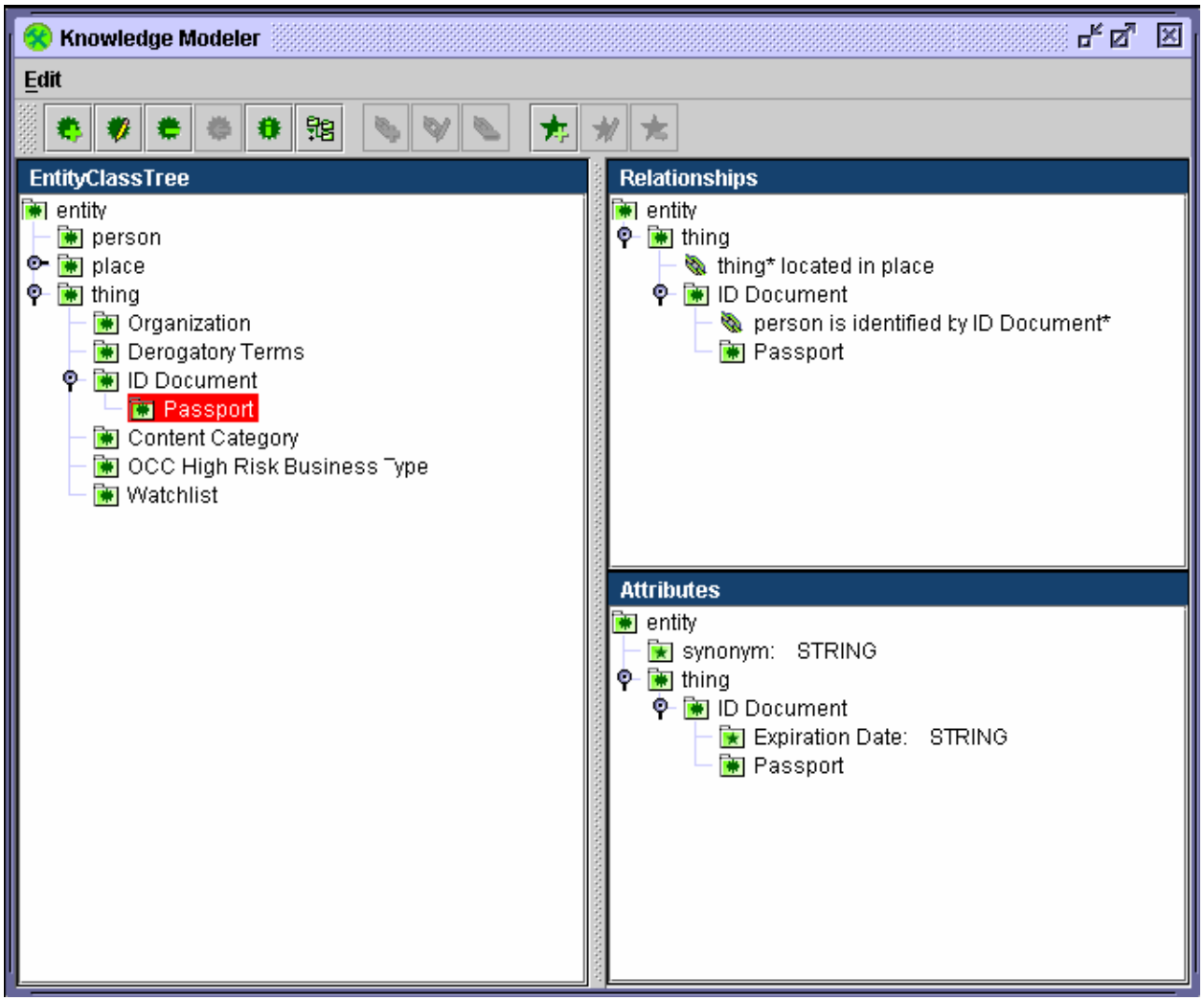


Figure 2.1 Semagix Freedom's Knowledge Modeler [ShethandAvant04]

CHAPTER 3

RELATED WORK

Reasoning and knowledge discovery over graph data models has been studied in the Graph mining community and more recently in the context of the Semantic Web. The remainder of this section highlights work which is most relevant to ours.

The work most directly related to graph-based knowledge discovery and reasoning for the Semantic Web is that of Semantic Associations. To the best of our knowledge this is the only existing work of this type. The nature of web data [Albert02] often leads to an overwhelming amount of associations between two entities. To combat this problem, [Anyanwu05][Aleman-Meza05] propose to rank Semantic Associations. As an alternate approach, the method in [Mukherjea04] filters the search space before computing associations. They adapt Kleinberg's hub and authority scores [Kleinberg99] to compute importance of Semantic Web resources and then only consider nodes with importance greater than some threshold when computing Semantic Associations. Their preprocessing step based on importance thresholds is likely to discount those paths that contain even a single unimportant node. Our approach to this problem is fundamentally different from these two. We try to find the 'best' set of associations which contain a visually[4] comprehendible number of resources.

There has been a considerable amount of work done in the field of graph mining to detect patterns in graphs. Patterns discovered are characterized either by their anomalous nature or frequent occurrence, among other things. Efficient algorithms have been developed for many variations of the frequent subgraph discovery problem [Yan03][Huan04][Kuramochi02]. Community and group detection is another well-studied graph mining problem which attempts to discover communities and groups based on link analysis. The problem has been studied on both the web graph [Flake02][Gibson98] and other data sets [Adibi04]. These graph mining problems focus on graphs with single node types and single edge types, however. For the Semantic Web we need algorithms which take into account the semantics of different node and edge types. *Novel Link Discovery* was introduced in [Lin03] and involves finding *novel*[5] paths between entities, *novel* loops, and significantly connected nodes. The methodology used in this work considers different node and edge types but differs from ours in that importance is determined purely from rarity. Thus, in the context of a semantic web, novel links would be those links that display rare patterns of relationships (labeled edges) or types of semantic associations. Also the paths examined in [Lin03] are considerably shorter than the ones we examine. In [Eliassi-Rad05], the authors discuss pruning a potentially large search space using a heuristics-based search method. The heuristics they describe use frequency statistics for node types and edge types in order to define a probability model for measuring the uncertainty of an edge's occurrence in the semantic graph." The heuristics attempt to find only less *likely* paths between a pair of nodes.

---

[4] A visually comprehensible number of resources is a graph which can be easily visualized on a single screen. See Figure 7.12 for an example query result which can be entirely visualized on a single screen.

[5] [Lin03] defines *novel* links as connections that may indicate previously unkown, but significant kinds of connections.

CHAPTER 4

ALGORITHMS

In [Faloutsos04], the authors develop two algorithms which take as in put a weighted graph to obtain a display graph connecting two resources. For our purposes we compute weights on an RDF graph (as explained in chapter 5). The authors present an algorithm for extracting a so-called *candidate graph* from an input graph. They also propose an algorithm based on electrical circuits to extract a *display graph* from the *candidate graph* for a given budget *b.* Hence, the *goodness* function—mentioned in the *Connection Subgraph Problem* statement—is a measure of maximum delivered current through the electrical network that is present in a display graph of *b* nodes. For our purposes we refer to these as *Candidate ρ-graph* and *Display ρ-graph*, respectively. We therefore define a *ρ-graph* as a graph composed of the edges and entities which appear in (a subset of) the semantic associations connecting a pair of entities[6]. We assume that the properties (edges) in the RDF graph are undirected. Consider a query which asks to find the relevant ways in which entity *Y* is related to entity *X*. We make this assumption of undirected edges to prevent the exclusion of a path of the form $X \xrightarrow{p_1} \ldots \xleftarrow{p_i} \ldots \xrightarrow{p_n} Y$, where $1 < i < n$. By relaxing the directionality restriction on edges, we obtain a graph in which edge distances are symmetric, i.e. *d(u,v)* is equal to *d(v,u)*, as are edge weights, i.e. *w(u,v)* is equal to *w(v,u)*[7].

4.1 Candidate ρ-graph generation algorithm

The *candidate ρ-graph* generation algorithm is based on a notion of distance between two nodes. The algorithm grows a set *S* around the source node *s* and a set *T* around the sink node *t* (*s* and *t* are referred to as the roots of their respective sets) until a certain threshold is met: a maximum number of total nodes or maximum number of cut edges between *S* and *T*. At each iteration, a pending list is maintained for each of these sets which consists of those nodes $n \notin S$ and $n \notin T$ and adjacent to some node $n' \in S \cup T$. The sets *S* and *T* are expanded by choosing from the pending list the node with shortest distance to either *s* or *t*. For an edge *(u, v)* the distance between *u* and *v* is given by the following formula, and the length of a path is the sum of the distances between its edges:

$$distance(u,v) = log\left(\frac{(degree(u) + degree(v))^2}{w(u,v)^2}\right) \quad \textbf{(1)}$$

where *w(u,v)* is the weight on *(u,v)*.

The aim of our initial experiments is to determine the quality of the *Candidate ρ-graph* in terms of its ability to capture the best paths between the query endpoints.

---

[6] A *ρ-graph* must contain the two endpoints for the respective semantic associations.

[7] This is not strictly the case when the SPAN heuristic is used, as discussed in section 5.3

4.2 Display ρ-graph generation algorithm

The display graph generation algorithm prunes the generated candidate graph down to a smaller size while ensuring that the resultant pruned graph conveys maximum information. In [Faloutsos04] the authors present a rather elegant solution to this by modeling the graph as an electrical circuit where the edge weights represent the conductance values in the circuit. They use the fact that current flows from high voltage to low voltage, to impose direction on an otherwise undirected graph. By modeling the candidate graph as an electrical network, where the first entity in the query is the source *s* and the second entity is the sink *t*, information flow through the graph is analogous to the flow of current through an electrical network from *s* to *t*. Using Ohm's law and Kirchoff's law, a system of linear equations is created with voltages at each node as a variable in these equations. Edge weights are analogous to edge resistance in an electrical network.

Let *R(u,v)* be the resistance of edge *(u,v),* let *I(u,v)* be the current flow from *u* to *v* and let *V(u)* be the voltage at *u*. We then have Ohm's Law:

$$\forall u, v : I(u,v) = \frac{(V(u) - V(v))}{R(u,v)} \qquad \textbf{(2)}$$

giving the relationship between voltage, current, and resistance, and Kirchoff's Law:

$$\forall v \neq s,t : \sum_{u} I(u,v) = 0 \qquad \textbf{(3)}$$

stating that the current flowing into a node must equal the current flowing out of a node ultimately giving us the system of linear equations:

$$V(u) = \sum_{v} \frac{V(v)R(u)}{R(u,v)} \qquad \forall u \neq s,t \qquad \textbf{(4)}$$

where

$$R(u) = \sum_{v} R(u,v) \qquad \textbf{(5)}$$

is the total resistance of edges incident on *u* and

$$V(s) = 1, \quad V(t) = 0 \qquad \textbf{(6), (7)}$$

Solving this system of equations gives voltages at each node and currents on each edge. The voltages and currents are then used to flow along paths from sources to sink. This step takes $\Theta(n^3)$ time, which motivates the need for the Candidate graph generation process. The greedy display generation algorithm attempts to find a display graph of at most *b* (set to 100 in our experiments) nodes which maximize the amount of total current delivered from the start node to the end node. Starting with an empty subgraph, this algorithm iteratively adds paths until meeting the budget *b*.

At each of the iterations, a dynamic programming algorithm is used to make the greedy choice of which path to add to the subgraph. The greedy choice is the path which has the maximum ratio of delivered current to number of new nodes added to the subgraph. Delivered current is defined as follows:

$$\hat{I}(s,u) = I(s,u) \tag{8}$$

$$\hat{I}(s = u_1,...,u_i) = \hat{I}(s = u_1,...,u_{i-1}) \frac{I(u_{i-1},u_i)}{I_{out}(u_{i-1})} \tag{9}$$

Where

$$I_{out}(u) = \sum_v I(u,v), \qquad \forall v : V(u) > V(v) \tag{10}$$

In our experiments we test the model based on current flow used to compute these display graphs.

4.3 System Architecture

Figure 4.1 illustrate the architecture of system which supports querying an RDF metadata store using the above described algorithms. Two points are of significance: the metadata store is annotated by multiple schemas (which is the motivation for our Span heuristic presented in section 5.3), and the returned results are in the form of a subgraph which can be visualized by an analyst. The system operates by allowing an analyst to specify a pair of entities, as well as, the desired settings for the heuristics, which are then fed to the algorithms for discovery of a relevant subgraph.

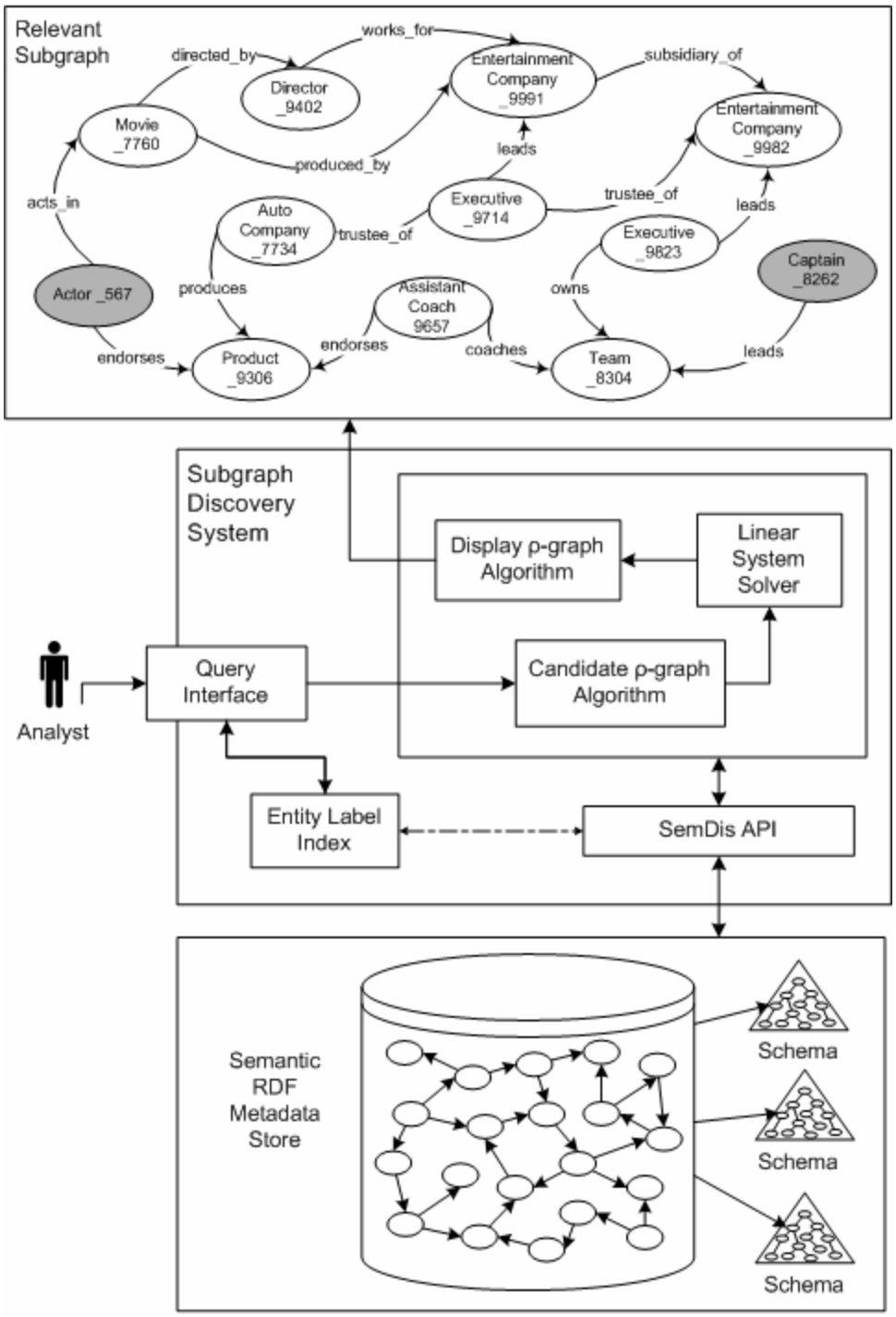


Figure 4.1. System Architecture The heuristic based algorithms access RDF data through the SemDis API. Results are then returned as a visualizable subgraph.

CHAPTER 5

HEURISTICS

RDFS vocabulary allows users to represent classes and properties thereby indirectly imposing meaning on resources defined by the types of relationships in which a resource can be involved. Hence we define three quantities indirectly based on semantics and RDF statement types and frequencies. Our aim in doing this is to use semantics to compute edge weights thereby guiding the algorithm in the subgraph discovery process. We formally define a schema S as the union of the following sets:

$$C = \left\{ c \middle| \left\langle c, rdf{:}type, rdfs{:}Class \right\rangle \right\}$$

$$P = \left\{ p \middle| \left\langle p, rdf{:}type, rdf{:}Property \right\rangle \wedge \exists c, c' \in C \middle| c \in rdfs{:}domain(p) \wedge c' \in rdfs{:}range(p) \right\}.$$

Further, we define an RDF data store $R = \langle \Pi, I \rangle$ where $\Pi = \bigcup S$ and I is the set of corresponding instance triples:

$$I = \left\{ \langle s, p, o \rangle \middle| \langle s, rdf{:}type, C_1 \rangle, \langle p, rdf{:}type, rdf{:}Property \rangle, \langle o, rdf:type, C_2 \rangle \right\}$$

We assume a resource that is classified as an instance of classes belonging to different schemas in our data set is uniquely identified by its URI. In other words, no data integration operation is required.

5.1 Class and Property Specificity (CS and PS)

Intuitively more specific resources and properties convey more information than general ones. For instance, a schema may contain a high level class named *Person* with many subclasses such as *Employee* and *Trustee*. In such a schema, classifying an entity as a *Trustee* conveys more information about the entity than classifying it as a *Person*.

As a result of the *rdfs:subClassOf* and *rdfs:subPropertyOf* properties provided by RDF schema it is possible to impose a partial ordering of properties and classes in the schema resulting in a well formed hierarchy of classes and properties. For a given property p_i, let $d(p_{iH})$ be the length of the longest path in the hierarchy tree that contains p_i, and for a given class c_j, let $d(c_{jH'})$ be the length of the longest path in the hierarchy tree that contains c_j. Properties and classes at the root of their respective hierarchy trees in the schema are considered most general while those at the leaves of these trees are considered most specific. Therefore a measure of specificity can be associated with each class or property commensurate with its position in its hierarchy. Let the depth of an arbitrary property in its property hierarchy be $d(p_i)$ and the depth of an arbitrary class in its class hierarchy be $d(c_j)$. Therefore, the specificity of property p_i and class c_j are given by

$$\mu(p_i) = \frac{d(p_i)}{d(p_{iH})} \qquad \mu(c_j) = \frac{d(c_j)}{d(c_{jH'})} \tag{11}$$

Every resource that is an instance of the class c_j is assigned the weight $\mu(c_j)$. If a resource r is an instance of k distinct classes it is assigned the value $\mu(r) = \max_{1 \le x \le k}\{\mu(c_x)\}$. To convert this node weight into an edge weight, the value is equally distributed among all edges incident on the resource r. This weighting scheme favors nodes with lower degree[8] since the node specificity is divided equally among its incident edges, therefore edges incident on nodes with high degree will get a lower weight. We feel that nodes involved in fewer relationships convey more specific and hidden information.

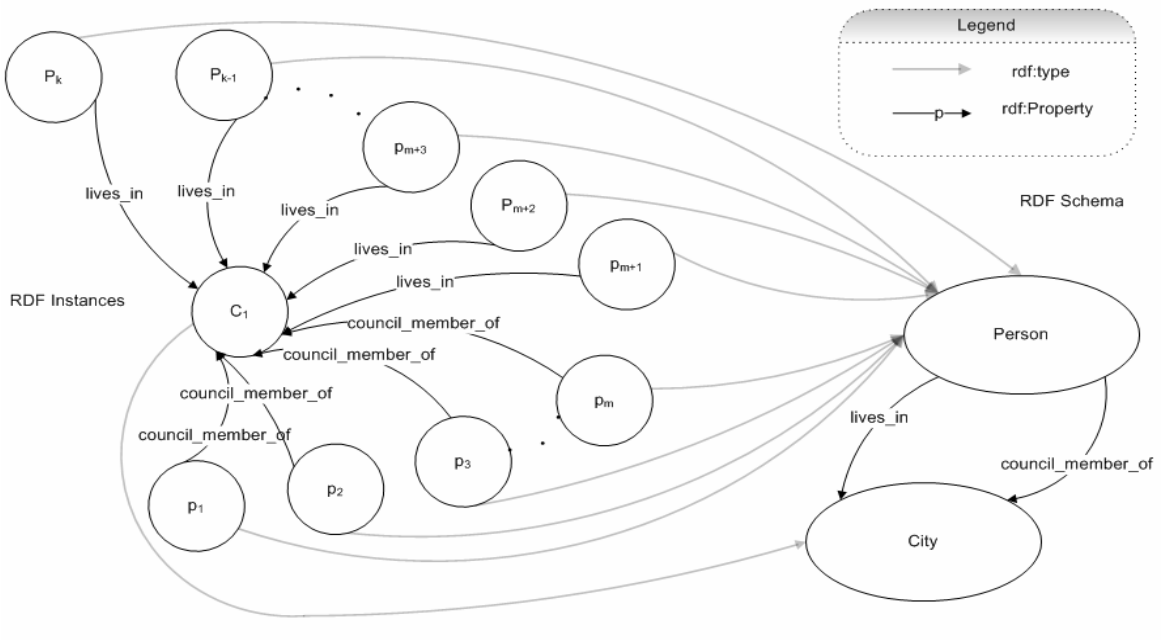


Figure 5.1 Instance Participation Selectivity Illustrative example for Instance Participation Distribution Instance Participation Selectivity (IPS)

5.2 Instance Participation and Selectivity (IPS)

Another rule-of-thumb is that rarer facts, or statements, are typically more informative than frequently occurring ones [Lin03]. Consider the example shown in Figure 5.1. The example shows two relationships *lives_in* and *council_member_of* defined on the classes *Person* and *City*. The instances $p_1, p_{2...}p_m$ of the class *Person* are members of the council of *City* c_1, hence the

[8] As previously stated, we consider edges in an RDF graph to be undirected. Hence, the degree of a node *u*, is the sum of the in- and out-degrees of *u*, i.e. *degree(u) = in-degree(u) + out-degree(u)*.

relationship *council_member_of* between each $p_1,p_{2...}p_m$ to c_1. Instances of class *Person* p_{m+1}, $p_{m+2},\ldots p_{k-2}$, p_{k-1}, p_k represent people who live in *City* c_1 and therefore are related to c_1 by the relationship *lives_in.* From the perspective of the node c_1, following an edge labeled *lives_in* will lead to one node among *k-m* possible nodes. In contrast, following an edge labeled *council_member_of* will lead to one node among *m* nodes. Given that rarer paths are considered more informative, the amount of information gained by choosing to traverse the *council_member_of* relationship to a node in the set $\{p_1,p_{2...}p_m\}$ is more than the gain achieved by choosing to traverse the *lives_in* relationship to a node in the set $\{p_{m+1}, p_{m+2},\ldots p_{k-2}, p_{k-1}, p_k\}$. This is akin to choosing the hop with maximum information gain [Anayanwu05].

To define this heuristic formally, we define the notion of the *type* of an RDF statement. The *type* of an RDF statement $\langle s,p,o \rangle$ is defined as the triple $\pi=\langle C_i,P_j,C_k \rangle$ where $typeOf(s)= C_i$, and $typeOf(o)= C_k$. Further, $|\pi|$ is thus the number of statements of type π in a given RDF instance base. We therefore define *Instance Participation Selectivity* for each RDF statement as $\sigma_\pi = 1/|\pi|$ giving greater weight to those statement types which are less common or more rare. Going back to Figure 5.1, let $\pi=\langle Person, lives\_in, City \rangle$ and $\pi'=\langle Person, council\_member\_of, City \rangle$. According to this example, $\sigma_\pi=1/(k-m)$ and $\sigma_{\pi'}= 1/m$ and if $k-m>m$ then $\sigma_{\pi'}> \sigma_\pi$.

Attributing further to the possible rarity of a *type* of an RDF statement is the notion of multiple classifications resulting from annotating with multiple schemas. These multi classified entities can allow for the instantiation of implicit types of RDF statements not defined explicitly in the corresponding schemas. For instance, an entity *e* which is classified as an *Actor* in a schema for the entertainment domain (Figure A.6) could also be classified as a *Spokesperson* in a schema for the business domain (Figure A.5). If the RDF data were to contain a statement such that *e acts_in* a movie *m*, instantiating the explicit statement type $\pi=\langle Actor, acts\_in, Movie \rangle$, then the implicit statement type $\pi' = \langle Spokesperson, acts\_in, Movie \rangle$ has been instantiated.

5.3 The Span Heuristic (SPAN)

In [Anyanwu05] the authors define a ranking metric known as *Refraction.* Given a path of the form $v_1, e_1, v_2, e_2\ldots e_{n-2}, v_{n-1}, e_{n-1}, v_n$ from v_1 to v_n, where $v_i \in Resources$ and $e_i \in Properties$ $\forall i$ $1 \le i \le n$, this path is said to *refract* if there exists at least a pair of statements $\langle v_i,e_i,v_{i+1} \rangle$, $\langle v_{i+1},e_{i+1},v_{i+2} \rangle$ such that $\neg\exists(S|e_i,e_{i+1} \in S)$. In other words a path passes through more than one schema. Figure 5.2 illustrates an ontology in which the instances may be annotated by multiple schemas as well as a semantic association which passes through multiple schemas.

Refraction measures the extent to which a given path conforms to a schema. As mentioned earlier, one of the characteristics of a discovery process is the detection of anomalous information. We consider resources that are instances of classes belonging to different schemas as being indicative of anomalous paths between the given entities, since they tie different domains together. This is especially important when analyzing semantic data that has been annotated by multiple disparate schemas, and it is possible only by allowing entities to belong to several classes. What makes such paths anomalous, and therefore interesting, is the fact that these paths represent a deviation from the expected paths suggested by the individual schemas. For example, in our scenario in Figure 6.1 an instance of the class *Person* may be classified as both an instance of class *Actor* in the *Entertainment* domain and an instance of class *SpokesPerson* in the *Business* domain. Such an instance serves to link different schemas.

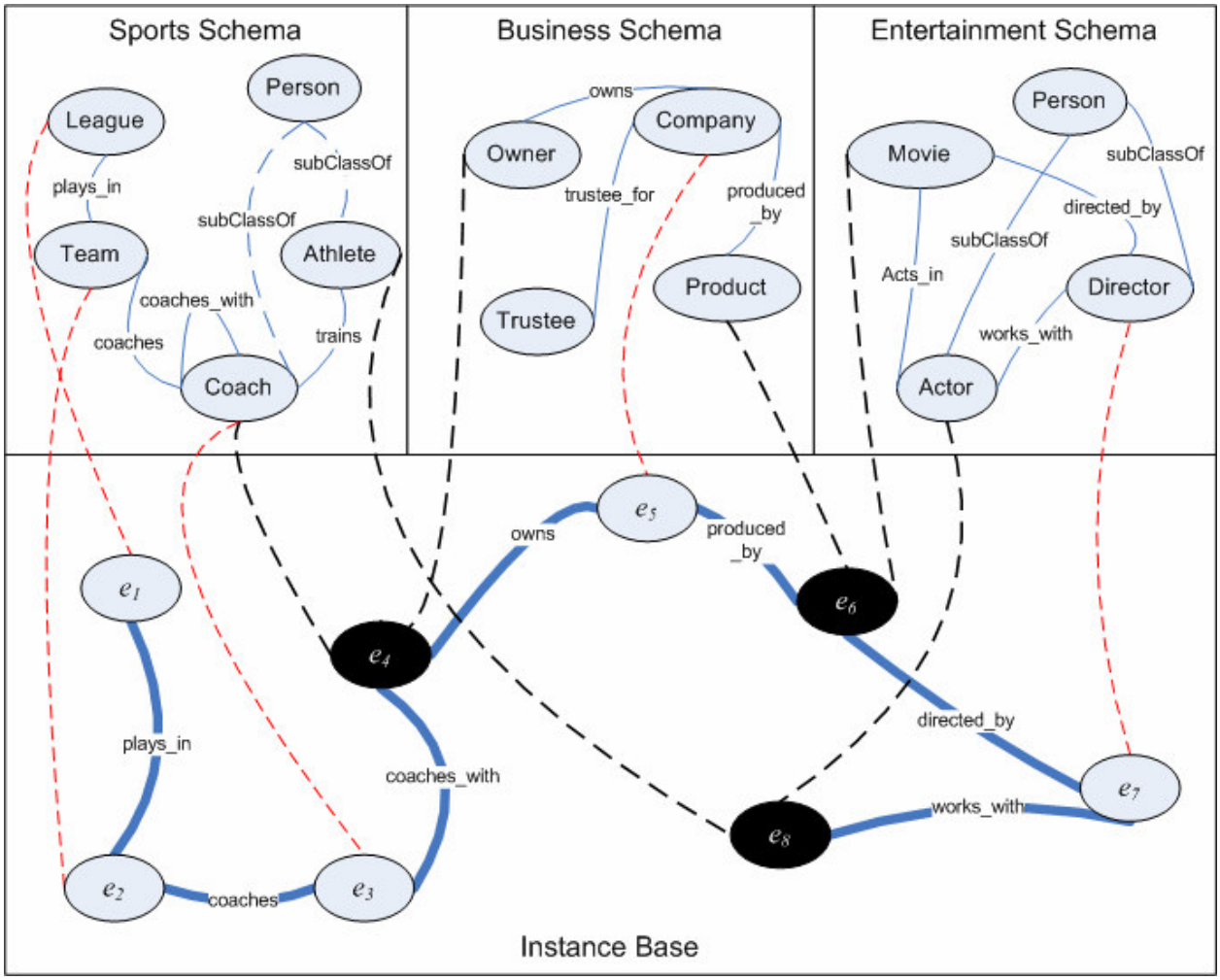


Figure 5.2. Multiple classification of instances.

We therefore need a heuristic that favors the addition of such *refracting* paths to our subgraph. Let us consider the example in Figure 5.3. For every node v in a given RDF graph we can define a set called *SchemaCover* = $\{S | \exists C \in S \wedge typeOf(v) = C\}$. The *SchemaCover* for each of the nodes in the set $\{u', u, v_1, v_2, v_3, v_4, v_5\}$ is shown adjacent to the respective node in Figure 5.3. To favor paths that span as many schemas as possible the search algorithm favors nodes that are classified under as many "new" schemas as possible at each step. By "new" we mean schemas that have been least recently encountered along a particular path. Let *SDiff(u,v)* represent the number of new schemas seen as a result of traversing from u to v, where the value of *SDiff(u,v)* = |*SchemaCover(v)-SchemaCover(u)*|. The idea behind *SDiff* is to ensure that the candidate discovery algorithm chooses a node that is in a "new" schema. However *SDiff* alone does not ensure that the search will continue through the "new" schema. To combat this problem we define the *Cumulative Schema Difference CSDiff(u',u,v) = 1+SDiff(u,v)+SDiff(u',v) for* $v \in adj[u] - \{u'\}$ measuring the number of different schemas that are involved in a pair of edges *(u',u)* and *(u,v)*, where *u'* is the predecessor node to u (the node adjacent to u on the shortest path to its respective root—as described in section 4.1). We then normalize this *Cumulative Schema*

Difference (*CSDiff*) measure to compute a factor $\beta_{u'\to u\to v}$ (for a two-hop path *(u',u,v)*) between 0 and 1 by using the maximum possible value of *CSDiff* (as given in the denominator of equation 12):

$$\beta_{u'\to u\to v_i} = \frac{CSDiff}{1+2(m-1)}, \text{ where } m \text{ is the number of schemas} \quad \textbf{(12)}$$

Adding 1 to the two *SDiff* terms prevents from obtaining a β value of 0 in the case that nodes *u'*, *u* and *v* are all classified in the same schema(s) as one another. We must then add 1 to the normalization factor as well to ensure that all β values are between 0 and 1. This is justified since the additions are performed in all cases, and are thus a constant increase to *CSDiff* and the corresponding normalization factor.

We then obtain the adjusted weight given by:

$$w'(u,v) = \beta_{u'\to u\to v} * w(u,v_i) \quad \textbf{(13)}$$

The effect of the factor β is to bias edge weights in the following way. Successor nodes that are instances of classes belonging to schemas other than those of the current and previous node are more likely to be visited, quantified by the two *SDiff* terms of *CSDiff*. More specifically, in the case of the example in Figure 5.4, a partial ordering is imposed by the adjusted weights *w'(u, v_i),* on the nodes as follows $v_1 \succ v_4 \succ v_3 \succ v_2 \succ v_5$. The node v_1 is therefore visited next. However, the measure β is not sufficient to distinguish between nodes in all cases. Consider the example in Figure 5.4. Nodes v_1 and v_2 have the same value of $\beta_{u'\to u\to v_i}$; but v_1 should be more desirable than v_2 because it has a larger *SchemaCover* value. For such cases, we define a factor called *SchemaCoverFactor* $\alpha(u, v)$:

$$\alpha(u,v) = \frac{1}{2}\left(\frac{|SchemaCover(u)| + |SchemaCover(v)|}{m}\right), \text{ where } m \text{ is the number of schemas} \quad \textbf{(14)}$$

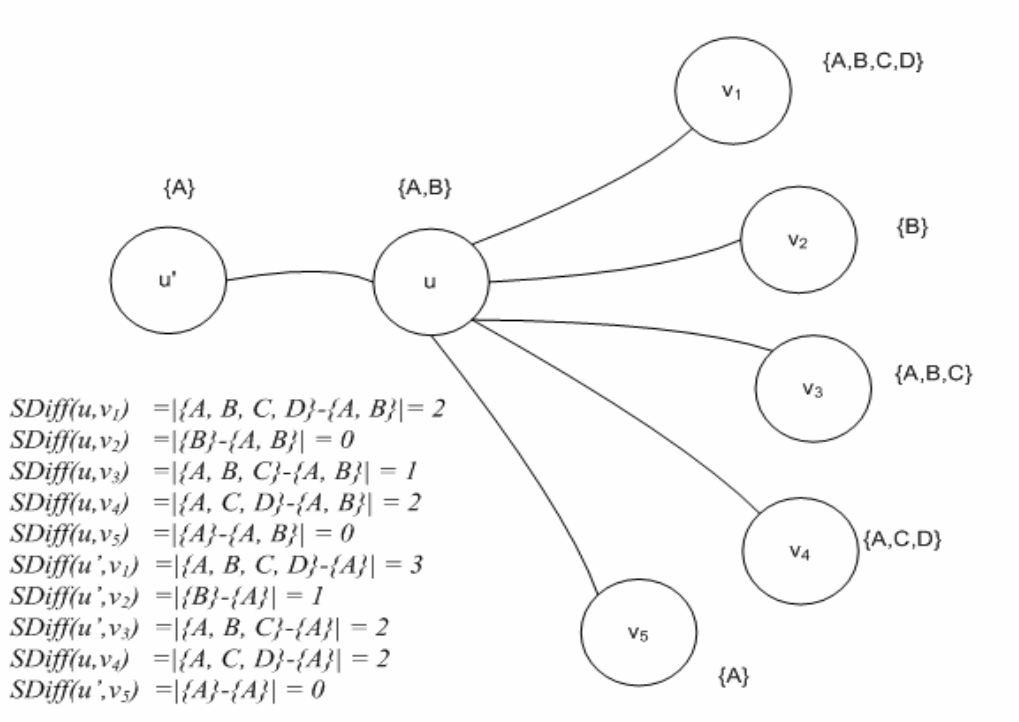


Figure 5.3. Span metric computation

As per the calculations shown in Figure 5.4, this factor treats the node v_1 preferentially over node v_2 i.e. $v_1 \succ v_2$ thus resolving the ambiguity. The value of $\beta_{u' \to u \to v}$ is computed at each step during the discovery process in contrast to the *a priori* values of edge weights computed using μ, σ and α shown in equation 16. $\beta_{u' \to u \to v}$ is then used to adjust the weights $w(u, v_i)$ $\forall i\, 1 \le i \le n$. Thus, if SPAN is used, the distance measure used in the *candidate ρ-graph* generation algorithm is augmented:

$$distance\ (u,v) = log\left(\frac{(degree\ (u) + degree\ (v))^2}{w(u,v) * \beta_{u' \to u \to v}}\right) \quad \textbf{(15)}$$

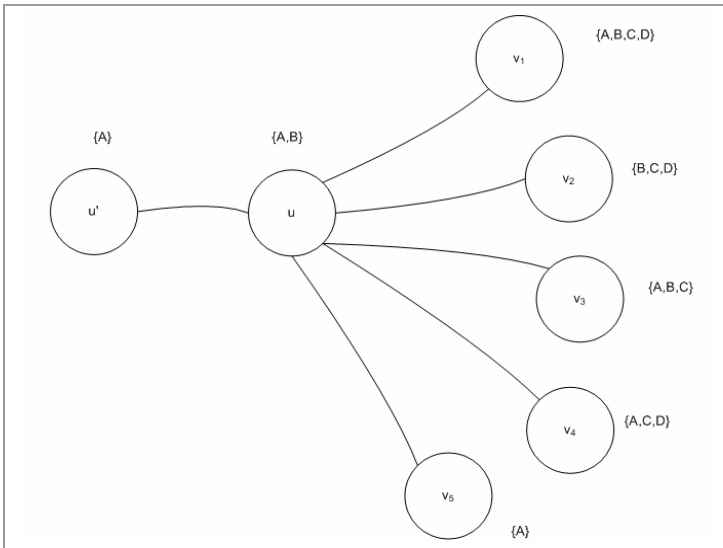
Finally, using these heuristics, the weight of an edge is calculated by the following equation

$$w(u,v) = \frac{\mu(p_{u \to v}) + \frac{1}{2}\left(\frac{\mu(u)}{degree(u)} + \frac{\mu(v)}{degree(v)}\right) + \sigma_{\pi} + \alpha(u,v)}{4} \quad \textbf{(16)}$$

where $p_{u \to v}$ is the property connecting the resource node u and v, and π is the type of the statement $\langle u, p_{u \to v}, v \rangle$.

As discussed in the beginning of chapter 4, we consider the edges in an RDF graph to be undirected. However, there is a caveat to this when using the span heuristic. When the heuristic is used, the distance on an edge *(u,v)* is dependent on the node from whose expansion *u* was added to the results, i.e. *u'*, and its classifications. Thus, while *d(u,v)* is dependent on the

classifications of *u'*, *d(v,u)*, is dependent on the classifications of *v'* such that node *v* was added to the results by expanding *v'*. In this case, *d(u,v)* may not be equal to *d(v,u)*.



SDiff(u,v1) =|{A, B, C, D}-{A, B}|= 2
SDiff(u',v1) =|{A, B, C, D}-{A}| = 3
CSDiff = 1+SDiff(u, v1) +SDiff(u', v1)

$$\beta_{u'\to u\to v_1} = \frac{\text{CSDiff}}{1+2(m-1)}$$

$$\beta_{u'\to u\to v_1} = \frac{1+3+2}{1+2(4-1)} = 0.8571$$

SDiff(u,v2) =|{B,C,D}-{A, B}| = 2
SDiff(u',v2) =|{B,C,D}-{A}| = 3
CSDiff = 1+SDiff(u, v2) +SDiff(u', v2)

$$\beta_{u'\to u\to v_2} = \frac{\text{CSDiff}}{1+2(m-1)}$$

$$\beta_{u'\to u\to v_1} = \frac{1+3+2}{1+2(4-1)} = 0.8571$$

Since

$$\alpha(u,v_1) = \frac{1}{2}\left(\frac{|SchemaCover(u)|+|SchemaCover(v_1)|}{m}\right) = 0.75$$

$$\alpha(u,v_2) = \frac{1}{2}\left(\frac{|SchemaCover(u)|+|SchemaCover(v_2)|}{m}\right) = 0.625$$

And,

$$w'(u,v_i) = \beta_{u'\to u\to v_i} * w(u,v_i)$$

Because $w(u,v_i)$ is calculated using $\alpha(u, v_i)$

$w(u,v_1) \neq w(u,v_2)$

$$\therefore w'(u,v_1) \neq w'(u,v_2)$$

even if $\beta_{u'\to u\to v_1} = \beta_{u'\to u\to v_2}$

Figure 5.4. Influence of the Schema Cover Factor α. Without using the Schema Cover Factor, the weights (in respect to the SPAN heuristic) on the edges (u, v_1) and (u, v_2) are equal, even though the scheme coverage of the two edges are different.

5.4 Semantics in Heuristics

These heuristics are clearly based on the structure of the semantics expressed in the schema(s) by which the data is annotated:

- Without the hierarchical relationships (*rdfs:subPropertyOf* and *rdfs:subClassOf*) expressed in the schema(s), the class specificity (CS) and property specificity (PS)

heuristics would not be possible as all classes and properties would share the same level of specificity.

- Without edge labels and entity classifications, types of statements, such as *Person council_member_of City*, as used in the instance participation selectivity (IPS) heuristic would not be possible as all edges would indicate the same meaning (as is the case in the data set used in [Faloutsos04]). Furthermore, these labels allow the representation of multiple relationships between entities, rather than single meaningless connections.
- Without the notion of multiple classifications for entities, as well as the ability to annotate data with multiple schemas, the span (SPAN) heuristic would not be possible: a semantic association can only span multiple schemas if it including entities which are multiply classified across multiple schemas.

An algorithm incorporating these heuristics can allow an analyst to specify what aspects of the semantics are the most significant. To this end, any subset of the heuristics can be used in computing a relevant subgraph between a pair of nodes. Based on these four heuristics, there are 2^4 (16) combinations of the heuristics, and we use each of these combinations in our empirical evaluation (see chapter 7) of applying our heuristics to the algorithms described in [Faloutsos04]. In each subset of the heuristics, we consider each heuristic used to be of equal significance. Thus, if only two heuristics are use, the final weight of an edge is the sum of the weights given by the respective heuristics; if all four are used, the final weight is the sum of the weight given by all four heuristics[9].

[9] In our experiments, we compare all 16 combinations, including using no heuristics. In this case, all edges are given the same weight; thus, the semantics represented in the data are ignored.

CHAPTER 6

DATASET AND SCENARIO

The types of datasets used in semantic analytic applications typically contain sensitive information and are therefore not publicly available. Thus, in order to apply the aforementioned algorithms and heuristics, we needed to simulate the population of the ontology schemas used in these applications. To this end, we used a synthetic dataset for our experiments which also provided us control over characteristics of the data. This helps us ensure that our results are not unduly affected by unknown aspects, i.e., connectivity, relative instance distribution, etc. of the dataset. Collection of real world data follows an almost opportunistic approach since availability often dictates design. As a result there is room for skew in instance data population. This skew may not always reflect real-world distributions, as was observed in our experience with SWETO [Aleman-Meza05]. Consequently, we used an open source tool, TOntoGen [Perry05], developed in the SemDis project, that takes as input a set of ontology schemas[10] and a properties file specifying relative distributions of instances of classes and properties that would be expected in the real world. For example, consider two classes in the *Business* ontology (Appendix Figure A.2.): *Trustee* and *Employee.* It would be reasonable to assume that if there are 5000 instances of the class *Employee* then there are ***unlikely*** to be 1000 instances of the class Trustee. Instances of the class *Trustee* are more likely to number less than 100. These numbers are domain specific. Our method for assigning values to these relative distributions is empirical and a discussion of this issue is beyond the scope of this thesis. The result of running this algorithm is an RDF graph that contains nodes and edges that are instances of classes and property types belonging to any or all of the classes in the given schemas. The graph for our experiments contains 30,000 nodes and 45,000 edges.

While the results and evaluation herein are based on this synthetic dataset, we are currently expanding SWETO to have a richer schema with more interesting relationships as well as developing an ontology based on medical documents. Such datasets will potentially allow user evaluation through the opinions of domain experts. For instance, a medical researcher could perform a query and verify the relevance of the results.

As a motivation for the domains used in our dataset consider the following example. A fraud investigator with the Securities and Exchange Commission (SEC) receives the following piece of information about a week after the stock prices for *EntertainmentCompany_9982* plummet. *Actor_5567* sold 70% of his shares of *EntertainmentCompany_9982* one week after *Capt_8262* sold all of his shares in the same company. Both transactions took place two weeks before the prices plummeted. The example subgraph shown in Figure 6.1 might help an investigator visualize the connections between the resources *Actor_5567* and *Captain_8262*.

[10] See section A.2 of the appendix for illustrations of the 3 schemas used in our synthetic dataset.

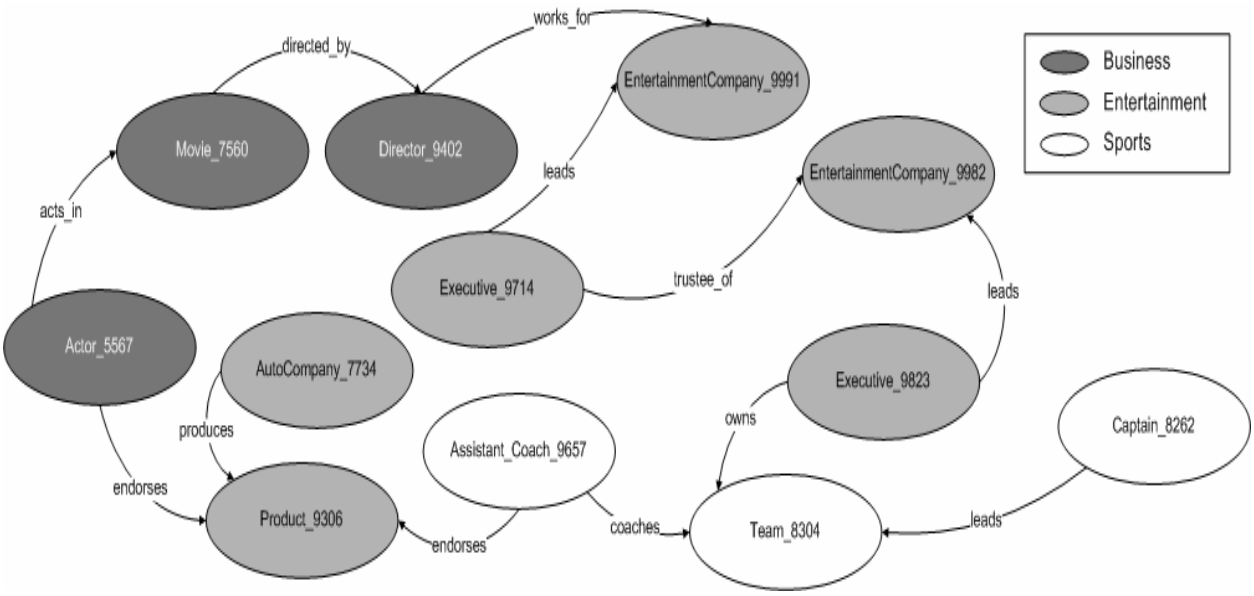


Figure 6.1. **Sample Query Result** Example snippet of a subgraph returned for the query ρ(Actor_5567, Captain_8262) on our synthetic dataset– Nodes in the above graph are color-coded according to the ontology to which their class belongs

CHAPTER 7

RESULTS AND EVALUATION

We recognize the fact that the notion "best" subgraph is very subjective and dependent on the user's perspective. It is however desirable to have an objective measure that could be used to quantify the quality of a generated subgraph. The issue of judging relevance of paths, i.e. path ranking, has been addressed in [Anyanwu05] and [Aleman-Meza05]. In [Lin03] the authors use *rarity* of the path as a measure of its interestingness. Currently, these are the only three efforts that measure path relevance in relational data. We therefore use these path ranking mechanisms to evaluate the quality of both the *Candidate ρ-graph* and the *Display ρ-graph*. In our experiments the *Candidate ρ-graphs* generated contained 3000[11] nodes and the *Display ρ-graphs* were restricted to a maximum of 100 nodes making them easy to visualize. The results presented herein are values averaged over 30 queries.

7.1 Evaluation using Path Ranks

In our data set there are over 60 million paths of length 13 between the two endpoints used in Fig. 4. Paths of this length are unlikely to be of much interest to the user. To evaluate our subgraphs, we run an exhaustive *k-hop* limited Depth-First Search (DFS) on the input graph between the two entities. We use a depth limit of 9 hops for our experiments for feasibility of path enumeration for ranking. Note that both the *Candidate ρ-graph* and *Display ρ-graph* generated do contain arbitrary length paths, but we only consider paths of length at most 9 for fairness of comparison[12]. We represent the paths returned by the *k-hop* DFS as the set $FGPaths_9$ *(paths of up to 9 hops in the full graph).* There are therefore 30 distinct $FGPaths_9$ sets, one for each query in our experiments. We rank the paths in each of the $FGPaths_9$ sets using the ranking mechanisms proposed in [Anyanwu05] and [Aleman-Meza05] in addition to what we call *Rarity Rank* based on the method suggested in [Lin03]. The rank of a path *p* based on the *Rarity Rank* scheme is given by the inverse of the number of paths that share the same type as path *p*. Each of the ranking mechanisms applied to the set $FGPaths_9$ results in a list of ranked paths. Let us assume that this leads to ranking from 1→M where M is the rank of the least relevant path. Let this set of ranked paths be represented as $FGRankedPaths_9$. We therefore have three distinct scales ($FGRankedPaths_9$ sets) against which the quality of both *Candidate ρ-graph* and the *Display ρ-graph* can be measured. In all of the graphs shown below the x-axis represents the 16 possible combinations of the 4 heuristics we use *viz. class* and *property specificity (CS and PS), Instance Participation Selectivity (IPS)* and *The Span Heuristic (SPAN).*

[11] This was the observed number of nodes in the *Candidate ρ-graph* for all the 30 queries used in our experiments. Further investigation revealed that this was an artifact of the connectivity of our dataset.

[12] See the appendix for a discussion on other k-limits for evaluating *Candidate Graph Quality*.

7.2 Measuring Candidate ρ-graph quality

To measure *Candidate ρ-graph* we compare the best paths in the entire graph to those in the *Candidate ρ-graph.* Let $CGPaths_9$ represent the set of paths in the *Candidate ρ-graph* with maximum length 9. For each path $p_{candidate} \in CGPaths_9$ we count the number of paths $p \in FGRankedPaths_9$ such that $rank(p) > rank(p_{candidate})$. This gives us the rank of each path in the *Candidate ρ-graph* with respect to all paths in the set $FGRankedPaths_9$. The score of a path is calculated by the cardinality of (9 hop) paths in the full graph minus rank of the path, formally given by:

$$score(p_{candidate}) = \left|FGRankedPaths_9\right| - rank(p_{candidate}) \quad \textbf{(17)}$$

The quality of the *Candidate ρ-graph* is therefore calculated as the sum of the scores in the *Candidate ρ-graph* divided by the sum of the *k* top ranked paths in $FGRankedPaths_9$, where the *Candidate ρ-graph* contains *k* paths, formally given by:

$$Q(CGPaths_9) = \frac{\sum_{p_{candidate} \in CGPaths_9} \left(score(p_{candidate})\right)}{\sum_{r=1}^{|CGPaths_9|} \left(\left|FGRankedPaths_9\right| - r\right)} \quad \textbf{(18)}$$

Figure 7.1 shows that the Candidate ρ-graph containing *k* paths obtained using our edge weighting schemes achieves between 80—90% of the score that can be achieved by choosing the *top-k* ranked paths from the full graph (entire dataset of 30000 nodes and 45,000 edges). The Candidate ρ-graphs in our results typically contain 30-40% of the paths in the entire graph between the endpoints yet are 80-90% as *"good"* as the top paths in the entire graph between the two endpoints.

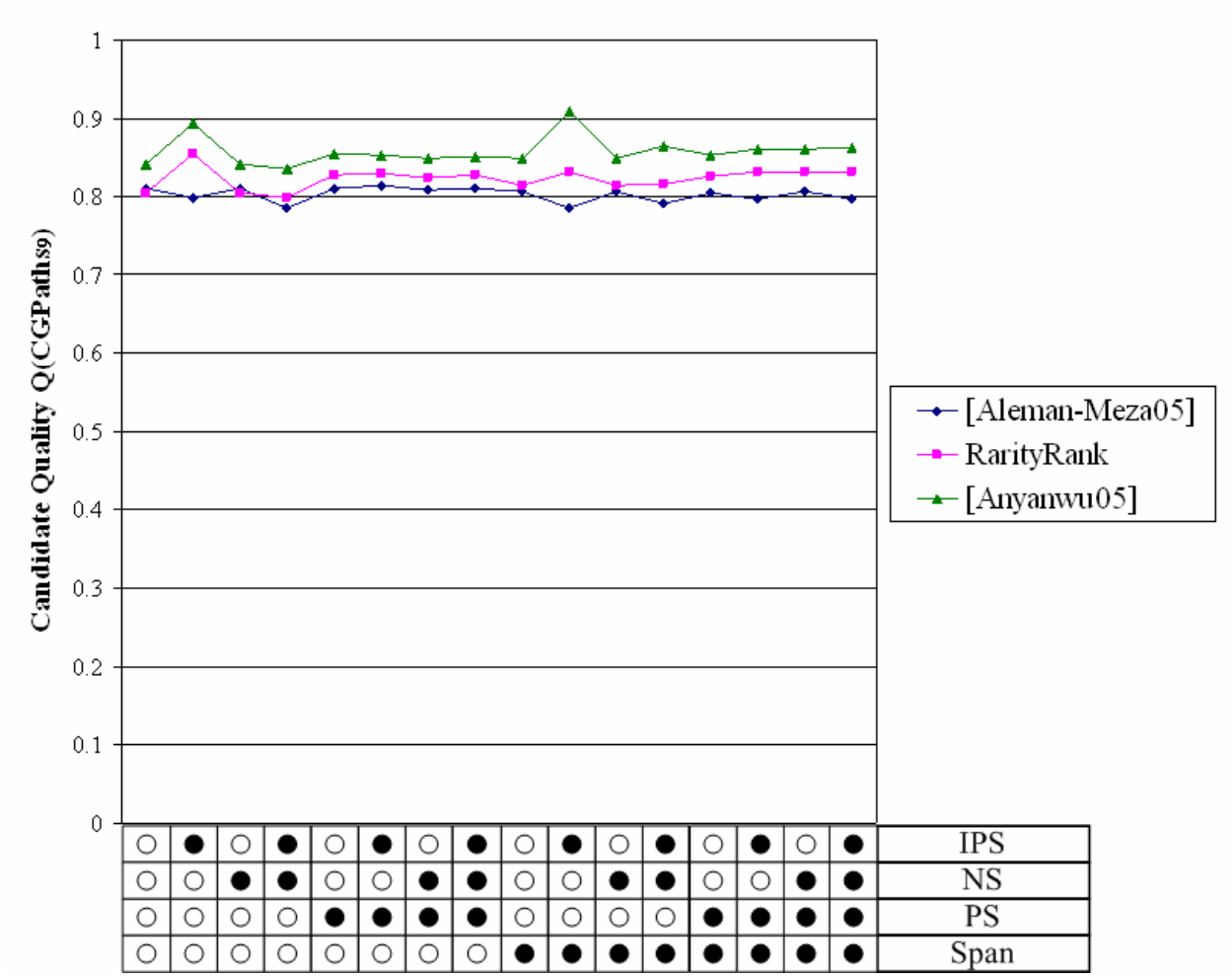


Figure 7.1. Quality of the Candidate ρ-graph

The use of heuristics to prune a search spaces poses a question as to what is being lost by using the algorithm. In order to gain insight into information loss, we calculated what percentage of the total number of paths was found in the *Candidate ρ-graph* (Figure 7.2). We then compared these percentages to the percentage of the total score found in the *Candidate ρ-graph* (Figure 7.3). To determine the percentage of the total score in a *Candidate ρ-graph*, we calculated the score obtained by summing the score of all paths for a query. What is most interesting about Figures 7.2 and 7.3 is that they show the same trend as one another. Furthermore, the percentage of total paths found is very similar to the percentage of the total score obtained in the *Candidate ρ-graph*.

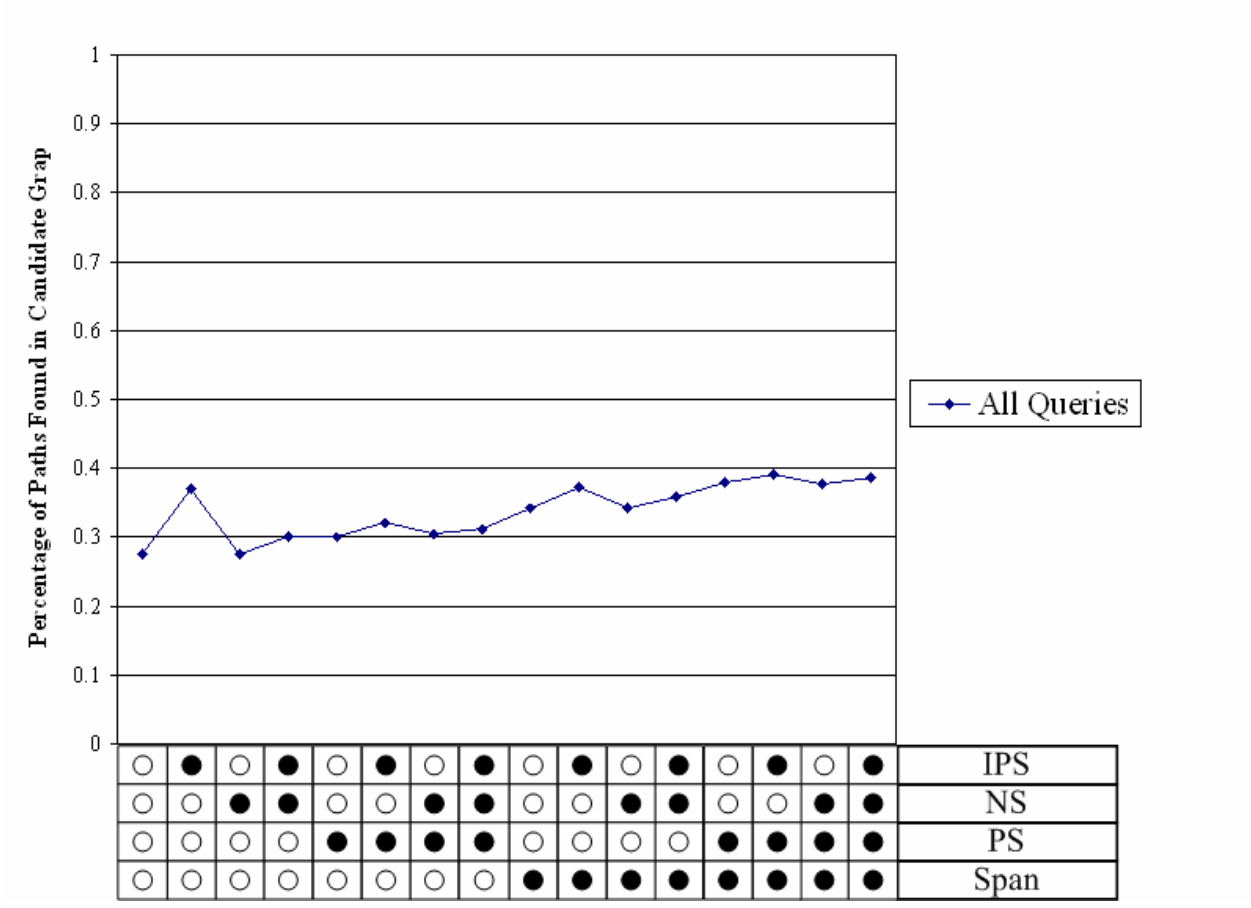


Figure 7.2. Percentage of All Paths Found in Candidate ρ-graph

In regards to timing, it takes approximately a few hundred milliseconds to compute Candidate ρ-graphs. This satisfies our requirement of an interactive subgraph generator, as described section 4.3, but further timing comparisons with other algorithms would be necessary.

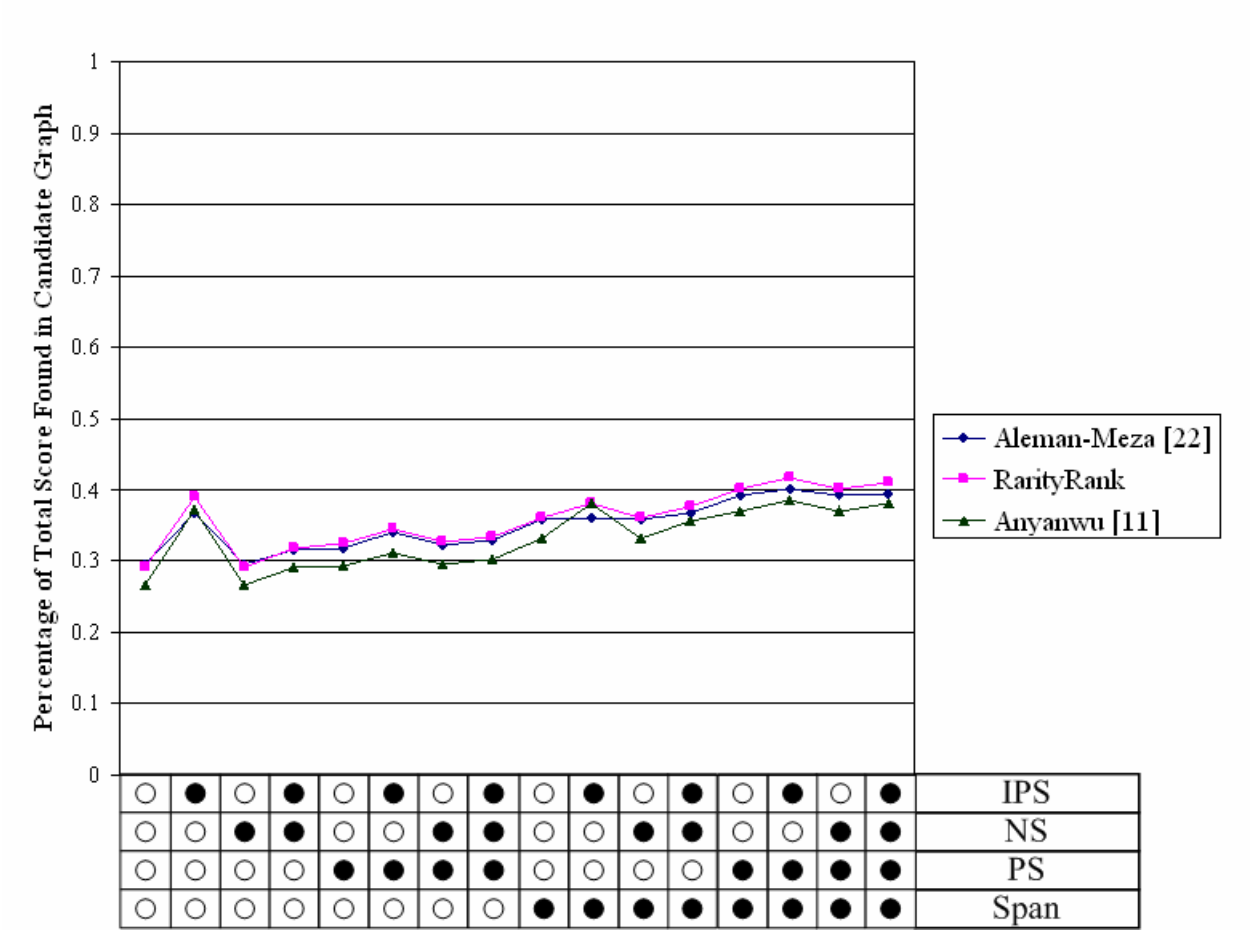


Figure 7.3. Percentage of Total Score Found in Candidate ρ-graph

7.3 Measuring Display ρ-graph quality

Similar to *Candidate ρ-graph* quality, we compare the paths in the *Display ρ-graph* to the best paths in the entire graph. Let the set *DGPaths* represent the paths in the *Display ρ-graph*. The rank of a path in the *Display ρ-graph* is computed exactly the same way the rank of a path in the *Candidate ρ-graph* is computed, as is the score.

$$score(p_{display}) = \left|FGRankedPaths_9\right| - rank(p_{display}) \quad \textbf{(19)}$$

The quality of a display graph is computed by comparing its cumulative score to the best possible display that could be obtained from the ranked set of paths in the full graph. We refer to this best possible display as *Pseudo-Display ρ-graph*. In our experiments we use a *budget* of 100 nodes for our *Display ρ-graphs*. Starting with an empty *Pseudo-Display ρ-graph* and the path with rank 1 in the set $FGRankedPaths_9$ we add paths to the *Pseudo-Display ρ-graph* until size (number of nodes) of the *Pseudo-Display ρ-graph* matches the size of the corresponding *Display ρ-graph*. The cumulative score of the *Pseudo- Display ρ-graph y* is then computed as the sum of the scores of the paths. The quality of a *Display ρ-graph* is therefore given by:

$$Q(DGPaths) = \frac{\sum_{p_{display} \in DGPaths} score(p_{display})}{\sum_{p_{pseudo} \in Pseudo-Display} score(p_{pseudo})} \quad \textbf{(20)}$$

Figure 7.4 shows that starting with a *Candidate ρ-graph*, giving 80—90% quality, the corresponding computed *Display ρ-graph* captures a maximum of 84% of the score that can be obtained by computing a *Pseudo-Display ρ-graph* from the best *k* paths in the full graph, where the *Display ρ-graph* contains *k paths*. Our results show the quality of Display ρ-graphs with respect to SemRank [Anyanwu05] to be surprisingly low – 43%. Further investigation of the methods used revealed that the difference between the ranking scheme in [Aleman-Meza05] and that in [Anyanwu05] is that in the former instance node degrees affect the rank of a path (nodes of lower degree being favored) whereas in the latter rank of path is determined purely by properties in the path. Our heuristics favor lower degree nodes and hence the observed trend. A personal communication with the authors of [Anyanwu05] revealed that extending SemRank to include the effect of nodes is an intended follow up to this work.

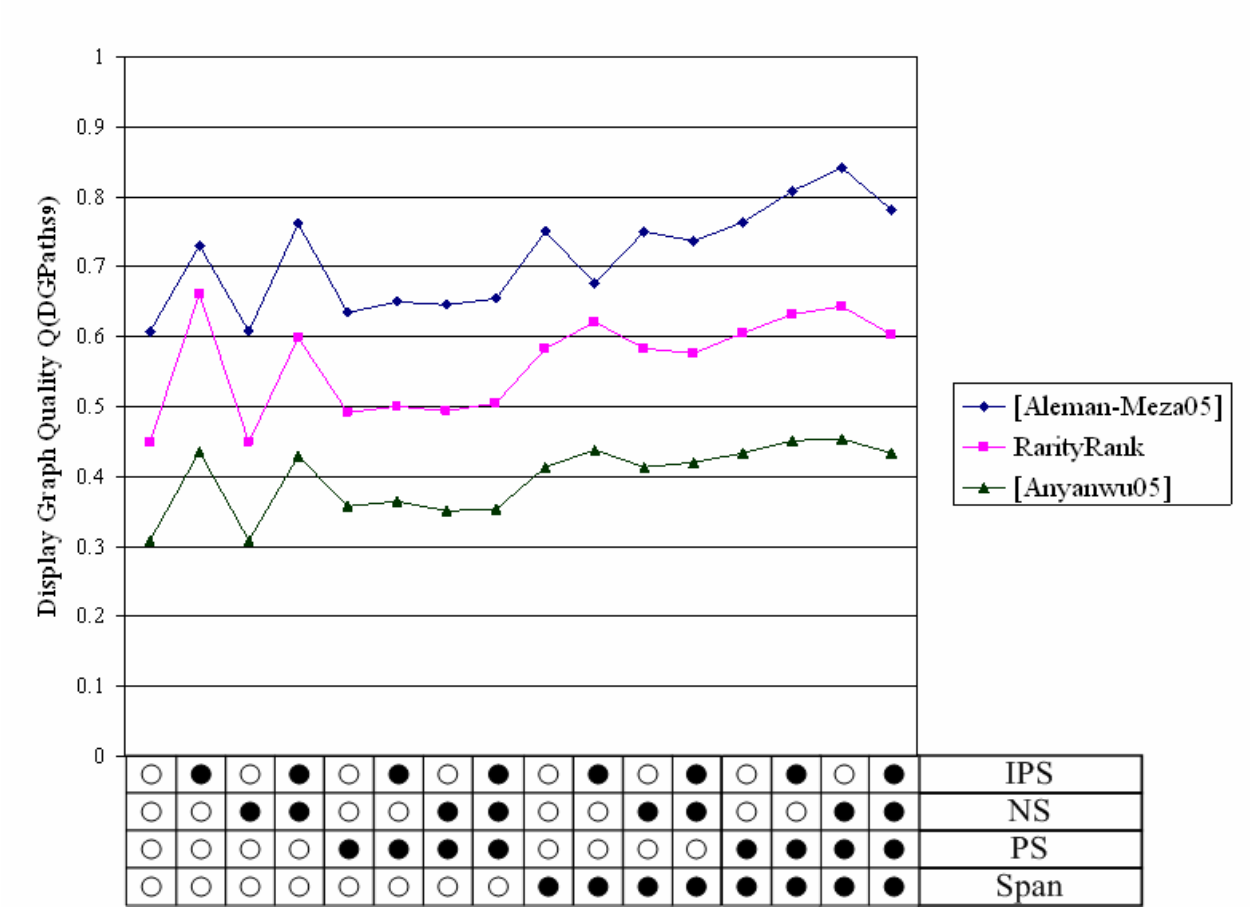


Figure 7.4. Quality of the Display ρ-graph – Note that all weighting heuristics turned off results in poor graph quality in contrast with all heuristics turned on

As another method of comparing a *Display ρ-graph* to a *Candidate ρ-graph,* we created a *Pseudo-(k)-Display ρ-graph* by starting with an empty graph, and then adding successive top ranked paths until the number of paths in the *Pseudo-(k)-Display ρ-graph* matched the number of paths in the *Display ρ-graph*. Figure 7.5 shows that the *Display ρ-graphs* computed from the *Candidate ρ-graphs* captures a maximum of 85% of the score that can be obtained by creating a graph of the best *k* paths—the *Pseudo-(k)-Display ρ-graph*.

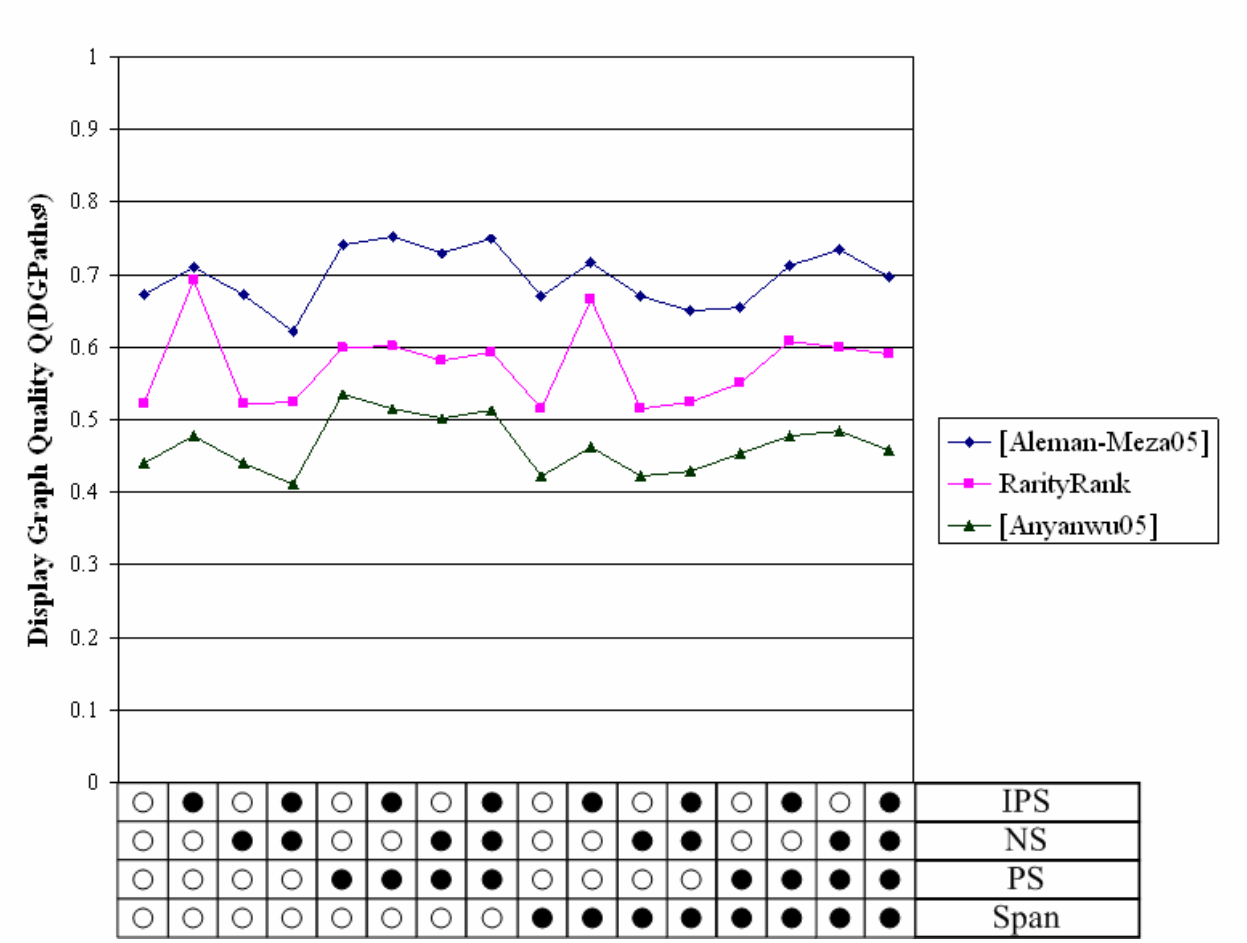


Figure 7.5. Quality of Display ρ-graph with respect to corresponding Candidate ρ-graph

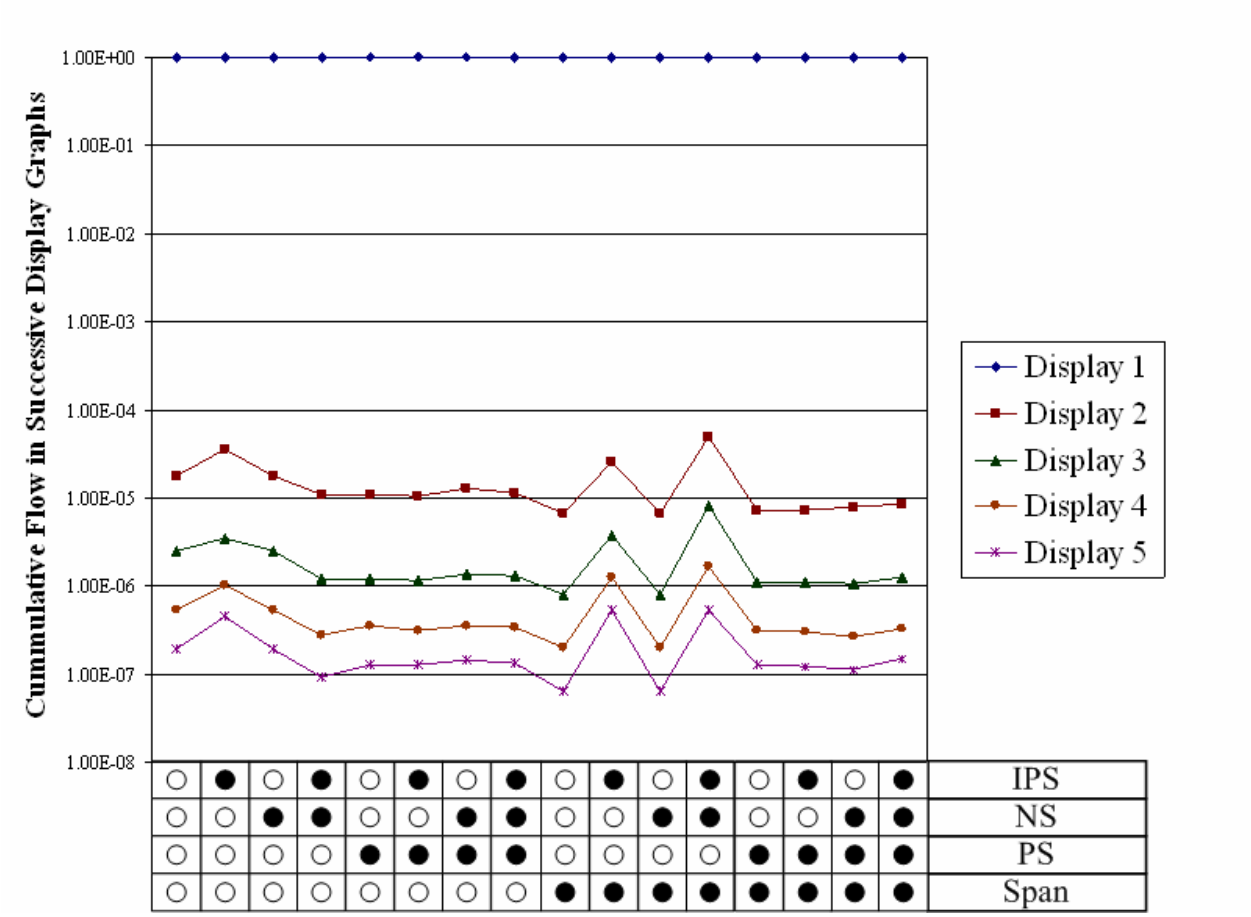


Figure 7.6. Successive Current Flow Current Flow in 5 Successive Display ρ-graphs relative to the best

7.4 Successive Display ρ-graph quality

With the intention of validating the current flow model for subgraph relevance [Faloutsos04] we conducted the following experiment. We computed what we term as *Successive Display ρ-graphs*. To construct these displays we successively run the *Display ρ-graph* generation algorithm on the candidate graph. At each successive run we discount the paths used in previous displays. This results in the next best Display ρ-graph at every successive run. This process is repeated five times in our experiments to obtain five Display ρ-graphs. The current flow in each of these Display ρ-graphs is plotted relative to the current flow in the first Display ρ-graphs on a ***log*** scale in Figure 7.6. The quality of these Display ρ-graphs is plotted relative to the quality of the first Display ρ-graph in Figure 7.7. There is a large difference both in the current flow and the display quality between the first display and the next display. This confirms that there is a correspondence between current flow in the Display ρ-graphs and their quality. This in turn supports the electricity based model for RDF graph relevance. Note that the plots below are averages of the relative differences of successive displays over all ranking schemes.

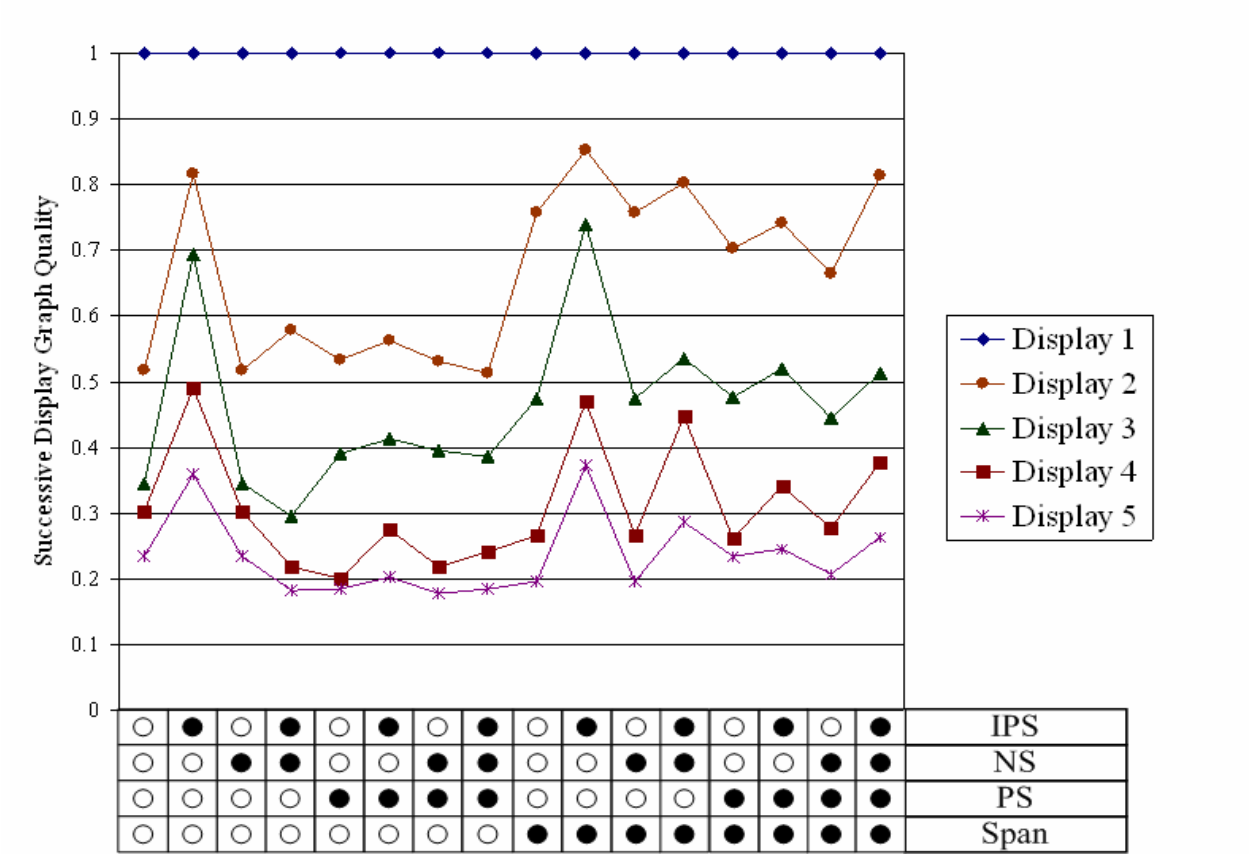


Figure 7.7. Successive Display ρ-graph Quality The quality of 5 Successive Display ρ-graphs relative to the best

7.5 Comparison of Different query types

We differentiate our queries into two types: Inter-domain and Intra-domain. As the names suggest Inter-domain queries are those that seek paths for queries involving entities that are classified as instances of classes belonging to different schemas. Intra-domain queries seek to find paths between entities of classes belonging to the same schema. Of the 30 queries used for our evaluation, 15 are inter-domain queries and 15 are intra-domain queries.

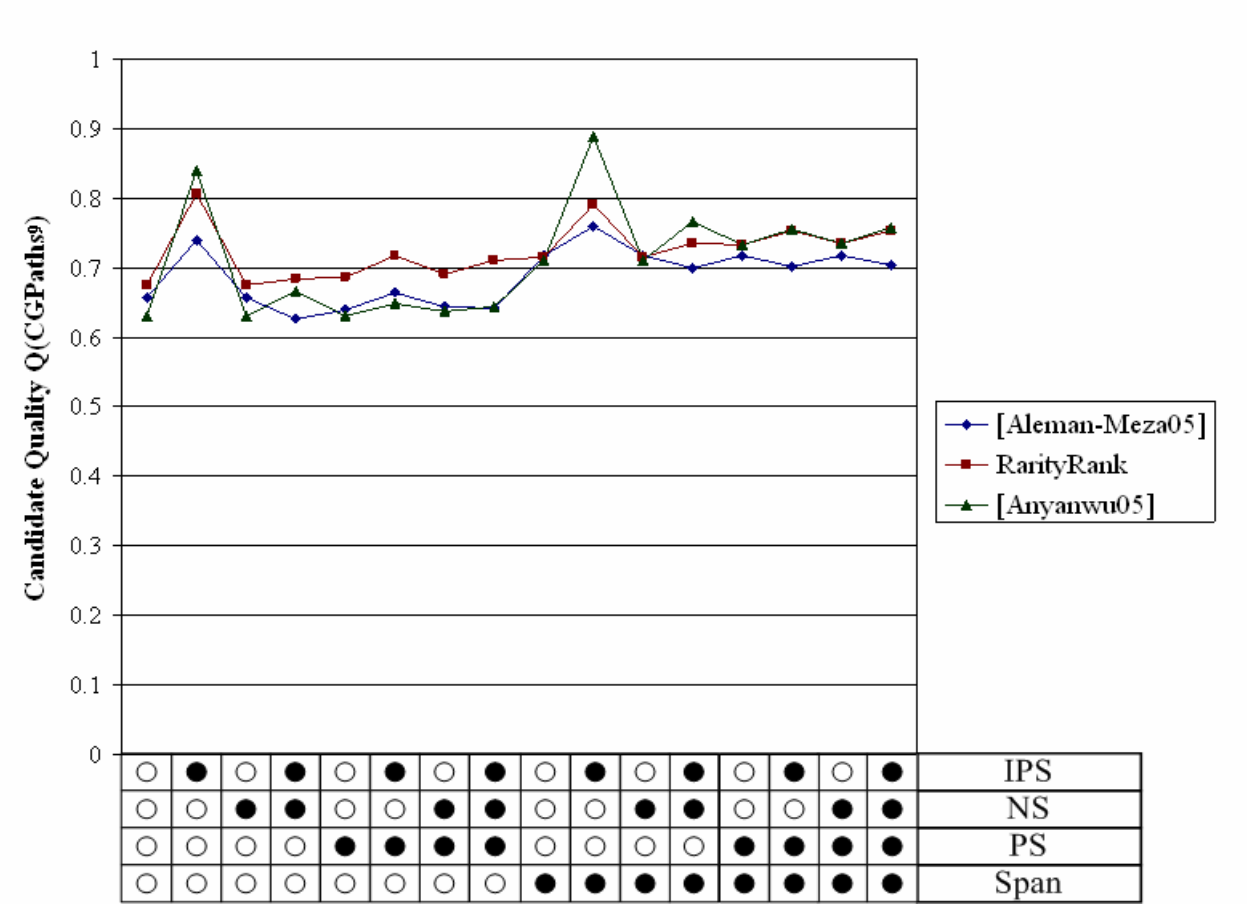


Figure 7.8. Quality of Candidate ρ-graph for Inter-domain Queries.

In Figures 7.8 and 7.9 we present the Candidate ρ-graph quality Q(CG-Paths$_9$) for the Inter-domain and Intra-domain queries, respectively, in an attempt to gain insight into which of our weighting schemes work best for each query type.

Figure 7.8 shows peaks at the 2nd and the 10th combinations of settings of the four heuristics *viz.* Instance Participation Selectivity (IPS the 2nd) and Span + Instance Participation Selectivity (the 10th). IPS favors rarer paths, based on the rarity of each statement in the path, and since paths passing through multi-classified nodes are rarer, the 2nd combination results in better quality *Candidate ρ-graphs* than other settings. Combining SPAN with IPS (10th combination) results in even better Candidate ρ-graphs as paths traversing multiple schemas are more rare than those remaining within a schema. IPS and SPAN therefore are better settings to discover Inter-domain Display ρ-graphs.

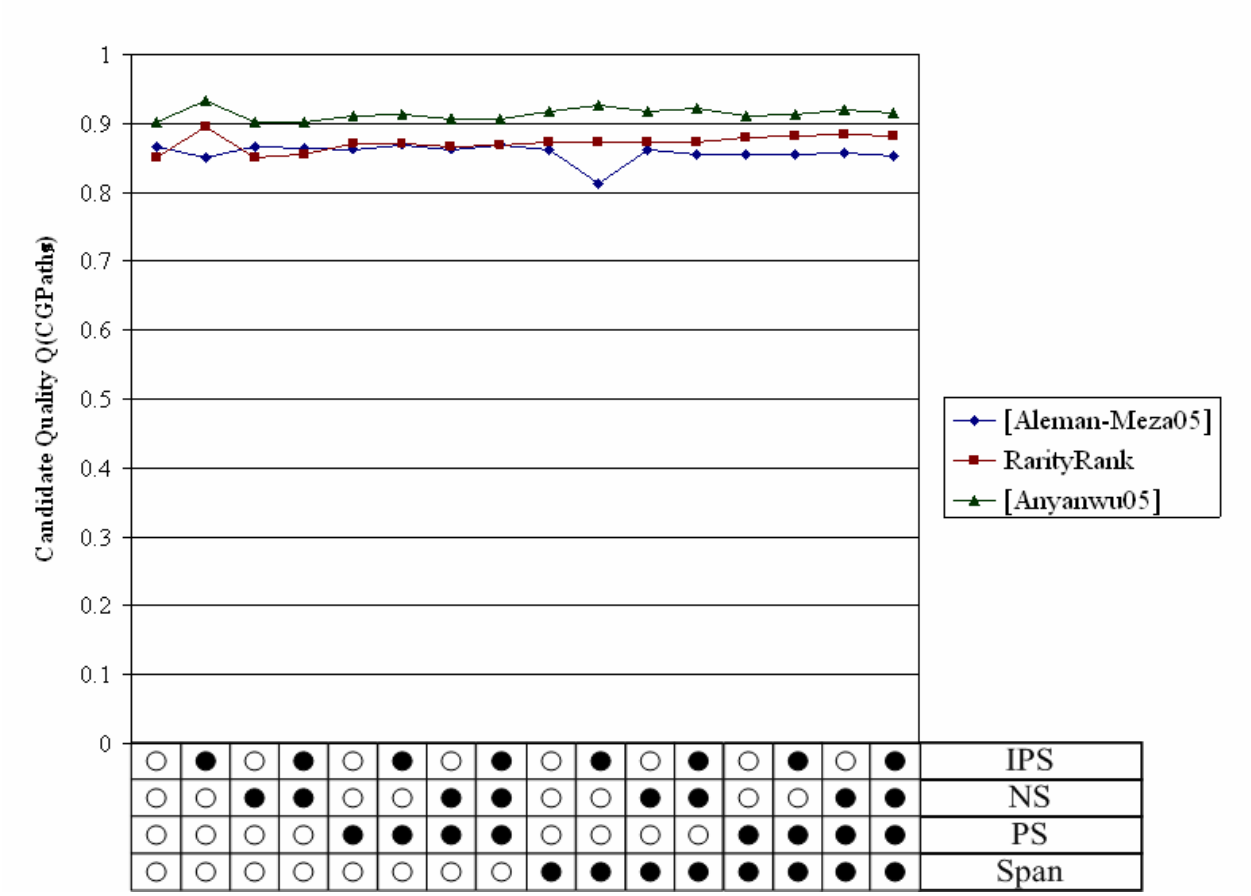


Figure 7.9. Quality of Candidate ρ-graph for Intra-domain Queries.

Figure 7.9 also shows a slight peak at the 2nd combination of settings, but not as defined as in Figure 7.8. Furthermore, the 10th combination of settings in Figure 7.9 actually shows a dip in *Candidate ρ-graph* quality compared to the ranking metrics described in Aleman-Meza [Aleman-Meza05]. This is due again to rarity being favored by SPAN and IPS: most paths between queries in the same domain will not traverse multiple domains, and thus, the use of SPAN adds no benefit for inter-domain queries. In fact, the quality of the *Candidate ρ-graph* for Inter-domain queries portrays little variability among the setting combinations. Intuitively, this occurs because there is not much variability in respect to the affect of the heuristics within a single domain. For instance, there is little variety of statement types within a domain, whereas undefined statement types (undefined in respect to any of the individual corresponding schemas) may occur when considering multi classified entities.

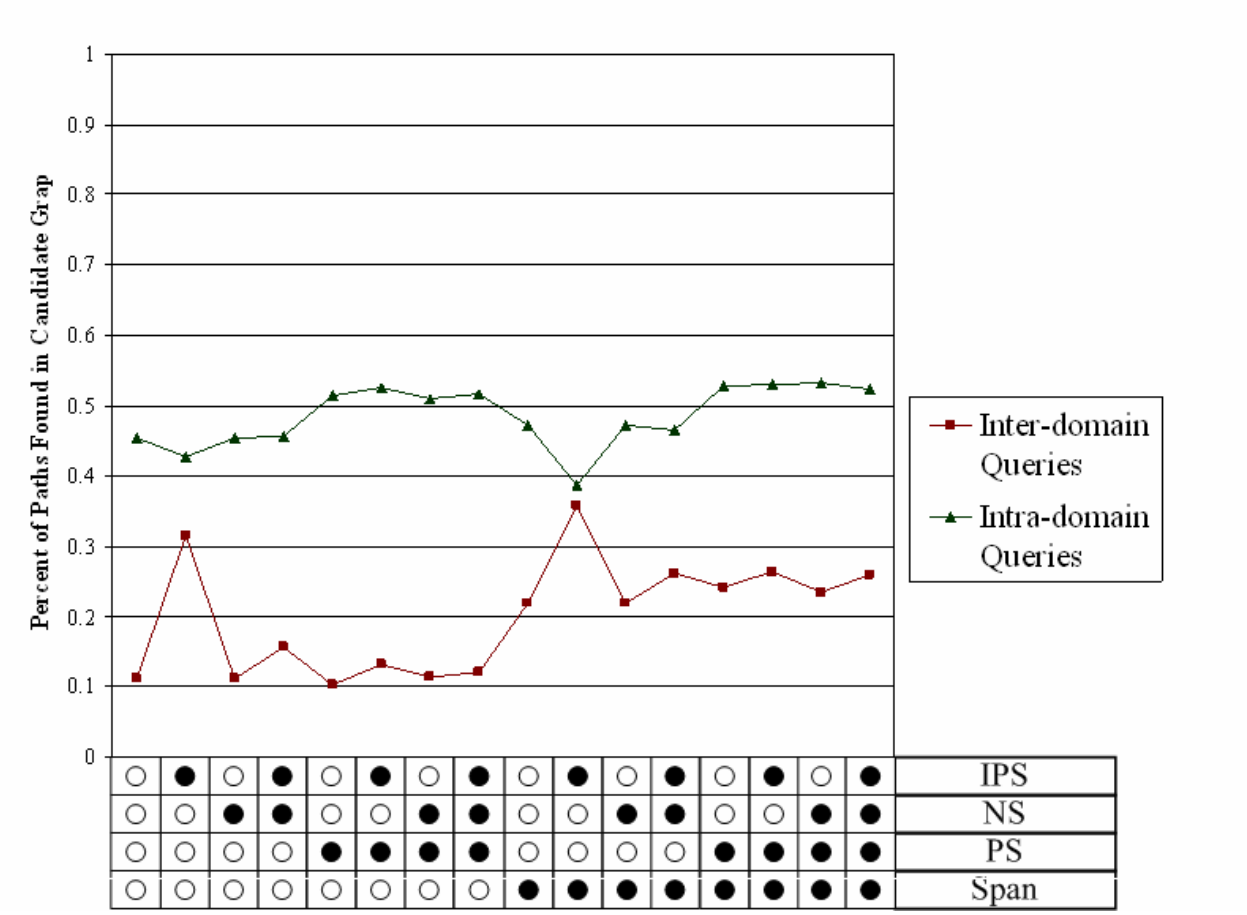


Figure 7.10. Percentage of Total Paths found in Candidate ρ-graph.

The difference between query types is reiterated by Figure 7.10 and Figure 7.11. The former figure illustrates the percentage of paths existing in the full graph which were actually found in the *Candidate ρ-graph*. The top line represents the percentages for Intra-domain queries, while the bottom line represents the same for Inter-domain queries. As shown by the figure, on average a greater percentage of paths exist for Intra-domain queries than for Inter-domain queries. The number of paths for Inter-domain queries is typically greater than that for Intra-domain queries. Averaged over the queries, we find approximately 15,000 to 30,000 paths for Inter-domain queries, while we find between 5,000 and 10,000 for Intra-domain queries. Paths between entities are typically shorter within a domain, and are thus more likely to lie within a *Candidate ρ-graph*.

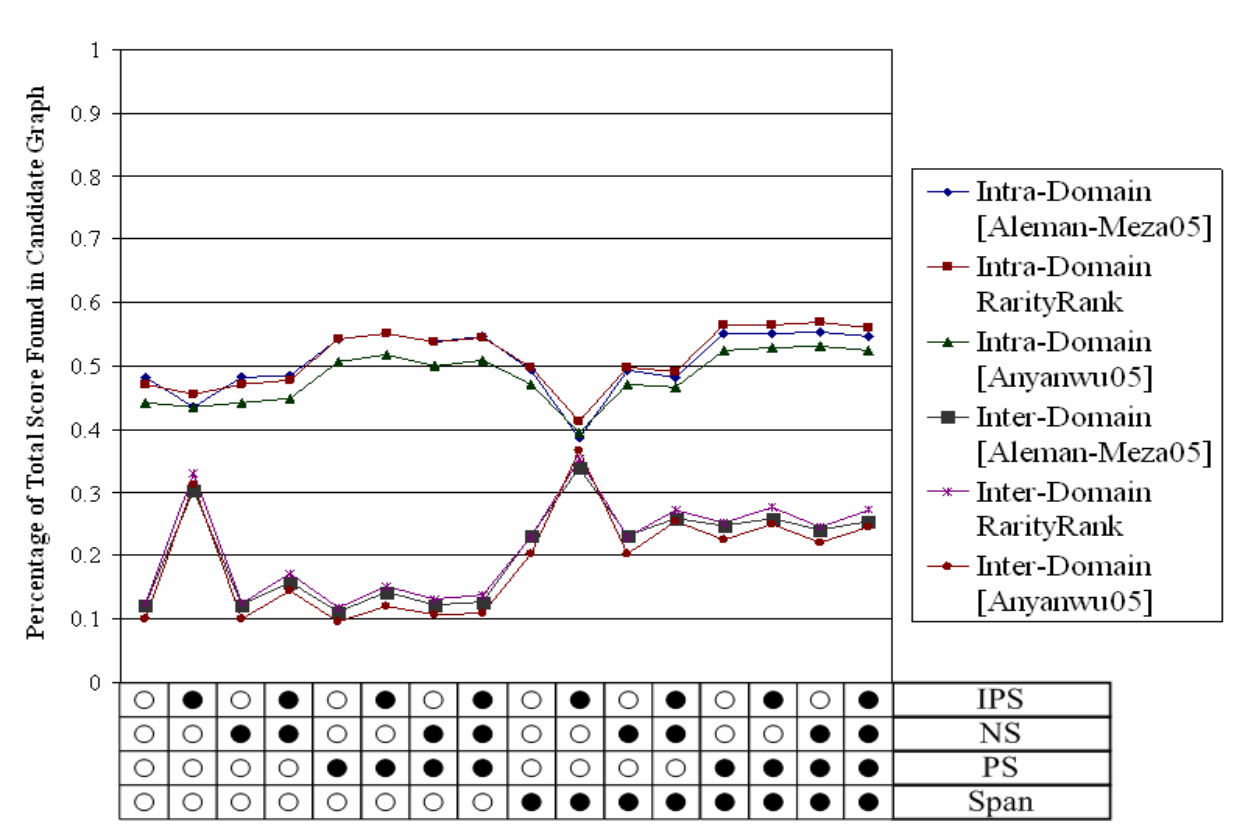


Figure 7.11. Percentage of Total Score found in Candidate ρ-graph.

An interesting aspect of Figure 7.10 is what we see for the 10th combination: SPAN and IPS. For both Inter-domain and Intra-domain queries, the candidate generation algorithm obtains subgraphs containing nearly the same percentage of total paths indicated by the peak for Inter-domain and the dip for Intra-domain queries. The types of paths between entities of different domains are likely to be rare resulting in a greater benefit when using a combination of the SPAN and IPS heuristics. Due to the commonality of path types within a domain, the opposite is illustrated for Intra-domain queries.

Figure 7.11 illustrates similar comparison with respect to the percentage of the total score rather than the number of paths. This graph was generated by computing the score given by obtaining a subgraph containing all paths between entities in a query, and then comparing that to the score for the corresponding *Candidate ρ-graph*. The figure shows the same trends as in Figure 7.10 with the combination of SPAN and IPS giving similar results for both Intra-domain queries ant Inter-domain queries. Considering that Figure 7.11 compares the score of a *Candidate ρ-graph* with the score of all paths, this trend should be mirrored in Figures 7.10 and 7.11.

7.6 Timing Evaluation

In regards to timing, we compared the speed of an exhaustive search over the full graph to the speed for computing a *Candidate ρ-graph* and performing a subsequent exhaustive search. The exhaustive search method used is a *bidirectional join* algorithm. We built a generated database table for the triples in the RDF graph—a *triple table*. The table contains a pair of tuples for each triple: the second tuple in each pair is an inverse of the first. We utilized secondary indexes on the endpoint of each triple. The algorithm takes as input the two entities and a *k*-hop limit. It then iteratively joins the *triple table* on itself from opposite directions, where each directional table corresponds to one of the endpoints. After each directional join, the current tables (produced from either direction) are joined (at the growing edges of the tables) to find the associations of less than length *k*. The algorithm halts upon joining the two directional tables such that the resulting table contains all associations of length *k*.

Table 7.1 shows results for *k*-hop limits from 5 to 8. The first second column in the table shows the average time, over the same 30 queries used in all of our evaluation, for the exhaustive search over the full graph; whereas, the third column shows the average time for a *Candidate ρ-graph* search (computing a *Candidate ρ-graph* and performing a subsequent exhaustive search on the *Candidate ρ-graph*). (All times are given in milliseconds.) The last column shows the ratio of the *Candidate ρ-graph* search to the full graph exhaustive search. While the ratios for a *k*-hop limit of 5 or 6 shows poor results for the computing a *Candidate ρ-graph* to exhaustively search over, the ratios for *k*-hop limits of 7 and 8 show much better results. While the time for an exhaustive search increases by at least an order of magnitude 10 with each successive hop limit, the same increase is much less for a *Candidate ρ-graph* search.

Table 7.1 Timing results. Comparison of an exhaustive search over the full graph to Candidate ρ-graph search (computation of a *Candidate ρ-graph* and a subsequent exhaustive search over the ρ-graph).

k-hop limit	Full graph search *(λ)*	Candidate ρ-graph search *(φ)*	Ratio: ϕ/λ
5	504	2389.313	4.740699
6	1,686	2617.063	1.552232
7	17,354	3808.938	0.219485
8	1,261,099	76063.88	0.060316

7.6 Sample Query Result

Figure 7.12 shows the TouchGraph [TG] representation of a result obtained for the query described in the scenario in chapter 6. The semantic association described in Figure 6.1 for the scenario is shown by the darkened nodes and edge. The other, smaller association illustrated in Figure 6.1 also appears in Figure 7.12. Clearly, the result is a visualizable subgraph containing the relevant associations. The heuristics for this query were Instance Participation Selectivity and Property Specificity.

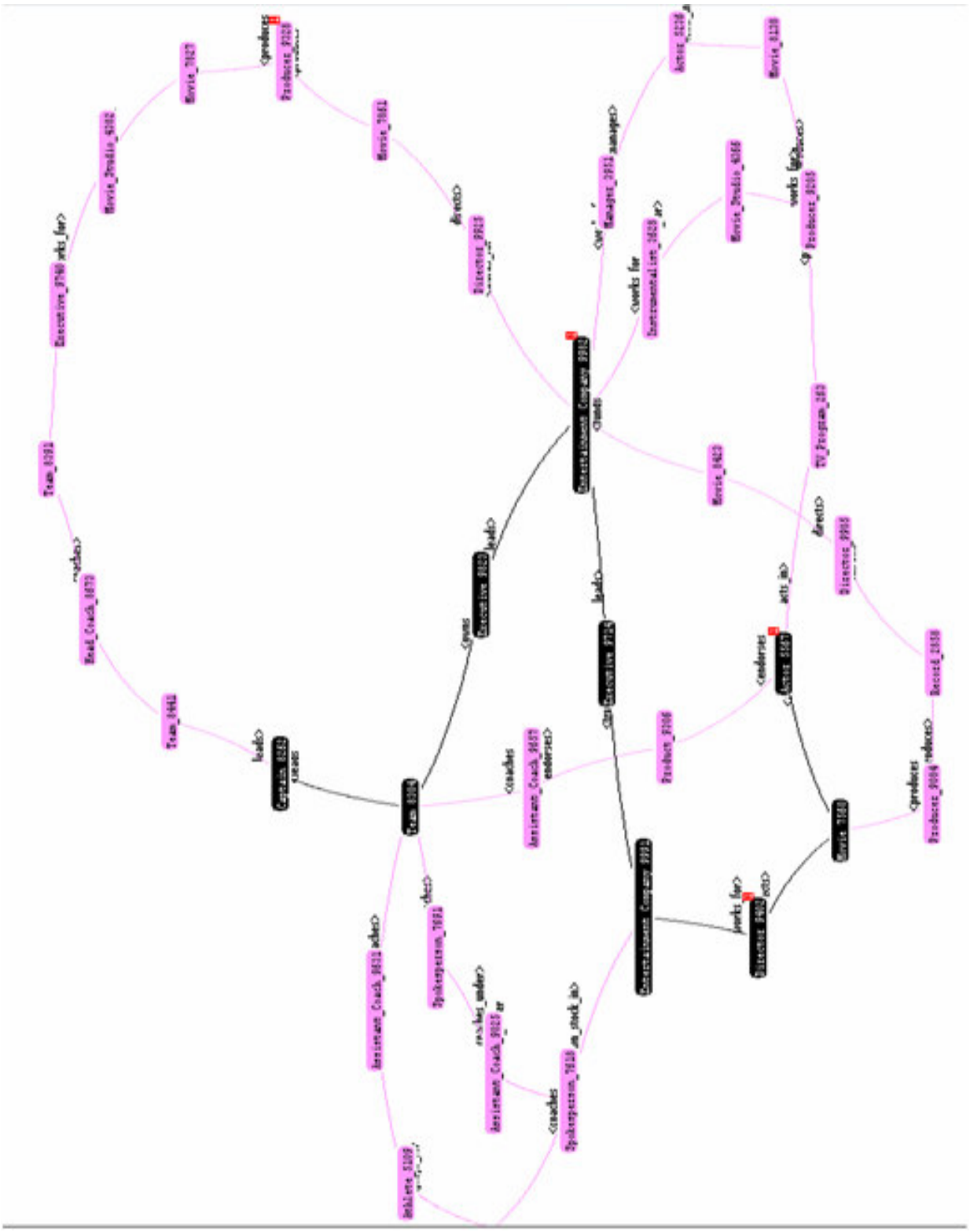


Figure 7.12. TouchGraph for Scenario Query Result. This is an example result from the scenario query. The image highlights the path described in the scenario which was obtained in the results.

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

In this thesis we address the growing need for better techniques to discovery meaningful associations in a Semantic Web. To this end, we have adapted the algorithms presented in [Faloutsos04] for use in semantic analytics. Our contribution was three-fold:

1. We utilize the degree of a node *v* when computing the distance on an edge *(u,v)*. Due to the fact that we are applying the described algorithms to (a main-memory representation of) an RDF graph model, we are able to know *a priori* the degree of *v*.
2. We apply the algorithm to graphs with multiple types of edges, as defined in the schema(s), rather than one edge-type, i.e., co-occurrence of names.
3. We compute edge weights based on the structure of the semantics by which the instance base is annotated.

We then conducted extensive empirical evaluation of the results of 30 analytic queries using all possible combinations of our heuristics for computing edge weights. The evaluation included separate examination of the affects of the heuristics on each of the algorithms developed in [Faloutsos04], as well as examination of both intra-domain and inter-domain queries.

Our results suggest that using edge weights generated by our weighting scheme results in highly relevant *Candidate ρ-graphs*, where relevance is judged using established path ranking metrics. Further evidence supporting this claim can be seen from quality of the *Display ρ-graphs*. In our experiments, the ranking metrics proposed in [Aleman-Meza05] show that the quality of the *Display ρ-graphs* are best when using Class Specificity (CS), Instance Participation Selectivity (IPS) and Span together. Results for the Successive Displays serve to support the electricity flow based model for RDF subgraph relevance, besides validating our edge weighting schemes. Results presented in this paper seem very promising for application domains like Ontology based Scientific Discovery where the ability to visualize relevant relationships between metadata entities is crucial. As a follow up to this work we plan to apply our techniques to develop tools for finding correlations between Glycosylation patterns and patterns of gene expression within a cell line in the Glycomics [Sheth04] domain.

We further propose to develop algorithms to support queries involving *n* endpoints for RDF graphs. For instance, in the scenario described herein, an analyst may want to ask a question such as "how do the two individuals in question relate to one another in respect to the company in which they sold all of their stocks?" Due to the candidate generation algorithm's linear time reduction of the search space, the algorithm could be extended to generate a candidate ρ-graph containing neighborhoods around all entity nodes in the query. A neighborhood would be found for each node, continually growing until the stopping condition is met. Furthermore, changes to the stopping conditions would have to be made in order to ensure that cut edges exist for all neighborhoods.

8.1 Using Closeness for Node Expansion

The method for deciding which node to expand next can be altered to consider the distance from one node to all nodes in the query. We have begun investigating methods that determine how close one node is to all other nodes. There are two notions of closeness [Friedkin91]: "closeness centrality" is a measure of how close one node is in respect to all other nodes, whereas "betweenness centrality" is a measure of pair wise closeness respective to each pair of nodes in the graph. To use these notions for candidate ρ-graph generation, closeness measures can be restricted to consider the closeness of a node in respect to only the nodes specified in the query. Our intuition is that the notion of closeness will help in neighborhood growth by ensuring that the neighborhoods grow towards one another. In the current implementation, these neighborhoods grow in a random direction irrespective of one another.

For answering such queries with n points, new methods for display ρ-graph generation need to be investigated. The current sense of network flow considers one source and one sink; however, this is not conducive for queries with n points. One method for closeness based on network flow [Brandes05] considers multiple source and sink nodes for an electrical network. However, care must be taken in deciding which query nodes are source nodes and which are sink nodes.

One further issue pertaining to *Display ρ-graph* generation based on network flows is the direction imposed on an edge by the sense of downhill flow in a network of electrical circuits. It may be the case that an edge (u_i, u_j), where $i < j$, exists in the highest ranked path from node s to node t. Yet, upon solving the system of linear equations necessary to determine the downhill flow in the electrical network, u_i may be found to be downhill form u_j; and thus, while the highest ranked path may have been found in the *Candidate ρ-graph*, no path containing the edge (u_i, u_j) can be added to the resulting display graph. This possibility increases the need to investigate other methods of generating a *Display ρ-graph*.

8.2 Defining and Specifying Context

Another interesting direction involves formalizing the notion of *Context* and investigating *Context-Aware Subgraph Discovery* algorithms. Furthermore, results of a query could be used define context. For instance, a localized representation of the schema(s) could be generated respective of the resulting subgraph, and then used refined for further querying.

8.3 Candidate ρ-graph and ranking metrics

While we have shown that the algorithm and heuristics for *Candidate ρ-graph* generation return high quality results, it is evident that the *Display ρ-graph* does not perform quite so well compared to ranking metrics, even given a high-quality *Candidate ρ-graph*. One reason for this is the direction imposed on edges by computing voltages and currents on nodes and edges, respectively. Consider a situation in which the highest ranked path in the *Candidate ρ-graph* contains an edge (u, v), such that u is closer (in respect to number of hops) to the source node than is v, yet upon solving the system of linear equations, the voltage on v is greater the voltage on u. In such a situation, the highest ranked path in the *Candidate ρ-graph* cannot be added to the *Display ρ-graph*. A further disadvantage of generating the *Display ρ-graph* is the need to

solve the system of linear equations: this becomes very time consuming as the size of the *Candidate ρ-graph* increases.

Thus, due to the performance of the former algorithm, along with its linear time reduction (a node and/or edge is visited only once during *Candidate ρ-graph* generation) of the search, a system could be developed to combine candidate generation and ranking metrics. The resulting combination would be a two step process: prune the search space to obtain a *Candidate ρ-graph*, and then search for all paths utilizing a ranking scheme. The outcome could either be all paths listed in order of rank, or a subgraph containing the top k paths (as in the case of the *Pseudo-Display ρ-graph*).

REFERENCES

[Adibi04] Jafar Adibi, Hans Chalupsky, Eric Melz and Andre Valente. The KOJAK Group Finder: Connecting the Dots via Integrated Knowledge-Based and Statistical Reasoning. In Proceedings of the Sixteenth Innovative Applications of Artificial Intelligence Conference (IAAI-04), 2004.

[Aleman-Meza03] B. Aleman-Meza, C. Halaschek, I. B. Arpinar, and A. Sheth, Context-Aware Semantic Association Ranking, First International Workshop on Semantic Web and Databases, Berlin, Germany, September 7-8, 2003, pp. 33-50

[Aleman-Meza04] B. Aleman-Meza, C. Halaschek, A. Sheth, I. B. Arpinar, and G. Sannapareddy, "SWETO: Large-Scale Semantic Web Test-bed", In Proceedings of the 16th International Conference on Software Engineering & Knowledge Engineering (SEKE2004): Workshop on Ontology in Action, Banff, Canada, June 21-24, 2004, pp. 490-493.

[Aleman-Meza05] Boanerges Aleman-Meza, Christian Halaschek-Wiener, I. Budak Arpinar, Cartic Ramakrishnan, and Amit Sheth. Ranking Complex Relationships on the Semantic Web. To Appear in *IEEE Internet Computing, Special Issue - Information Discovery: Needles & Haystacks May-June 2005.*

[Albert02] Albert, R., and Barabási, A.-L., "Statistical mechanics of complex networks", Reviews of Modern Physics 7J (January 2002), 47-97.

[Alesso04] H. Peter Alesso: Semantic Search Technology. SIG SEMIS Semantic Web and Information Systems. 2004 http://www.sigsemis.org/columns/swsearch/SSE1104/

[Anyanwu03] Kemafor Anyanwu, Amit P. Sheth: ρ-Queries: enabling querying for semantic associations on the semantic web. WWW 2003: 690-699.

[Anyanwu05] Kemafor Anyanwu, Angela Maduko, Amit Sheth, SemRank: Ranking Complex Relationship Search Results on the Semantic Web. The 14th International World Wide Web Conference, (WWW2005), Chiba, Japan, May 10-14, 2005

[Beckett01] Beckett, D., The Design and Implementation of the Redland RDF Application Framework. in *Tenth International World Wide Web Conference*, (Hong Kong, 2001), ACM.

[Berners-Lee01] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web: A new form of Web content that is meaningful to computers will unleash a

revolution of new possibilities", Scientific American, May 2001.

[Brandes05] Ulrik Brandes and Daniel Fleischer: Centrality Measures Based on Current Flow. Proc. 22nd Symp. Theoretical Aspects of Computer Science (STACS '05). LNCS 3404, pp. 533-544. (c) Springer-Verlag, 2005.

[Broekstra02] Broekstra, J., Kampman, A. and Harmelen, F.v., Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. in *International Semantic Web Conference 2002*, (Sardinia, Italy, 2002).

[Carroll04] J. J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, K. Wilkinson. Jena: Implementing the Semantic Web Recommendations. The 13th International World Wide Web Conference, (WWW2005), New York, New York, USA, May 17-22, 2004

[Cerebra] Cerebra, Inc.. http://cerebra.com/

[Chakrabarti04] Deepayan Chakrabarti, Yiping Zhan, Christos Faloutsos: R-MAT: A Recursive Model for Graph Mining. SDM 2004

[CIRAS] Semagix-CIRAS Anti-Money Laundering, Semagix, Inc. http://www.semagix.com/solutions_ciras.html

[Corese] Corese Semantic Web Fectory Homepage: http://www-sop.inria.fr/acacia/soft/corese/index.html

[Dill03] DILL, S., EIROL, N., GIBSON, D., GRUHL, D., GUHA, R., JHINGRAN, A., KANUNGO, T., RAJAGOPALAN, S., TOMKINS, A., TOMLIN, J. A., AND ZIEN, J.Y. 2003. SemTag and Seeker: Bootstrapping the semantic Web via automated semantic annotation. In Proceedings of the 12th International World Wide Web Conference (WWW-2003) (Budapest, Hungary, May 20-24 2003).

[Eliassi-Rad05] Tina Eliassi-Rad and Edmond Chow. Using Ontological Information to Accelerate Path-Finding in Large Semantic Graphs: A Probabilistic Approach. American Association for Artificial Intelligence 2005

[Faloutsos04] Christos Faloutsos, Kevin S. McCurley, Andrew Tomkins: Fast discovery of connection subgraphs. KDD 2004: 118-127.

[Flake02] Gary Flake, Steve Lawrence, C. Lee Giles, Frans Coetzee. Self-Organization of the Web and Identification of Communities. IEEE Computer, 35(3), 66-71, 2002.

[Friedkin91] Noah E. Friedkin. Theoretical foundations for centrality measures. American Journal of Sociology, 96(6):1478--1504, May 1991.

[Gibson98] David Gibson, Jon Kleinberg, Prabhakar Raghavan. Inferring Web Communities from Link Topology. In Proceedings of Ninth ACM Conference on Hypertext and Hypermedia, pages 225-234, New York, 1998.

[Gruber03] Thomas Gruber. It Is What It Does: The Pragmatics of Ontology. Invited presentation to the meeting of the CIDOC Conceptual Reference Model committee, Smithsonian Museum, Washington, D.C., March 26, 2003.

[Guha03] Ramanathan V. Guha, Rob McCool, Eric Miller: Semantic search. WWW 2003: 700-709.

[Hammond02] HAMMOND, B., SHETH, A., AND KOCHUT, K. 2002. Semantic Enhancement Engine: A Modular Document Enhancement Platform for Semantic Applications over Heterogeneous Content in Real World Semantic Web Applications. V. Kashyap & L. Shklar, Eds., IOS Press.

[Handschuh02] HANDSCHUH, S., STAAB, S., AND CIRAVENGA, F. 2002. S-CREAM Semi-automatic CREAtion of Metadata. In Proceedings of the 13th International Conference on Knowledge Engineering and Management (Sigüenza, Spain October 1-4 2002).

[Handschuh03] HANDSCHUH, S., AND STAAB, S. 2003. CREAM CREAting Metadata for the Semantic Web. Computer Networks. 42: 579-598, Elsevier.

[Haase04] Peter Haase, Jeen Broekstra, Andreas Eberhart, Raphael Volz. A Comparison of RDF Query Languages. In *Proceedings of the Third International Semantic Web Conference, Hiroshima, Japan, 2004*

[Huan04] Luke Huan, Wei Wang, Jan Prins, SPIN: Mining Maximal Frequent Subgraphs from Graph Databases. In Proceedings of the 2004 Conference on Knowledge Discovery and Data Mining (SIGKDD2004), 2004.

[Janik05] Maciej Janik and Krys Kochut. BRAHMS: A Workbench RDF Store and High Performance Memory System for Semantic Association Discovery. In *International Semantic Web Conference 2005*, (Galway, Ireland, 2005). (to appear).

[Jena] Jena Homepage: http://www.hpl.hp.com/semweb/jena.htm

[Karvounarakis02] G. Karvounarakis, S. Alexaki, V. Christophides, D. Plexousakis, M. Scholl, RQL: A Declarative Query Language for RDF, WWW2002, May 7-11, 2002, Honolulu,Hawaii, USA.

[Kleinberg99] Jon M. Kleinberg: Authoritative Sources in a Hyperlinked Environment. J. ACM 46(5): 604-632 (1999)

[Kuramochi02] Michihiro Kuramochi, George Karypis. Grew – A Scalable Frequent Subgraph Discovery Algorithm. In Proceedings of 2002 IEEE International Conference on Data Mining (ICDM), 2002.

[Lassila99] O. Lassila and R. Swick. Resource Description Framework (RDF) Model and Syntax Specification, W3C Recommendation. 1999.

[Lin03] Shou-de Lin, Hans Chalupsky: Unsupervised Link Discovery in Multi-relational Data via Rarity Analysis. ICDM 2003: 171-178

[McBride01] McBride, B. 2002. Jena: Implementing the RDF Model and Syntax Specification. Proceedings of the Second International Workshop on the Semantic Web (Hong Kong, China, May 1 2001).

[Milgram67] Stanley Milgram, "The Small World Problem", Psychology Today, May 1967. pp 60 – 67.

[Mukherjea04] Sougata Mukherjea, Bhuvan Bamba: BioPatentMiner: An Information Retrieval System for BioMedical Patents. VLDB 2004: 1066-1077

[MySQL] MySQL: http://www.mysql.com/

[Ontoprise] Ontoprise. http://www.scai.fraunhofer.de/829.0.html

[Oracle] Oracle Corporation: http://www.mysql.com/

[Page98] L. Page, S. Brin, R. Motwani, T. Winograd. "The PageRank Citation Ranking: Bringing Order to the Web", Stanford Digital Libraries Working Paper, 1998.

[Perry05] Matthew Perry "TOntoGen: A Synthetic Data Set Generator for Semantic Web Applications", AIS SIGSEMIS Bulletin Volume 2 Issue 2 (April - June) 2005, pp. 46 - 48

[Polikoff03] I. Polikoff and D. Allemang, "Semantic Technology," TopQuadrant Technology Briefing v1.1, September 2003. http://www.topquadrant.com/documents/TQ04_Semantic_Technology_Briefing.PDF

[PostgreSQL] PostgreSQL: http://www.postgresql.org/

[Protégé] Protégé. http://protege.stanford.edu/

[Prud'hommeaux05] Eric Prud'hommeaux and Andy Seaborne. 2005. SPARQL Query Language for RDF. W3C Working Draft, available at http://www.w3.org/TR/rdf-sparql-query/

[RDFCore] RDF Core Working Group: http://www.w3.org/2001/sw/RDFCore/

[RDFS] RDF Vocabulary Description Language 1.0: RDF Schema http://www.w3.org/TR/rdf-schema/

[RDFSemantics] RDF Semantics http://www.w3.org/TR/rdf-mt/

[Redland] Redland Homepage: http://librdf.org/

[Seaborne04] Seaborne, A. 2004. RDQL - A Query Language for RDF. W3C Submission, WWW Consortium (Cambridge, MA 2004), available at http://www.w3.org/Submission/2004/SUBM-RDQL-20040109/

[Semagix] Semagix, Inc. http://www.semagix.com/

[SemDis] Semantic Discovery: Discoverying Complex Relationships in Semantic Web. http://lsdis.cs.uga.edu/projects/semdis/

[SemDisAPI] SemDis API.

http://lsdis.cs.uga.edu/projects/semdis/sweto/index.php?page=5

[SeRQL] J. Broekstra and A. Kampman: The SeRQL Query Language. Technical Report, Aduna, 2003.

[Sesame] Sesame: http://www.openrdf.org/

[Sheth02] A. Sheth, C. Bertram, D. Avant, B. Hammond, K. Kochut and Y. Warke. Semantic Content Management for Enterprises and the Web, IEEE Internet Computing, July/August 2002, pp. 80-87.

[Sheth03] A. Sheth, I. B. Arpinar, and V. Kashyap, "Relationships at the Heart of Semantic Web: Modeling, Discovering, and Exploiting Complex Semantic Relationships," Enhancing the Power of the Internet Studies in Fuzziness and Soft Computing, M. Nikravesh, B. Azvin, R. Yager and L. Zadeh, Springer-Verlag, 2003.

[ShethandAvant04] Amit Sheth and David Avant. Semantic Visualization: Interfaces for exploring and exploiting ontology, knowledgebase, heterogeneous content and complex relationships. NASA Virtual Iron Bird Workshop, March 31 and April 2, CA

[Sheth04] Amit Sheth, William York, Christopher Thomas, Meenakshi Nagarajan, John A. Miller, Krys Kochut, Satya S. Sahoo, Xiaochuan Yi, **"**Semantic Web technology in support of Bioinformatics for Glycan Expression," W3C Workshop on Semantic Web for Life Sciences, 27-28 October 2004, Cambridge, Massachusetts USA.

[ShethDROPS05] Amit Sheth. From Semantic Search & Integration to Analytics. **http://drops.dagstuhl.de/opus/volltexte/2005/46/**

[Sheth05] Amit Sheth. Enterprise Applications of Semantic Web: The Sweet Spot of Risk and Compliance. Invited paper: IFIP International Conference on Industrial Applications of Semantic Web (IASW2005), Jyväskylä, Finland, August 25-27, 2005. http://www.cs.jyu.fi/ai/OntoGroup/IASW-2005/

[Shethetal05] A. Sheth, B. Aleman-Meza, I. B. Arpinar, C. Halaschek, C. Ramakrishnan, C. Bertram, Y. Warke, D. Avant, F. S. Arpinar, K. Anyanwu, and K. Kochut, Semantic Association Identification and Knowledge Discovery for National Security Applications, Journal of Database Management on Database Technology, 16(1):33-53, Jan-March 2005, Eds: L. Zhou and W. Kim

[Thacker03] Sanjeev Thacker, Amit Sheth, and Suchi Patel. Complex Relationships for the Semantic Web. Spinning the Semantic Web, D. Fensel, J.

Hendler, H. Liebermann, and W. Wahlster (eds.), MIT Press, pp. 297-316.

[TG] TouchGraph Homepage: http://www.touchgraph.com/

[Xu05] Jennifer XU and Hsinchun Chen. Criminal Network Analysis and Visualization. In Communications of the ACM, Volume 48, Issue 6, 100-107. 2005

[Yan03] Xifeng Yan, Jiawei Han. CloseGraph: Mining Closed Frequent Graph Patterns. In Proceedings of the 2003 Conference on Knowledge Discovery and Data Mining (SIGKDD2003), 2003.

APPENDIX

A.1 Evaluation Using Multiple Hop Limits

While we feel that using a 9-hop limit to evaluate our heuristics against ranking metrics is satisfactory, in this section we provide figures displaying evaluation based on hop limits of 5, 6, 7 and 8. However, comparing these two figures to Figure 7.1, we see that evaluation for paths limited to 9 hops indicates similar candidate quality ranging from that of 5 and 6 hop limits to 7 and 8 hop limits.

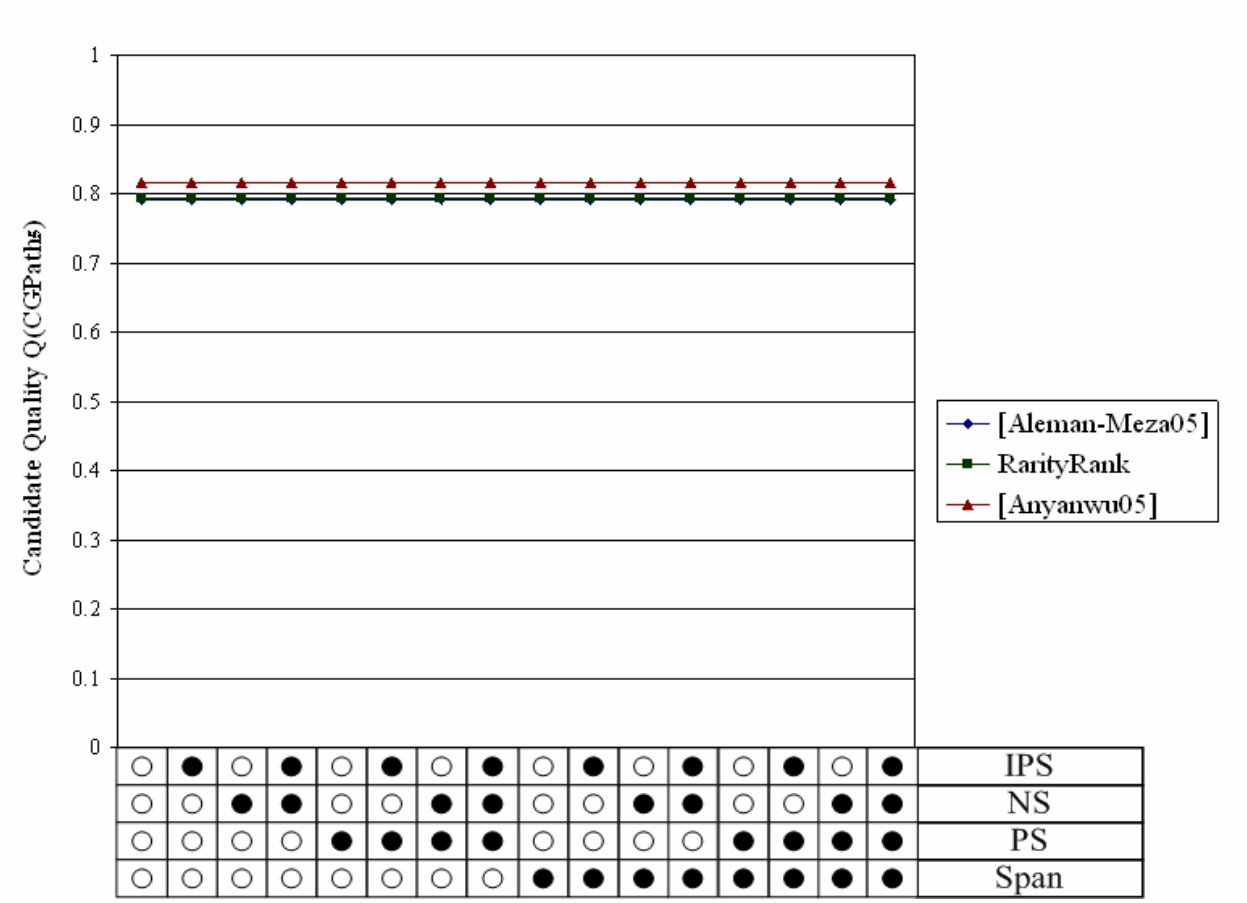


Figure A.1. Quality of Candidate ρ-graph for 5 hop limit.

Figures A.1 through A.4 reveal a 75-95% score for the quality of the Candidate Graph algorithm based on the heuristics presented herein. Interestingly, the Figures A.3 and A.4 portray better quality of candidate graphs in regards to 7 and 8 hop limits for ranked paths as compared to limits of 5 or 6 hops. Intuitively, this indicates that high quality paths of longer

sufficient length are being obtained during candidate generation. Evaluation of these inter- and intra-domain queries for these hop limits indicates similar trends to those seen in Figures 7.8 and 7.9 in that intra-domain queries show higher quality in general than inter-domain queries. Again, this is likely due to the typicall longer paths found for inter-domain queries over that of intra-domain queries.

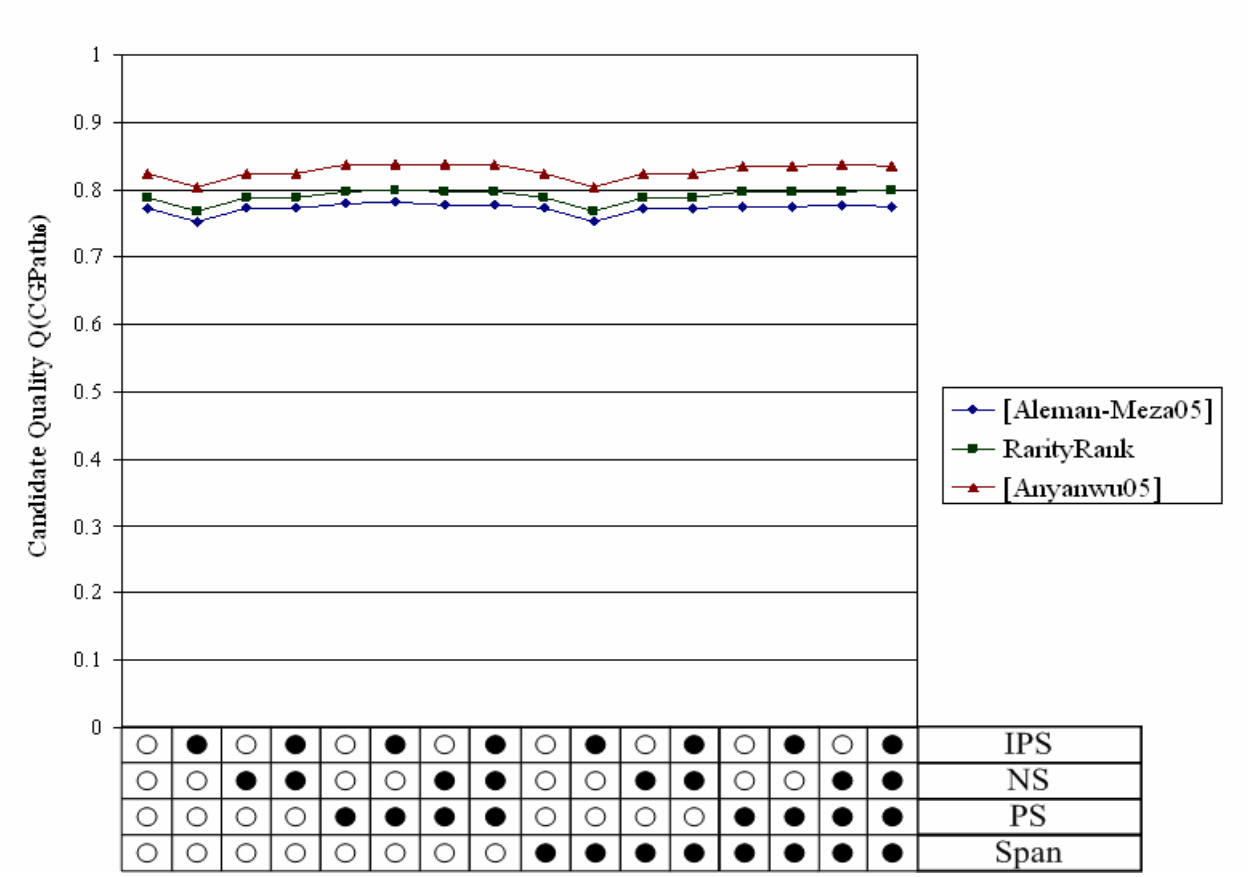


Figure A.2. Quality of Candidate ρ-graph for 6 hop limit.

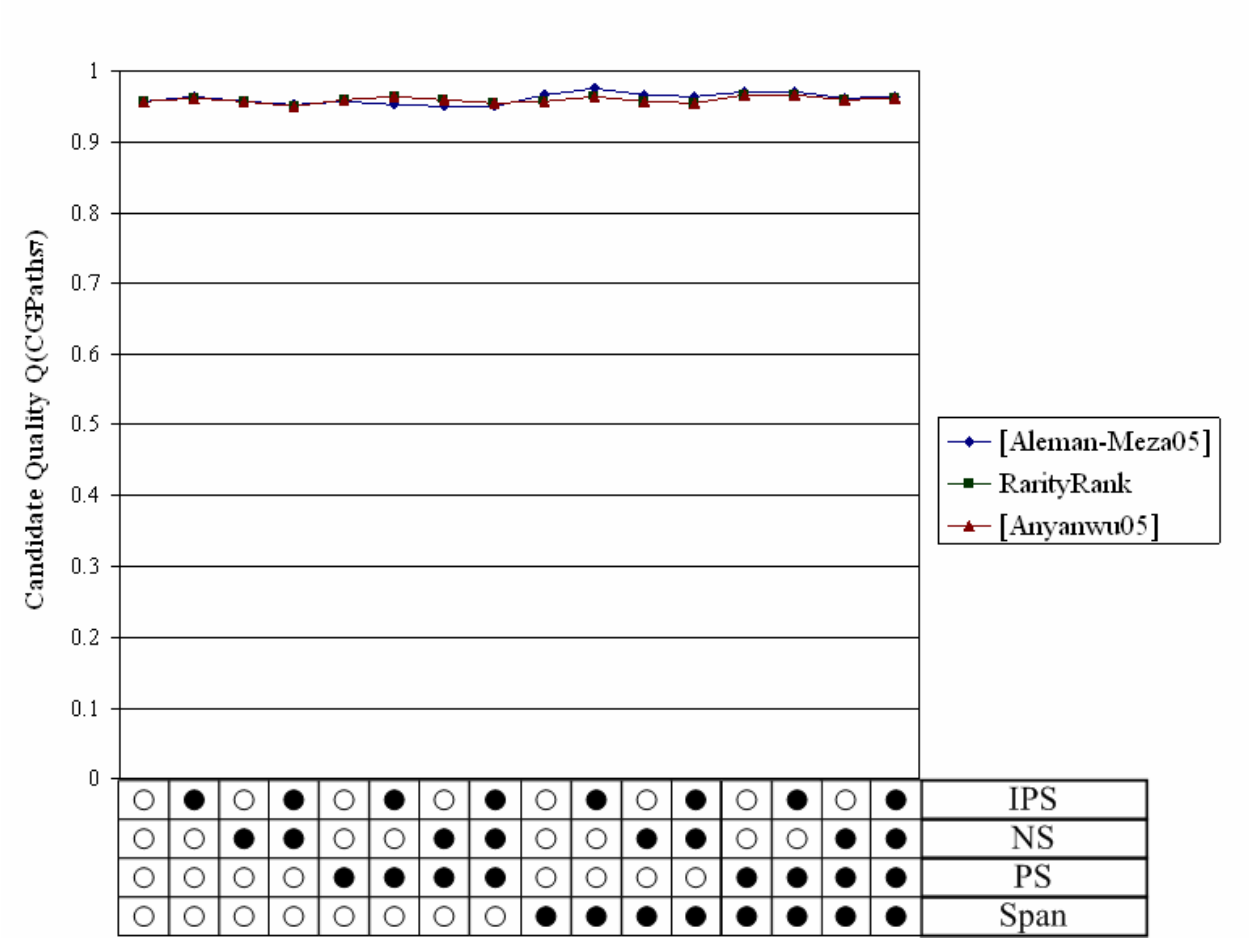


Figure A.3. Quality of Candidate ρ-graph for 7 hop limit.

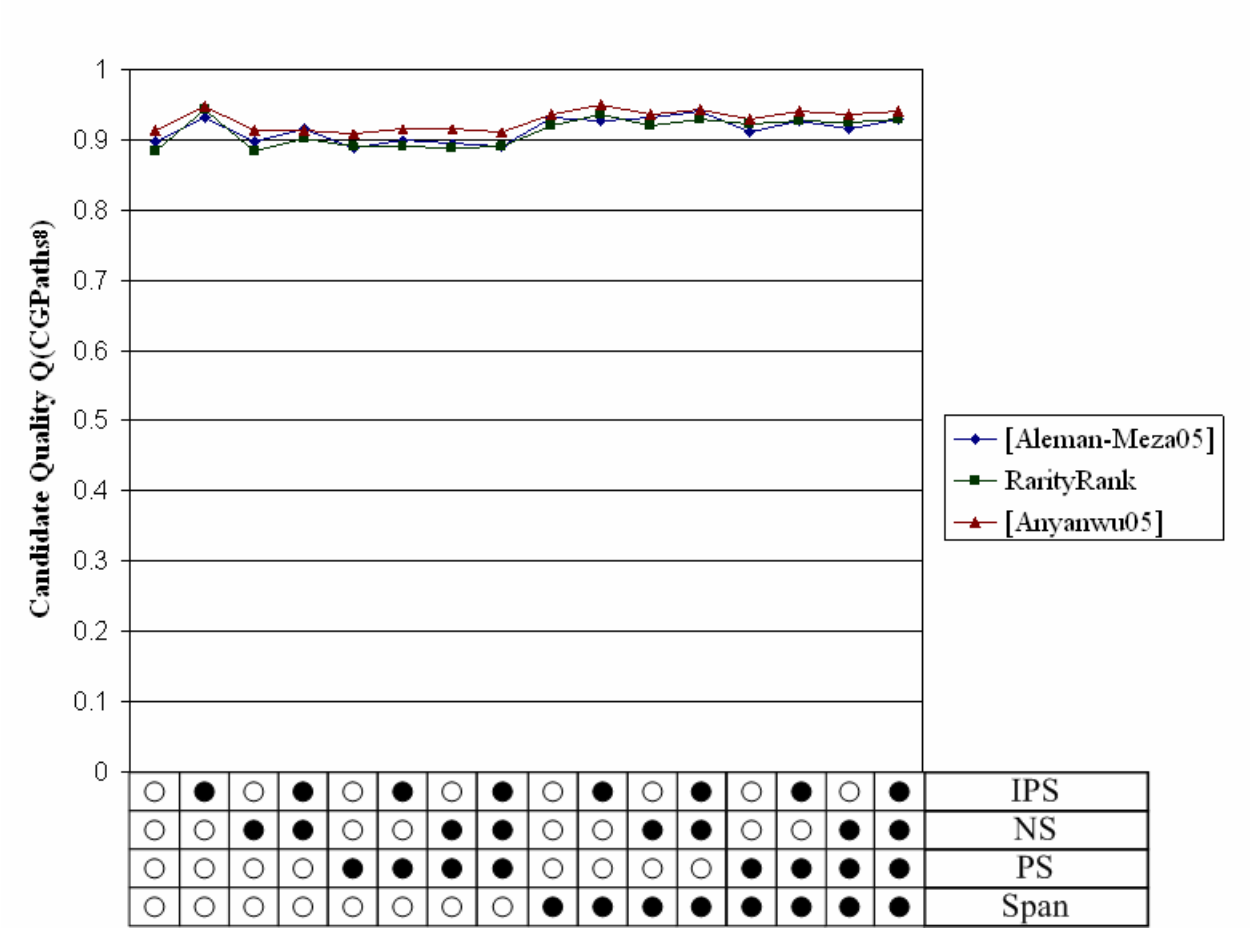


Figure A.4. Quality of Candidate ρ-graph for 8 hop limit.

A.2 Scenario Schemas

Figures A.5, A.6, and A.7 are graphical representations of the 3 schemas (business domain, entertainment domain, and sports domain) used for the synthetic dataset.

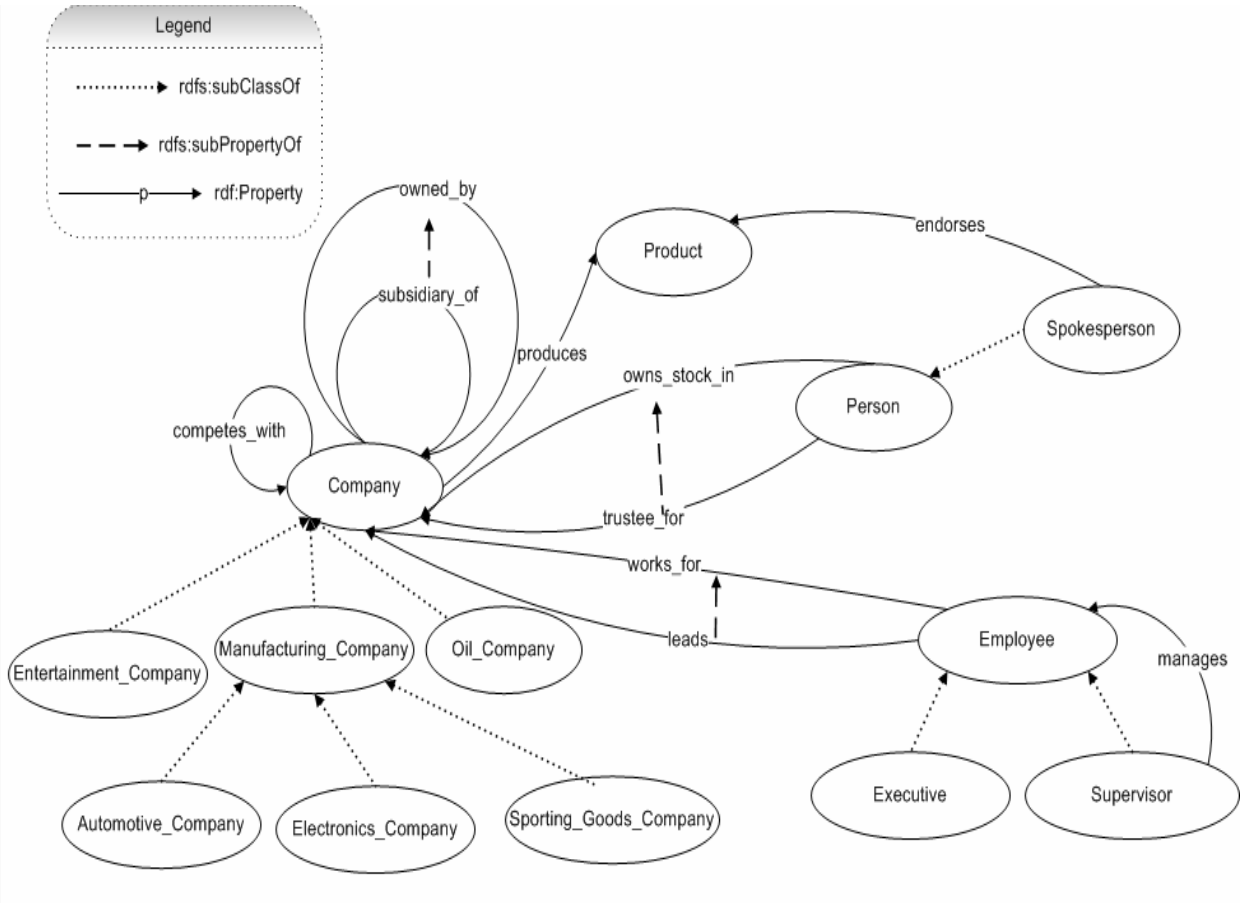


Figure A.2. Schema for the Business Ontology

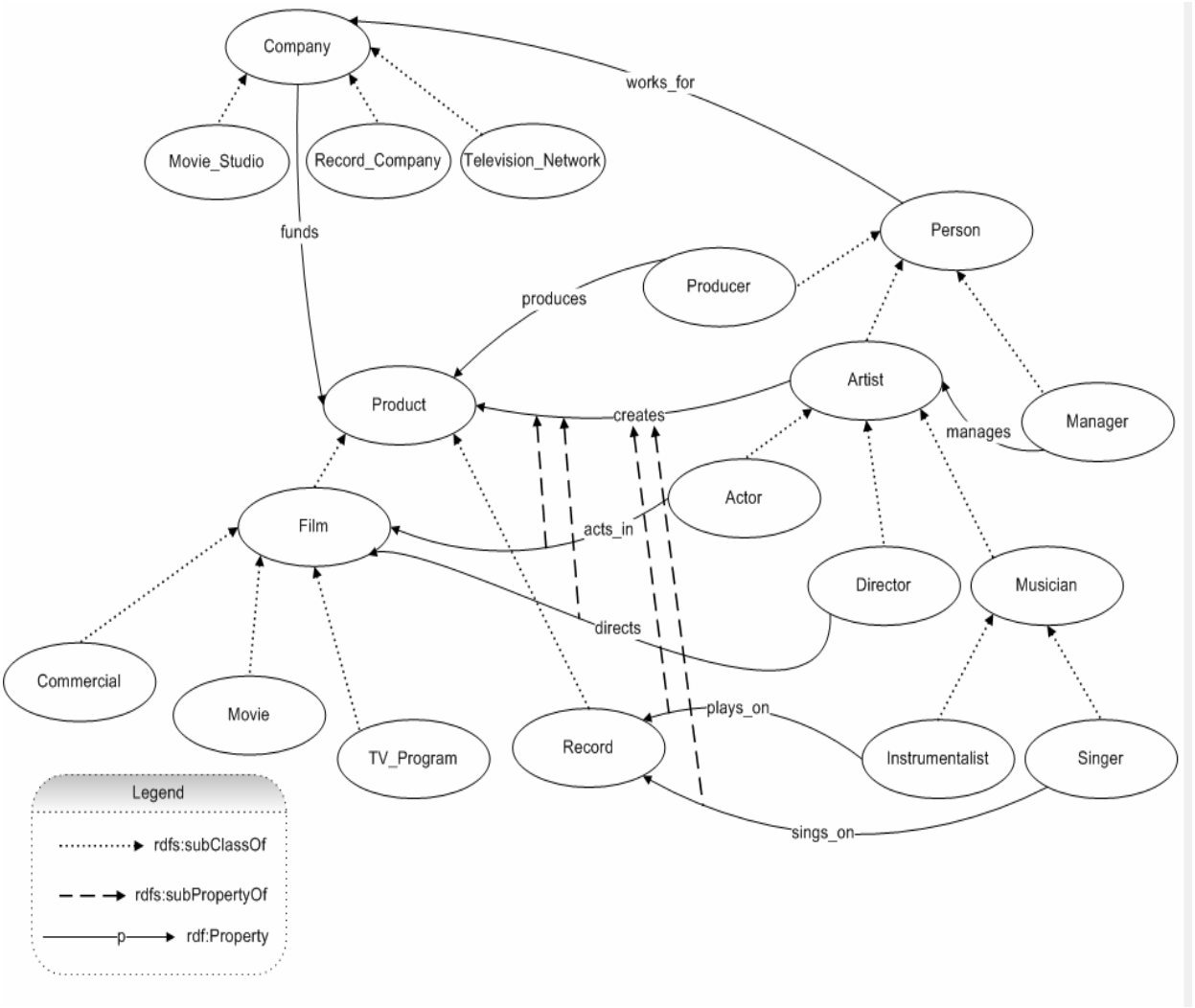


Figure A.2. Schema for the Entertainment Ontology

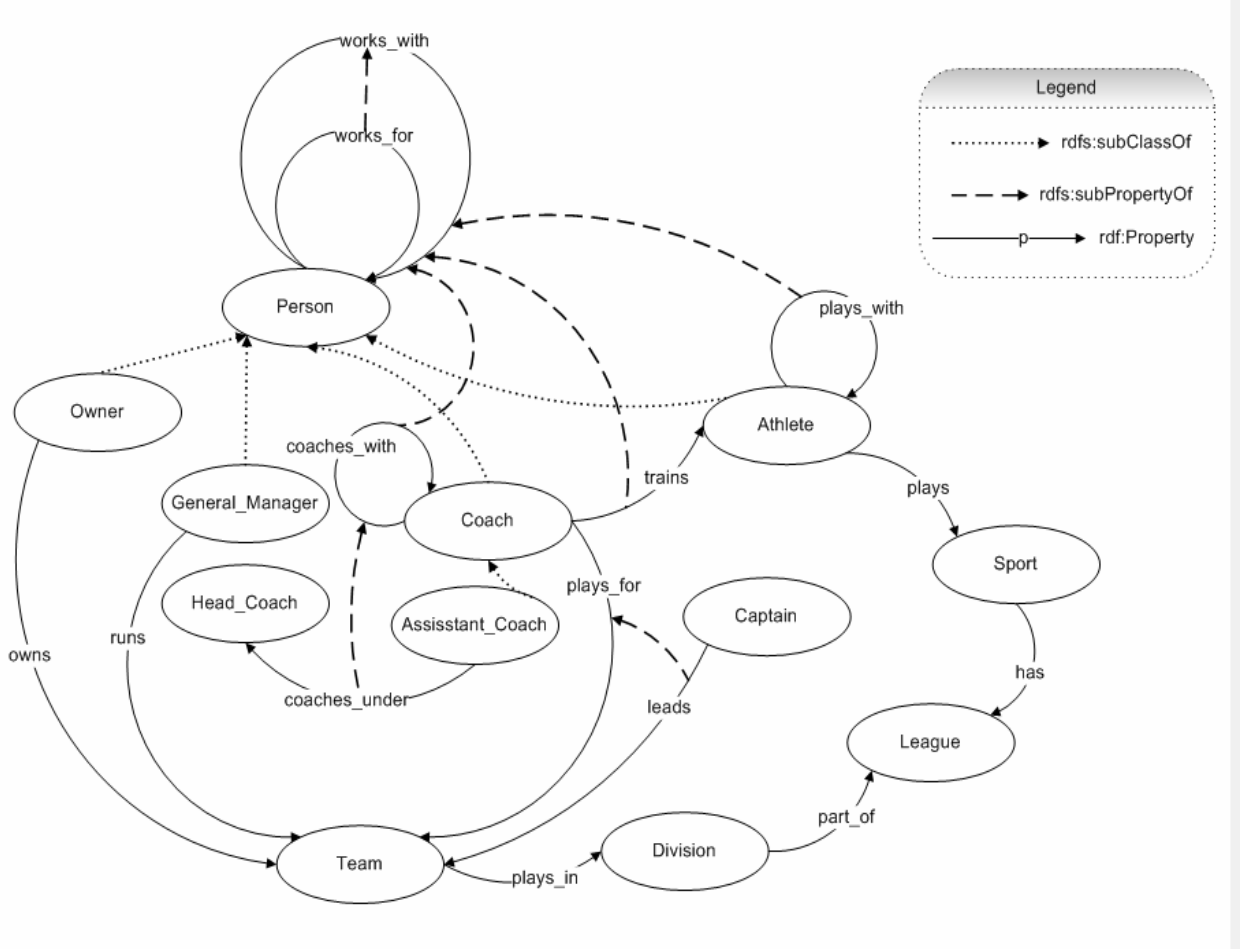


Figure A.3. Schema for the Sports Ontology