

Thesis Proposal

Vasco Calais Pedro

Thesis Committee :

Jaime Carbonell(Chair)

Eric Nyberg

Robert Frederking

Eduard Hovy(USC)

Abstract

In the last decades the number of available ontologies has grown considerably. These resources offer the promise of easily-accessible, open-domain ontological information, but the existence of such diverse ontologies raises the issue of information merging and reuse. A comparison of available ontologies reveals both redundant and complementary coverage, but the variety of frameworks and languages used for ontology development makes it a challenge to merge query results from different ontologies.

This research proposes to address this problem by building an ontological middleware level for only small fragments of ontologies in an on-demand basis by querying multiple ontologies and merging the query results from multiple knowledge base systems. We then follow ontological chains and inferences across ontologies, using partial query results from one ontology to query another. This is a more complex version of cross-data-base joins, where the data schemas are sufficiently compatible.

An initial evaluation used the federated ontology search for answer type checking in question answering. The results of the evaluation show that it is possible to obtain results that outperform querying ontologies independently in both precision and recall. Solutions to additional problems in querying multiple ontologies such as concept identification and ontology selection will be provided using string and structural similarity measures. We propose the creation of an evaluation framework using the question and answer sets in the TREC evaluations to account for a wider set of ontological operations.

1.	PROBLEM STATEMENT	1
1.1.	MOTIVATION.....	1
1.2.	MOTIVATING EXAMPLE.....	3
1.2.1.	<i>Question Answering</i>	4
1.2.2.	<i>Example 1</i>	4
1.3.	THESIS STATEMENT	9
1.4.	EXPECTED CONTRIBUTIONS	9
1.5.	DOCUMENT STRUCTURE	10
2.	RELATED RESEARCH	11
2.1.	ONTOLOGY SELECTION	11
2.2.	ONTOLOGY MAPPING.....	12
2.3.	ONTOLOGY EVALUATION.....	14
3.	METHODOLOGY	15
3.1.	BASIC DEFINITIONS.....	15
3.1.1.	<i>Ontologies and Graphs</i>	15
3.1.2.	<i>Query</i>	16
3.1.3.	<i>Result</i>	16
3.2.	ONTOLOGICAL SEARCH.....	17
3.2.1.	<i>Operators</i>	18
3.2.2.	<i>Query</i>	22
3.2.3.	<i>Resource Description and Selection</i>	24
3.2.4.	<i>Merging</i>	25
3.2.5.	<i>Scoring Results</i>	30
3.3.	KNOWN ISSUES WITH CURRENT WORK	31
3.3.1.	<i>Identification of the correct concepts and relations in ontologies</i>	31
3.3.2.	<i>Concept and edge similarity between concepts in different ontologies</i>	32
3.3.3.	<i>Chains of indirect inference</i>	32
3.3.4.	<i>Insensitivity in Boosting graphs of different structural properties</i>	32
4.	PRELIMINARY RESULTS	33
4.1.1.	<i>Type Checking</i>	33
4.1.2.	<i>Experimental Setup</i>	33
4.1.3.	<i>Results and Analysis</i>	34
5.	ROADMAP.....	36
5.1.	THESIS SCOPE	36
5.2.	RESEARCH ACTIVITIES	37
5.2.1.	<i>Graph Matching</i>	37
5.2.2.	<i>Ontological Metadata</i>	38
5.2.3.	<i>Query and Resource Description</i>	39
5.2.4.	<i>Resource Selection</i>	40
5.2.5.	<i>Result Merging</i>	42
5.2.6.	<i>Result Scoring</i>	42
5.3.	EVALUATION.....	43
5.3.1.	<i>Factoid TREC QA Task</i>	43
5.3.2.	<i>Possible evaluation tasks</i>	44
5.3.3.	<i>Condition of Success</i>	45
5.4.	TIMETABLE	46
6.	REFERENCES.....	47

1. Problem Statement

1.1. Motivation

Although several Ontology definitions co-exist [Guarino, 1998], an Ontology can be defined as a model that represents a domain and is used to reason about objects in that domain and the relations between them. It is usually composed of concepts, relations between those concepts, concept properties and instances. Within the scope of this work we consider taxonomies, semantic nets [Quillian, 1967] and lexical resources such as Wordnet [Miller, 1995] as ontologies. The use of ontologies is now widespread in areas as diverse as biomedical research, information extraction and knowledge engineering and management.

In the last decades the number of available ontologies has grown considerably. Several proprietary and open-domain ontologies such as Cyc [Lenat, 1995], SUMO [Niles and Pease, 2001], Omega [Philpot, et al., 2003], Scone [Fahlman, 2005], ThoughtTreasure [Mueller, 1997], Wordnet [Miller, 1995], VerbNet [Schuler, 2003], Framenet [Baker, et al., 1998] and Propbank [Kingsbury and Palmer, 2002] have become available. Swoogle [Ding, et al., 2004] has now indexed more than 10 000 ontologies. These resources offer the promise of easily-accessible, open-domain ontological information, but the existence of such diverse ontologies raises the issue of information merging and reuse. A comparison of the ontologies reveals both redundant and complementary coverage, but the variety of frameworks and languages used for ontology development makes it a challenge to merge query results from different ontologies. The number of available languages for ontological knowledge engineering such as RDF, OWL, DAML+OIL and CYCL, combined with the existence of independent interfaces aggravates the issue. The lack of a formal way to access and combine the knowledge from different ontologies is an obstacle to more effective re-use and combination of these resources.

One approach to the multi-ontology issue is to absorb all the knowledge into a common ontology ahead of time. However this approach has several drawbacks, as defined in

[Klein, 2001; Serafini and Tamin, 2005], such as (i) non-scalability, (ii) losing language and reasoning specificity of distinct ontologies, (iii) losing privacy and autonomy of ontological knowledge (iv) language level mismatches such as syntax mismatches, differences in logical representation and different semantic primitives and (v) Ontology level mismatches, such as difference in scope, coverage and granularity, making this challenge thus far too daunting in practice. A second approach is to query more than one ontology via different interfaces, and interpret the results of each ontology individually, essentially moving the entire challenge from the ontology provider to the application builder. A third approach is to build an ontological middleware level for only small fragments of ontologies in an on-demand basis, that is:

- Query multiple ontologies and then merge the query results from multiple knowledge base systems, much like Federated Search in information retrieval [Si and Callan, 2005].
- Follow ontological chains and inferences across ontologies, using partial query results from one ontology to query another. This is a more complex version of cross-data-base joins, where the data schemas are sufficiently compatible.

Currently, the main approaches to a solution for these problems focus on ontology integration, by creating a mapping between the concepts and relations of different ontologies. Some cases, such as the Semantic Web project [Bemers-Lee, et al., 2001] primarily rely on merging two ontologies by establishing a full mapping between them. Some efforts have tried to produce a merged ontology automatically using a bottom-up approach such as FCA-Merge [Stumme and Maedche, 2001]; most involve some degree of semi-supervised mapping. Other approaches, such as the one taken by CYC, try to absorb other ontologies into a single main ontology while maintaining coherence [Reed and Lenat, 2002]. One disadvantage of these approaches is the prohibitive cost of producing a mapping or absorbing an ontology, given their increasing scale and rate of availability. Another disadvantage is that it is not always possible to establish a one-to-one mapping between the concepts and relations in one ontology and the concepts and relations in another. Furthermore, there is the problem of keeping the mappings updated as the original ontologies evolve. A large number of available ontologies are considered

works in progress and are updated frequently, which implies a constant updating of any mappings associated with those resources.

Most applications that use ontological information would benefit from an approach that models the information need, queries the relevant ontologies and retrieves the best result while providing a single unified interface to the client application. If we look to other domains for inspiration on how to proceed, we can find a similar problem in the field of Federated Search [Callan, 2000; Fryer, 2004]. Information Retrieval is usually based on a single database model of text retrieval. But to cope with proprietary information spread around the world in separate databases, distributed information retrieval explicitly models multiple databases for text retrieval. Each database is queried independently, the results are merged when possible and a new global ranking is established.

In the same fashion, we can model our ontologies as individual sources, construct a query that describes the information need, query each ontology independently and merge the results into one ranked list.

Using Federated Ontology Search we can parallelize query execution while respecting the structure of the individual ontologies, taking advantage of both redundant and complementary knowledge in the available ontologies to improve the overall performance of the system.

1.2. Motivating Example

Although there are many applicable areas for this research, type checking in the factoid QA domain is suitable to prove the utility of our approach. The advantages of this are threefold. First, the application of this research in factoid QA can be well defined and the ontological operations involved are conceptually clear but yet not trivial. Second, the training and test data created for the TREC QA evaluations [Voorhees, 2003], consisting of corpora, question sets and answers keys, provide the required evaluation material.

1.2.1. Question Answering

In Question Answering, the goal is to take a question in natural language and provide an answer also in natural language. In the JAVELIN I question answering system [Nyberg, et al., 2003] a set of modules is used, specifically the Question Analyzer (QA) module, which analyzes the question; the Retrieval Strategist (RS) modules, which retrieves the relevant documents; the Information Extractor (IX) module, which analyzes the documents retrieved by the RS and provides candidate answers; and finally the Answer Generator (AG) module which analyzes the candidate answers and generates a final ranked list of answers. Ontological information is typically used within JAVELIN to determine the relation between the expected answer type and the candidate answers or to provide answer verification, if the answer can be found directly in an available ontological resource [Ko, et al., 2006].

1.2.2. Example 1

For this example let's assume that we have access to the following ontologies

Ontology
CYC
Wordnet
U.S. Gazetteer

Table 1 - Available Ontologies for example 1

Let's consider the question: *What is the largest city in Germany?*

Part of the QA module's responsibilities in analyzing the question is to determine, amongst other things, the type of answer we are expecting and the constraints on that answer. In this case the QA module determined that the type of answer is *location* with an additional constraint that the answer must be a *city*. Table 2 shows a partial output from the QA and Table 3 shows Javelin's output.

Question	What is the largest city in Germany
Answer Type	<i>location</i>
Subtype Constraint	<i>city</i>

Table 2 – Output of the Question Analyzer. We can see the constraints expected in the answer.

AG output		
Rank	Answer	Judgment
1	Italy	Not a City
2	Berlin (Correct)	Correct
3	Horten	Incorrect
4	Norway	Not a City
5	South Africa	Not a City
6	Dusseldorf	Incorrect
7	Spain	Not a City
8	Moscow	Incorrect
9	France	Not a City
10	Swiss	Not a City
11	London	Incorrect
12	Oslo	Incorrect
13	Cologne	Incorrect
14	Pretoria	Incorrect

Table 3 – Output of the Answer Generator

As we can see the correct answer is ranked second. Furthermore, the first ranked answer, as well as several others, does not obey the constraints set by the QA module because it is not a city. We need a way to enforce those constraints on the candidate answers. One possibility is to use ontological knowledge to verify these constraints, but given that QA is an open domain field, there is presently no one ontology with the adequate coverage.

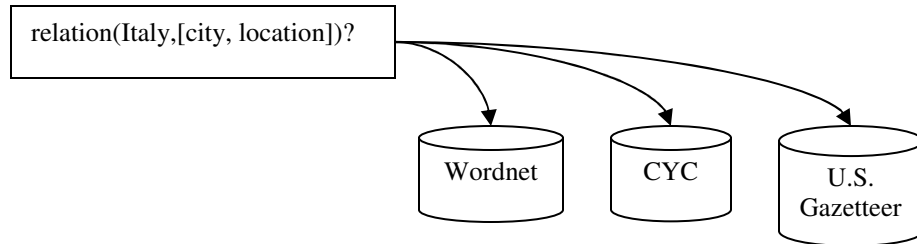
Consider an example which illustrates the federated ontology search approach. The main idea in this case is that we would submit each of the candidate answers along with the answer type constraints for verification.

I will show the procedure for the first two answers in the ranked set. The rest of the answers would proceed in similar fashion.

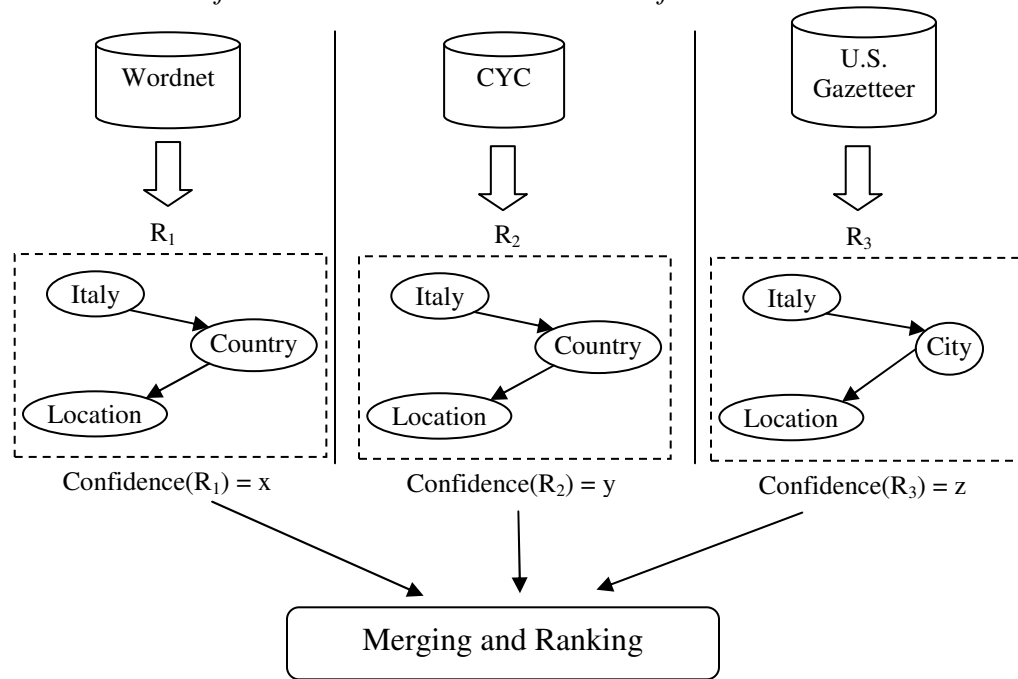
Constraints : *city, location*

a) Answer1 : *Italy*

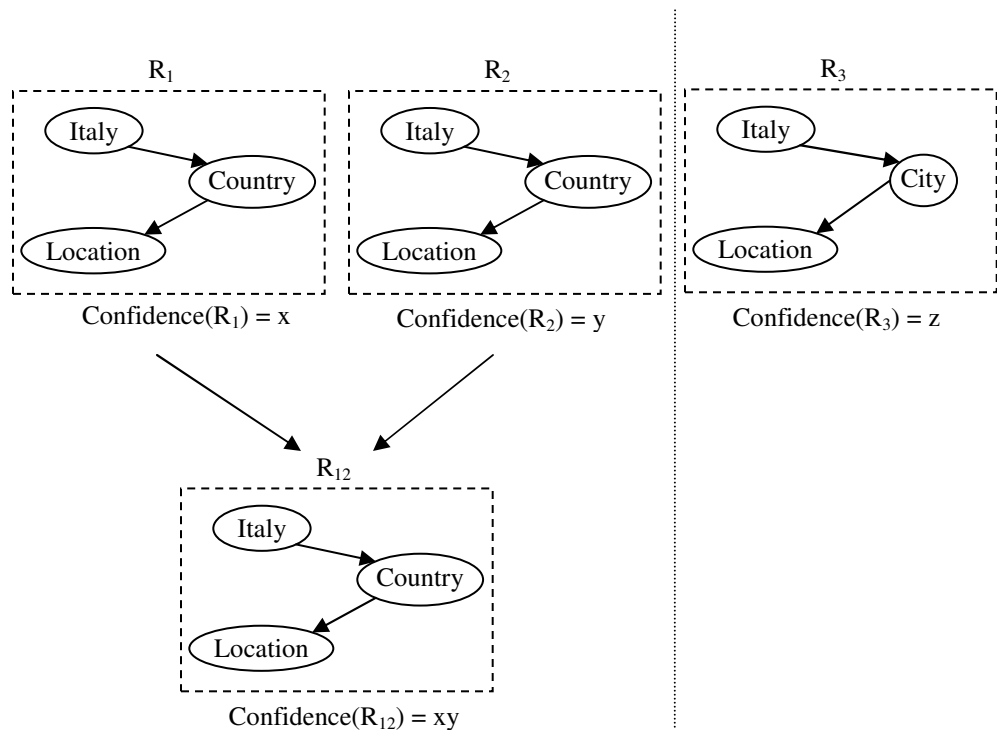
a. *The available ontologies are queried regarding if there is any relation between Italy and city*



b. *The results are collected. Each result contains a confidence of the correctness of the result as well as a source confidence*



c. *Merge and rank the results. When two results are merged, their confidence is boosted*



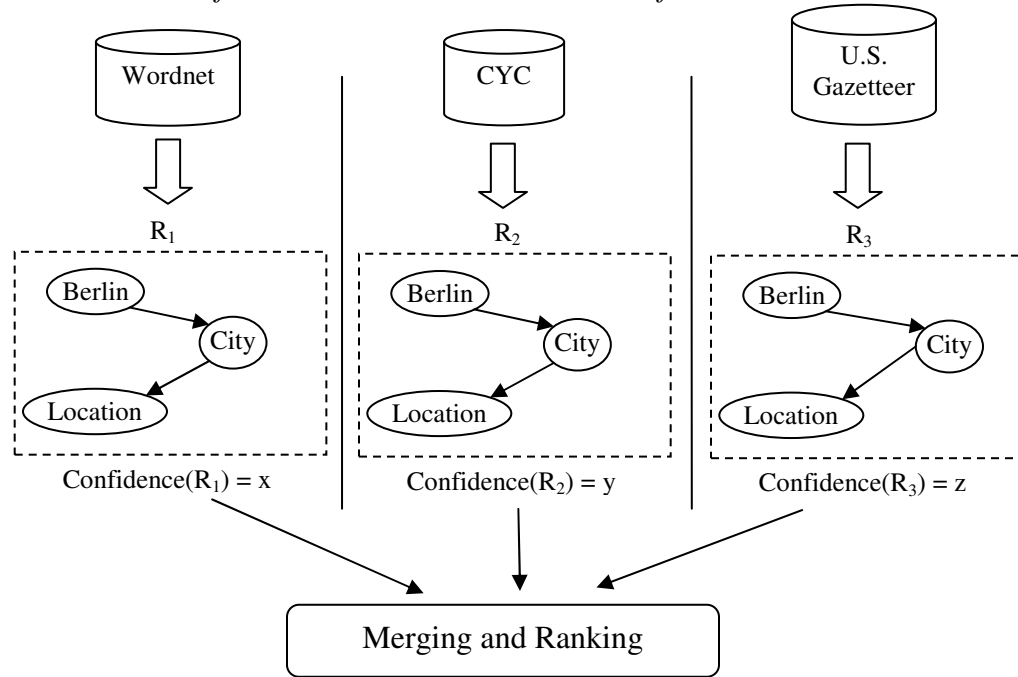
d. Finally a ranked list is produced by the server and the highest ranking answer is that Italy is a country rather than a city.

Rank	Result
1	<p>Confidence(R_{12}) = xy</p>
2	<p>Confidence(R_3) = z</p>

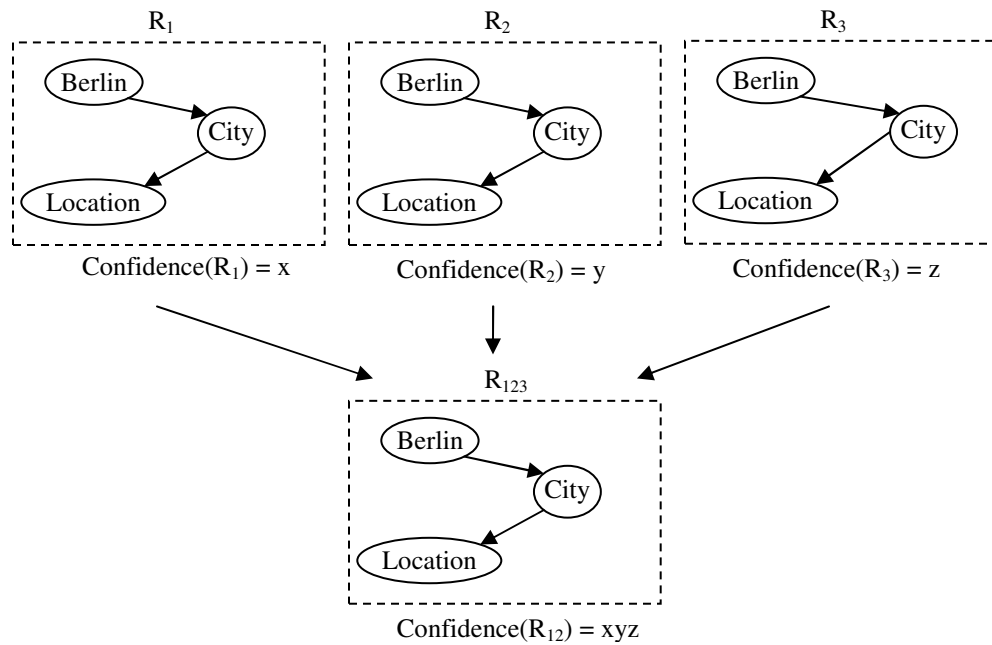
b) Answer 2: *Berlin*

a. Step a would be the same as before

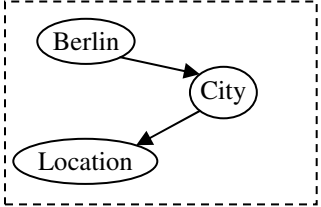
b. The results are collected. Each result contains a confidence of the correctness of the result as well as a source confidence



c. Merge and rank the results. When two results are merged, their confidence is boosted.



d. Finally a ranked list is produced by the server shows Berlin as the only answer

Rank	Result
1	Confidence(R_{123}) = xyz 

After applying this procedure to all the answers, Table 4 shows the final ranked list of answers. We can see that the correct answer is now on top.

AG output		
Rank	Answer	Judgment
2	Berlin (Correct)	Correct
3	Horten	Incorrect
6	Dusseldorf	Incorrect
8	Moscow	Incorrect
11	London	Incorrect
12	Oslo	Incorrect
13	Cologne	Incorrect
14	Pretoria	Incorrect

Table 4 – Output of the Answer Generator

1.3. Thesis Statement

Our hypothesis is that given a set of ontologies it is possible to use the combined knowledge contained in those ontologies while maintaining the individual ontologies in a distributed approach. This can be achieved by defining a set of elementary ontological operations which allow for the creation of complex ontological queries through composition of the simpler operations.

1.4. Expected Contributions

The proposed research is expected to contribute the following;

- A novel framework for ontology integration, including:

- A novel ontology-independent query algorithm for ontologies.
- A merging algorithm for merging ontological results.
- A scoring metric for scoring merged results.
- A demonstration that it is possible to use multiple ontologies while keeping them independent.
- A similarity metric for ontological results.
- A task-based evaluation task for ontologies, as well as suitable data set.

1.5. Document Structure

The rest of the document is organized as follows. Chapter 2 surveys the current literature on the relevant subjects. Chapter 3 describes the methodology used and the preliminary results conducted so far to determine the feasibility of our approach. In Chapter 4 we describe the planned experiments to demonstrate our claims and the proposed evaluation of those experiments. At the end of chapter 4 we describe the expected timetable for the thesis work.

2. Related Research

This work intersects with many areas of research, it is impossible to be exhaustive when describing related research given the quantity of relevant works. We will address some of the most salient works in each area of interest and try to give an overview of the different methods that are relevant to the proposed approach.

2.1. Ontology Selection

Ontology selection deals with the selection of an ontology given a query. SWOOGLE [Ding, et al., 2004] uses traditional Information Retrieval techniques to retrieve semantic web documents (SWD), specifically character based N-Grams, n-character segments of the text which spans inter-word boundaries, or URIs as keywords. The system indexes ontologies primarily designed with the OWL language which supplies by design a set of metadata which is extremely useful for identification of the SWD. Since the words are usually compounded into URI terms, this N-Gram approach is particularly efficient to index and retrieve the SWD.

Link analysis is used in [Patel, et al., 2003; Zhang, et al., 2004] to rank ontologies in respect to queries in the OntoSearch system and in the OntoKhoj system. Alani and Brewster in [Alani and Brewster, 2005] create the AKTIVERANK algorithm, aggregate a number of measures that look into the structural features of concepts such as concept similarity and structural density.

Necessary to the task of ontology selection is the subtask or concept identification, where the concepts in the query are identified in the ontologies. Resnik in [Resnik, 1995] uses information content, as defined in [Ross, 1976], to determine the semantic similarity of two concepts, the author restricts himself to the use of is-a relations to calculate the concept similarity. Jiang and Conrath [Jiang and Conrath, 1997] combines a lexical taxonomy with corpus statistical information to measure semantic similarity between words and concepts.

2.2. Ontology Mapping

Although work as been done in ontology integration such as [Reed and Lenat, 2002] and [Hovy, et al., 2003], where the goal is to incorporate several ontologies into one larger ontology, recently the focus seems to be in ontology mapping.

Stumme and Maedche in [Stumme and Maedche, 2001] based their work on the work of Ganter and Wille's [Ganter and Wille, 1997] work on formal concept analysis. Their method, the FCA-Merge is semi-automatic method for merging ontologies that uses natural language techniques to derive a lattice of concepts which is then explored by a knowledge engineer. The FCA-Merge assumes that a corpus relevant to both ontologies to be merged is available and relies on the availability of classified instances in those ontologies.

Using the Barwise-Seligman theory of information flow [Barwise and Seligman, 1997], Kalfoglou and Schorlemmer [Kalfoglou and Schorlemmer, 2002] created the IF-MAP method, a method for automatic ontology mapping. IF-Map generates a logic infomorphism given two ontologies. This method relies on a partial translation from the source ontologies to horn clauses, which is then used to discover the infomorphisms, if any. The result is stored for future reference.

Ontology mapping and alignment has been tackled by Noy and Musen through the creation of several tools that work as plug-ins for the open-source Protégé-2000 ontology editor [Grosso, et al., 1999]. The first tool was SMART [Noy and Musen, 1999], followed by PROMPT [Noy and Musen, 2000] and PROMPTDIFF [Noy and Musen, 2002] . The tools use linguistic similarity metrics for matching concepts. The authors claim that PROMPT not only uses linguistic similarity but also the similarities of the surrounding structures of the concepts to be merged. A set of heuristics is then applied to the performed the merging procedure. The PROMPT tool, as well as Chimaera [McGuinness, et al., 2000], provide semi-automatic guidance for the knowledge engineer. Similarly the SHOE system [Heflin, et al., 2003] provides with a set of heuristics

designed to align ontologies, offering the user a set of suggestions regarding ambiguous concepts.

Another approach is to use machine learning to develop a mapping between ontologies, examples of this kind of approach are given by Lacher and Groh [Lacher and Groh, 2001], with the CAIMAN system, Doan et. al. [Doan, et al., 2004], with the GLUE system, use a set of practical similarity measures to identify similar concepts. A Bayesian approach is used by Prasad et. al. [Prasad, et al., 2002] for deciding between similarity comparisons.

OntoMorph [Chalupsky, 2000] presents a method for translation of symbolic knowledge, integrated within the PowerLoom knowledge representation system [MacGregor, et al., 1997]. Using syntactic rewriting through pattern matching, the author claims that the potential of this translation system is adequate to handle complicated syntactic transformations. Semantic rewriting is applied to conflate large classes of concepts.

DRAGO [Serafini and Tamilin, 2005] uses the peer-to-peer paradigm with Distributed Description Logics to supply distributed reasoning services in multiple ontologies. Within what the authors call the contextual reasoning paradigm, the authors propose a distributed tableau algorithm to avoid the drawbacks of scalability and proprietary information and is able to provide with a distributed verifiability capability. Piazza [Halevy, et al., 2003] proposes a language based in XQuery [Boag, et al., 2002] that is used to describe semantic queries and that can be used with RDF style sources, although primarily developed for XML. OBSERVER [Mena, et al., 2000] uses interontology relationships such as synonyms, hyponyms and hypernyms to rewrite user queries to obtain translations across ontologies.

A more extensive survey on the subject can be found in [Kalfoglou and Schorlemmer, 2003] and in [Noy, 2004].

2.3. Ontology Evaluation

The increase in the number of available ontologies demands the question of ontology evaluation. Many approaches were taken on this issue. Two good reviews of different approaches are given in [Brank, et al., 2005] and [Hartmann, et al., 2004].

Porsel and Malaka [Porzel and Malaka, 2004] use a task based evaluation approach to evaluate ontologies. Given that an ontology will typically be used in some task, this provides a direct comparison of two or more ontologies for that task. The problem with this is that the range of tasks necessary to provide enough coverage for the typical applications in which ontologies are used is very large, making impractical to use as a generic evaluation metric. Yet given the diverse nature of the ontologies themselves, this seems to be the only feasible method at times.

The OntoMetric approach [Lozano-Tello, et al., 2004] establishes a set of processes a user should follow, given the specific system requirements, to evaluate different ontologies. The OntoMetric system uses a set of 160 characteristics based in features such as the content represented in the ontology, the ontology language, the methodology used to develop the ontology, the costs of using the ontology in the system and the software environment used.

Maedche and Staab [Maedche and Staab, 2002] use a gold standard to which they compare the ontology to evaluate. A data driven approach is used in [Brewster, et al., 2004], where the authors use latent semantic analysis to compare a set of concepts from a domain specific corpus and a set of concepts in the ontology, determining a fit between the two sets.

3. Methodology

Federated Search identifies four key areas of research for a problem solution. We will show that the same problems apply in the area of Federated Ontology Search:

- Resource Selection
- Query Execution
- Result Merging
- Result Ranking

Before we consider each of these topics, we must establish some basic definitions

3.1. Basic Definitions

3.1.1. Ontologies and Graphs

Although there is no consensual definition of *ontology*, a good start comes from G. Stumme and A. Maedche (Stumme and Maedche, 2001). The authors claim that most ontologies share a few common items such as

- Concepts, a hierarchical IS-A relation and further relations.
- Some ontologies have constraints, functions or axioms

For the purposes of our research, an ontology can be as simple as a semantic network [Quillian, 1967], where no distinction is made between concepts and instances, and the only relation possible is of the *is-a* type, or as complex as CYC [Lenat, 1995], with a clear distinction between concepts and instances, where multiple inheritance is allowed and there is an extremely reach set of possible relations.

A basic ontology definition could given by a tuple $O := (C; is\ a; R)$, where C is a set whose elements are called concepts, *is a* establishes a partial order on C and R is a set whose elements are called relation names. An example is given below.

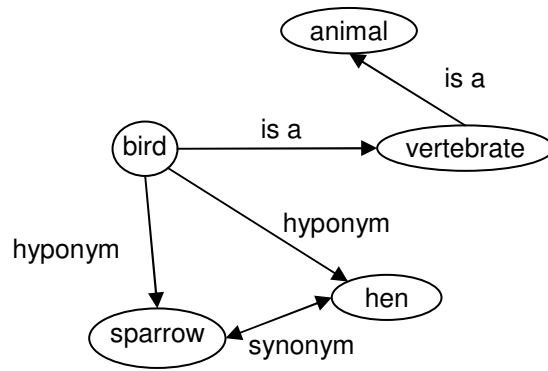


Figure 1 - an example of a basic ontology

A graph definition could be given by $G = (V,E)$ where V is the set of vertices and E is the set of edges. An example is given below.

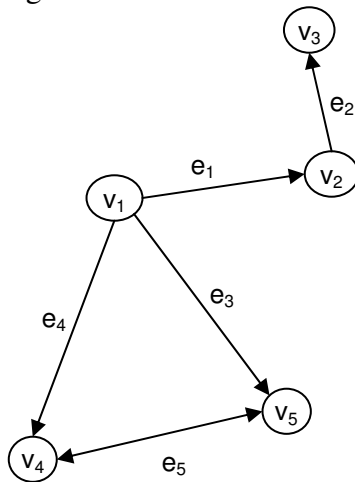


Figure 2 - A graph structure

Given the two definitions one can see that graphs fit the basic structure of ontologies very well. Vertices are considered concepts, Labeled edges as relations.

3.1.2. Query

A query is a request for information from the set of existing ontologies. It is comprised of operators, as defined in section 3.2.1.

3.1.3. Result

A Result is the rooted directed acyclic graph (RDAG) that results from executing a query.

3.2. Ontological Search

The success of the proposed approach hinges on the definition of a search method that is independent of any ontology. For this purpose we introduce the concept of *operator* and a concept of query based on operators. The main purpose of an operator is to decouple the search process from the information need. Instead of describing a complete semantic framework, the goal is to describe the information request in terms of a decomposable query that can be transformed into a set of operators. This would provide an elegant abstraction from the formal representations implemented by our ontological sources, allowing each operator to be an independent request.

As an example, Figure 3 shows the execution of the query *children(car)*. This query uses the *children* operator to get all the children of the concept *car*.

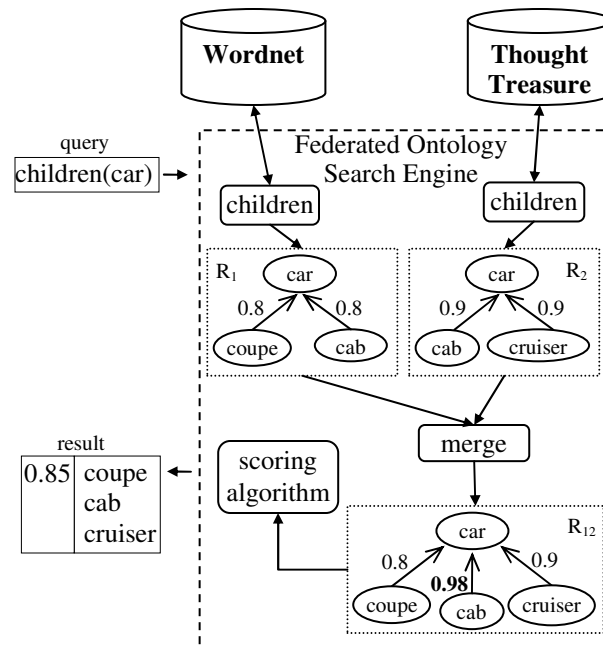


Figure 3 - Query Execution

The query is interpreted and performed in each ontology separately. The results are then merged and a final list is ranked according to the scoring algorithm. The final set of results is then returned.

It is important to note that by defining a set of operators we are in fact delegating responsibility for their execution to the ontologies themselves, therefore making no restrictions on whatever processes are executed in order to obtain the necessary information. This means that operators can be implemented using extended features of ontologies (e.g. inference, grounding, restrictions and theorem-provers). The only constraint is that the output of each query execution is a Rooted Directed Acyclic Graph (RDAG).

Next I will give a description of a few possible basic operators as an example. These operators form the base for my experiments, but not a final set of operators, since defining a set of operators is one of the goals of this thesis. Following that I will describe the concept of query within this context and finally I will address the problems of resource selection, merging and results scoring in detail.

3.2.1. Operators

An atomic operator is an atomic search operation on an ontology. It takes as input a graph and produces a ranked list of graphs as output. An operator is defined as an operation on

$$\text{op}(g): g \rightarrow g', \text{ where } g, g' \text{ is a RDAG}$$

We now define a set preliminary set of operators used in the current evaluations. By no means does this constitute the base set of operators, necessary to represent the basic ontology operations. That is subject of further research.

3.2.1.1. *children operator*

The children operator takes a graph g as input and expands each concept in g to the set of children concepts. Currently defined as

3.2.1.1.1. Base Case

In base case let's look at an example where the set of vertices of g , $V(g) = 1$, that is, the graph is comprised of one concept.

Query : $\#children(g)$

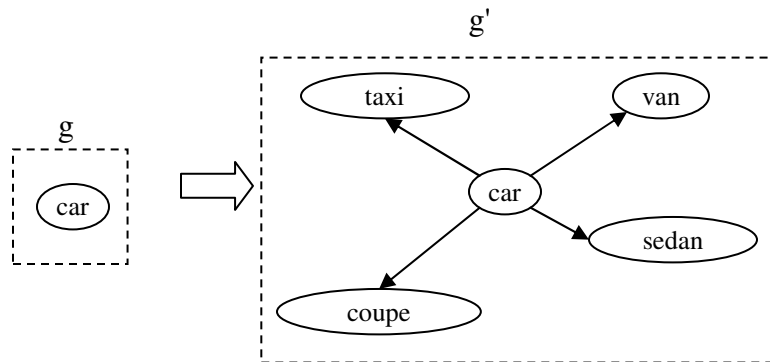


Figure 4 – Base case for children operator

As we can see the base case expands one concept into a set of concepts, but what happens when we apply this operator to a set of concepts?

3.2.1.1.2. Complex Case

In this case each concept in g is extended to the set of similar concepts

Query : $\#children(g)$

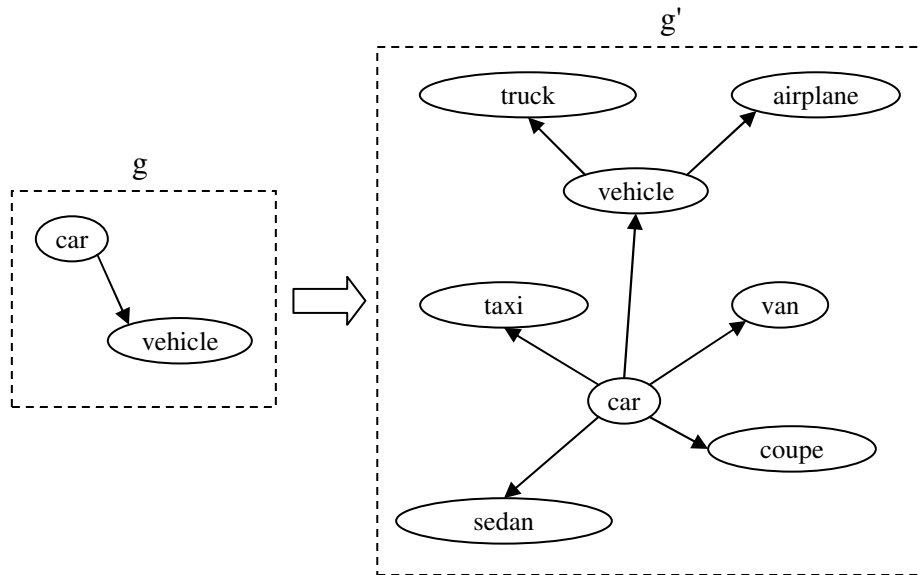


Figure 5 – Complex case for children operator

3.2.1.2. *relation operator*

$\#rel(g_1, g_2, [r_1, r_2 \dots r_n])$

The relation operator takes g and for each of its concepts tries to find the relationship to each of the concepts in g' using the relations r_1 to r_n . If no relations are specified, then all relations are considered.

Example : $\#rel(car, vehicle, [is_a])$

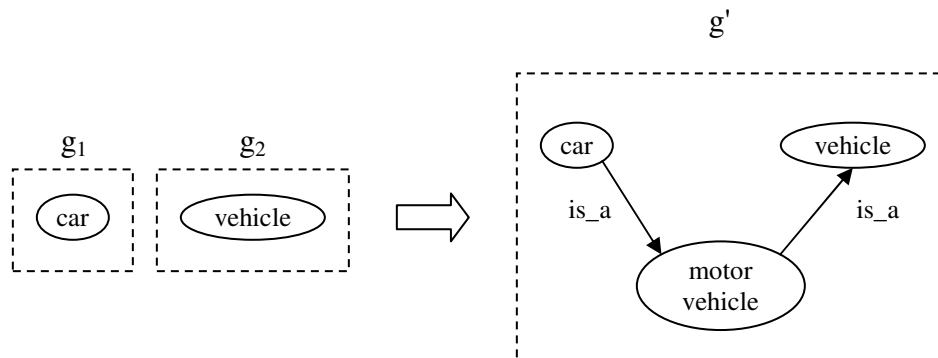


Figure 6 – relation operator

3.2.1.3. *and operator*

$\#and(g_1, g_2)$

The *and* operator represents the intersection operation. It takes g_1 and g_2 and return g' where g' is the intersection of the two.

Example

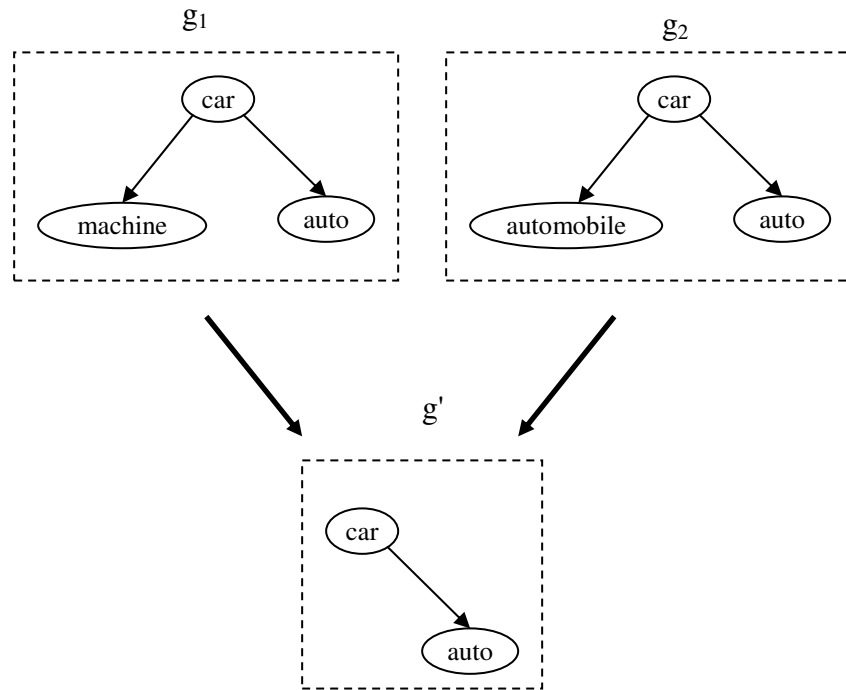


Figure 7 – and operator

3.2.1.4. *or operator*

$\#or(g_1, g_2)$

The *or* operator represents the intersection operation. It takes g_1 and g_2 and return g' where g' is the union of the g_1 and g_2 . This operator does not apply boosting. Given two similar edges one of the edge is picked arbitrarily.

Example

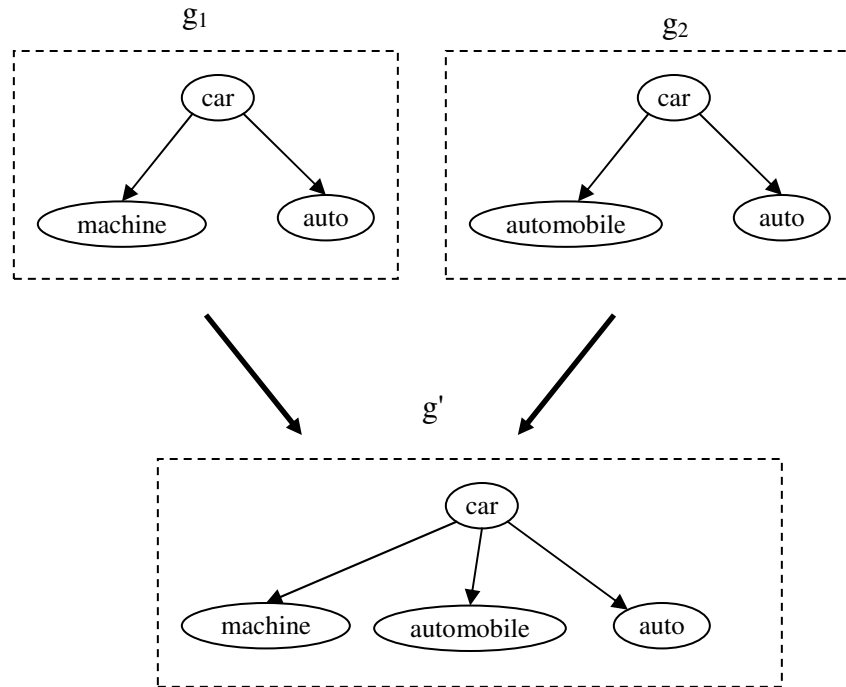


Figure 8 – or operator

3.2.2. Query

The general execution of a query is described as follows. First we select the appropriate ontologies. Second, we query the selected resources. Third we merge the results according to a graph merging algorithm. Finally we rank the results according to a confidence estimation algorithm.

A query operation is composed of atomic operators and Boolean operators. Each query is reduced to a linear sequence of atomic operators.

$\#rel(\#sim(lymphoma),cancer)$

Query 1 : $\#sim(lymphoma) \rightarrow g$

Query 2 : $\#rel(g,cancer) \rightarrow g_1$

Each of the atomic operators is considered an atomic query and performed on the selected resources

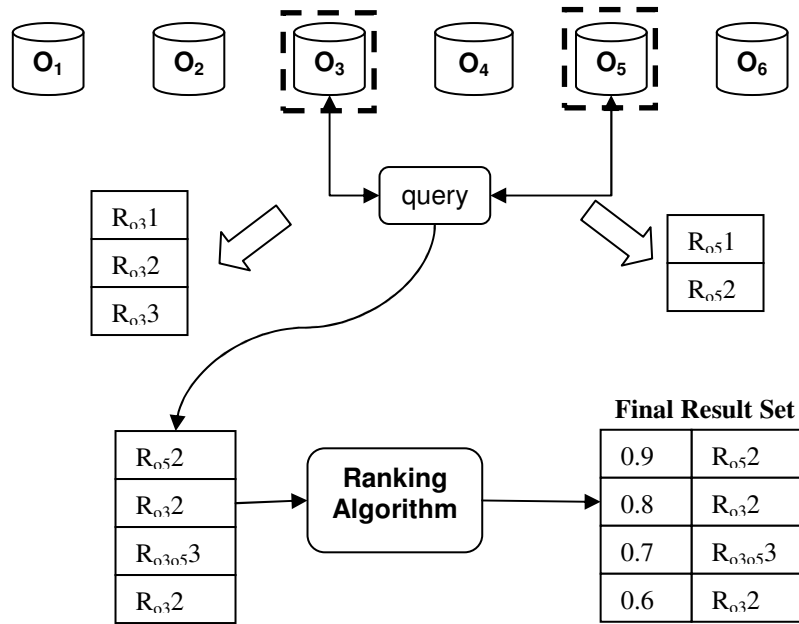
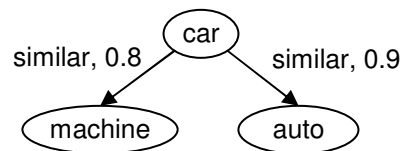


Figure 9 – Diagram showing the execution steps of an operator

3.2.2.1. Query Results

A query result is a set of Rooted Directed Acyclic Graphs (RGAD). The graph contains labeled edges and attributes are modeled as relations. Each edge contains a confidence associated with it. This confidences expresses the confidence of the source in the relation.

Example



3.2.3.Resource Description and Selection

3.2.3.1. *Resource Description*

The increasing trend in the availability of resources suggests that often we will be able to find overlapping sources for a given query. At the same time, some resources will be very domain oriented which brings up the problem of resource description and selection.

Ideally we would like to be able to select the resources to query in order to maximize the probability of success. We would like to model the success of a query given an ontology. We must consider two cases, similar to the situation in information retrieval, cooperative and uncooperative resources [Si and Callan, 2005].

A cooperative source is a source whose knowledge is fully available for querying and indexing. In many cases though, it's not realistic and maybe not even desirable to expect cooperative sources. The proprietary content in some ontologies might not be made available by its authors, or perhaps part of the ontology might be available, with filters to control access to the information contained in such ontology. Examples of this can be taken from Cyc, which releases OpenCyc as a free smaller portion of the knowledge contained in the full Cyc. Although at this moment they are separate entities, one could conceive of a controlled access paradigm. We must also consider cases where the ontology has incorporated inference engines and logic mechanisms, the use of which is advantageous and important. Therefore, it is in our interest to evaluate the contents in terms of the produced results, rather than the information contained in the ontology.

For an example of the importance of this resource description, we can look to our example in 1.2.2., where selecting the wrong resource would lead to wrong results, Where without an adequate resource description, it will be extremely hard to differentiate between expert and non-expert sources.

One possible approach to resource description is to formulate a set of random queries that determine the content of the ontology to be queried. One obvious challenge in this

approach is the creation of such queries. Given the structured nature of ontological resources, automatic extraction of queries from a test corpus becomes increasingly hard. Another possible approach is to apply proximity models, where each result is judged regarding how close is to the other results. A commonsense result is built by taking the average of the results returned and the difference is measured for each result. This will not tell us the expertise of the ontology, but will give us a sense of how disparate each ontology is from the norm. This would provide a relative measure of ontology description.

3.2.3.2. *Resource Selection*

Resource selection refers to the selection of the most appropriate set of ontologies for a given query. The key issues in this task are the identification of concepts in ontologies and the matching of an ontology description with a query.

The problem of identifying concepts in ontologies is particularly vital given that although we typically have strings representing the concepts in the query these are usually not enough to eliminate ambiguity. Furthermore, the concepts in ontologies may be polymorphic, having multiple literal representations. This problem increases when we consider the use of multilingual ontologies, where the concepts are represented in several totally different languages. The use of ontologies in different languages would lead to necessity of a distance metric specifically tuned to multilingual issues.

3.2.4. Merging

One advantage of this approach is that rather than trying to merge two entire ontologies, we merge only the results. This significantly reduces the problem of merging ambiguous concepts given that queries are grounded in a query concept and therefore only relations that apply to the concept in the query are returned. Polysemic concepts will be explicitly modeled via separate results, with a variable confidence on the relations contained in each result.

The structured nature of the query results implies that the result merging problem in the ontology domain is very different from result merging in federated text search, where the results typically consist of unstructured data. In the case of Ontology Search the results are graphs, which allow us to treat merging as an instance of Inexact Graph Merging. While one of the goals of merging in Federated Search is to eliminate duplicates, the primary goal of merging in ontology search is to find complementary information. The goal of ranking in ontology search is to produce, at the top-ranked position, the most complete and accurate result. Generally speaking, we want to merge two results if they represent information about the same concepts, thus creating a more complete result.

As an example, let us consider three distinct ontologies, O_1 , O_2 and O_3 . Let us assume that we execute the query $sim(bank)$, with the purposes of finding concepts similar to bank. Imagine that ontology O_1 and O_2 both interpret *bank* as *river_bank* while ontology O_3 interprets *bank* as *money_bank*. Given this scenario, we would like the result set to contain two results, one referring to the similar concepts of *river_bank* and the other to the similar concepts of *money_bank*. The results from O_1 and O_2 should be merged since that is likely to yield an increase in the quantity of information contained in the result, as well as the confidence in the concepts common to the two results.

The problem can be formulated as follows. Given two RDAGs g_1 and g_2 we want to merge the two graphs if they are similar. Typically this is done by considering one of two options, either we measure the similarity between the graphs or measure the difference. But in our case we not only want to measure the similarity but also find the maximum common subgraph. We should look to the science of Inexact Graph Matching for guidance here.

Inexact Graph Matching occurs when we do not expect to find an isomorphism between the two graphs to be merged. This is one of the most complex problems in computer vision [Bienenstock & Malsburg, 1987] and is also an important part of chemical similarity searching [Raymond et al., 2002]. More specifically, inexact graph matching is proven to be an NP-Complete problem [Abdulkader, 1998].

In the next section we will discuss graph similarity.

3.2.4.1. Graph Similarity

Graph similarity Distance [A. Sanfeliu & K. Fu, 1981] is typically calculated in one of the following ways: Cost Based Distance, Feature Based Distance or Maximum Common Subgraph.

Cost Based Distance is based on edit operations on the graph, typically add nodes or edges, remove nodes or edges and re-label nodes or edges, where each operation is associated with a cost. Given two graphs g_1 and g_2 , the edit distance between g_1 and g_2 is the minimum number of edit operations necessary to transform g_1 into g_2 .

Feature based distances use a set of invariants established from the graph structural description, using these features in a vector representation to which we then apply distance or similarity measures.

The goal of the Maximum Common Subgraph approach is to find the largest Subgraph common to both g_1 and g_2 . To address this requirement, current approaches use the concept of maximum clique detection, or the concept of maximally connected sub graphs. Given the NP complete nature of the problem, the problem is then changed into finding the Maximum Common Edge Subgraph, which focuses on finding graphs with the maximum number of edges. In our case we use a variation of the overlapping coefficient for graphs, a measure whereby if graph g contains g' or the converse then the similarity coefficient is a full match.

3.2.4.1.1. Localized Boosting Algorithm

As stated before, Inexact Graph Matching is an NP complete problem. In order to tackle this problem, we take advantage of the fact that our graphs are RDAG's to reduce the complexity of the problem. The goal here is to create a set of tuples that will be the basis for comparison of the two graphs.

Given g_1 and g_2 as results of a query, the algorithm is as follows. After applying a screening procedure to determine the upper bound on similarity, as defined in [Raymond, et al., 2002], we are left with graphs where $sim(g_1, g_2) > T$, that is, graphs where the similarity between g_1 and g_2 is above a certain threshold T , defined in the screening procedure. The screening procedure produces a subgraph that for each graph given, which means that we now basically want to determine $g_1 \cap g_2$ for which we will apply localized boosting and then add the nodes and edges that were previously discarded.

The basic intuition behind the confidence boosting is that the confidence of the edges is boosted whenever two edges are merged. The boosting is determined through the use of the *Soft Or*, given by the formula:

$$1 - \prod_i (1 - c_i)$$

The motivation of using *Soft Or* to determine the boosting is that this gives us a smooth boosting curve with an upper bound of 1.

E.g. $A = 0.8, B = 0.7, \text{Result} = 1 - (1 - 0.8)(1 - 0.7) = 1 - (0.2 \times 0.3) = 0.94$

In order to apply confidence boosting we apply the concept of tuples, where $t_x = (c_x, c_y, r)$ is a tuple, c_x, c_y are concepts and r is a relation.

First we split g_1 and g_2 into tuples $t_x = (c_x, c_y, r), c_x, c_y, r \in g$, such that c_x and c_y are adjacent and $r(c_x, c_y)$. We then compare the sets of tuples from g_1 and g_2 and if $sim(t_x, t_y) > T$ then we boost the confidence of t_x .

.

An example is given in Figure 10

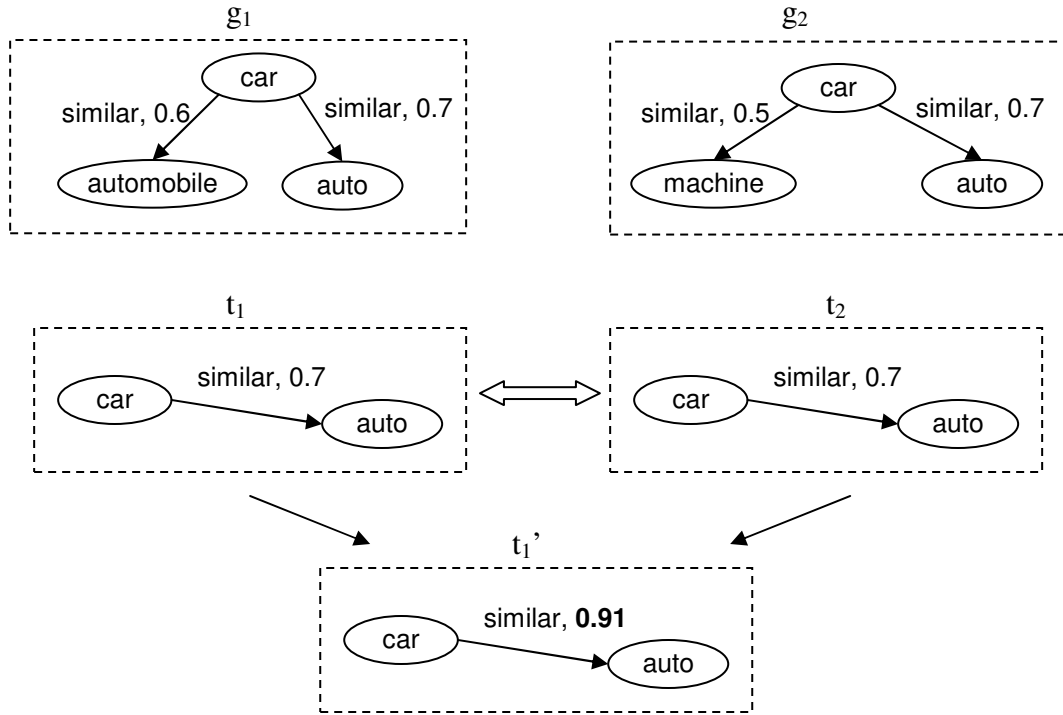


Figure 10 – Localized Boosting Algorithm

3.2.4.1.2. Tuple Similarity

Tuple similarity measures are based on the linear combination of the edge similarity measure and the concept similarity measure.

When comparing concepts or relations, we use the Q-Gram distance on the strings that represent them [Gravano et al., 2001]. A q-gram is character based N-Gram measure. The intuition behind the use of q-grams as a foundation for distance metric is that when two strings s_1 and s_2 are within a small edit distance of each other, they share a large number of q-grams in common. This metric is fairly robust to orthographic errors, morphological errors and compound words, which makes it suitable for our purposes.

The similarity between two tuples is given by the minimum similarity of the concepts and relations contained in the tuples. Formally

$$sim(t_x, t_y) = \min \begin{cases} sim(c_{x1}, c_{y1}) \\ sim(c_{x2}, c_{y2}) \\ sim(r_x, r_y) \end{cases}$$

3.2.5. Scoring Results

A result is scored in a compositional manner, by scoring the outcome of each operator used in a query individually before calculating the final score. A ranking will be computed from the scores of the results thus making the computation of the ranking score a key issue, much like in traditional Information Retrieval.

An operator can be either recall centric or precision centric. Operators that focus on recall will typically return results with as much information as possible. The *similarity* operator, for example, returns all the synonyms associated with a concept, the more synonyms the better the result should be, all else being equal. Operators that focus on precision will usually return chains of associations. They focus on precision of the relations. As an example, the *relation* operator finds the relation between two concepts. All else being equal, a direct relation would be better than a long chain of relations.

When a result is merged from two other results, the confidence in the sources from which the results were extracted is combined using the *soft or* rule, as described before. We now present two scoring metrics, to be used by precision type operators and recall type operators respectively.

3.2.5.1. Precision scoring metric

The goal of this scoring metric is to give preference to shorter graph lengths. Given that this metric is used for precision type operators, it is desirable to have results with short chains.

$$S(r) = C_s \frac{\prod_i (C_{ei})}{((avg_length)^2)^2}$$

Where c_s is the confidence of the source, c_e is the confidence on the edge and avg_length is the average distance of the paths contained in the graph, from root to leaf node. The 4th power was empirically determined to give an adequate curve to the confidence decrease.

3.2.5.2. Recall scoring metric

The recall scoring metric gives preference to graphs with large node degrees. Given that this metric is used by recall type operators, we want the score to increase with the quantity of diverse information contained in the graph. Thus

$$S(r) = C_s \times \left(1 - \frac{1}{((avg_degree)^2)^2} \right) \times \prod_i (C_{ei})$$

Where avg_degree is the average degree of the nodes contained in the result.

3.3. Known Issues with current work

The research work in this thesis is driven by a set of problems that are yet to be solved satisfactorily. Although this is not a complete enumeration of such problems, it represents the set of prominent problems at this stage. These constitute the focal points of the proposed research and will be addressed by it.

3.3.1. Identification of the correct concepts and relations in ontologies

String based comparison, currently the method used, is not adequate to deal with the phenomenon of polymorphic concepts, abundantly present in many ontologies as well as polisemic strings. The necessity of limiting ambiguity and identifying the correct concepts and relations requires an approach that takes into account more than just unique tokens, indicating that some notion of context must be present.

3.3.2. Concept and edge similarity between concepts in different ontologies

The identification of concept and edge similarity, currently done using string similarity metrics, is insensitive to the polymorphic problem, as mentioned before. Furthermore the issue of ontologies in different languages exacerbates the problem considerably. Intuition suggests that viable approaches should include topologic similarity as well as string similarity. This amounts to comparing contexts to define concept and edge similarity.

3.3.3. Chains of indirect inference

A chain of indirect inference is a chain in which parts come from different ontologies, that is, an inference chain that goes from A to B, where A exists one ontology and B exists in a different ontology. Chains of indirect inference reduce the reliability in the results. One of the factors seems to be the increase in error when unifying concepts in two different ontologies. In order to take advantage of multiple ontologies for indirect knowledge the unification of concepts must be restricted to highly compatible concepts.

3.3.4. Insensitivity in Boosting graphs of different structural properties

When two results inherently represent the same information but their structure differs, the current boosting algorithm tends to incorrectly assign the boost in confidence in equal amounts to all partial paths. Research in identifying the correct partial paths to apply boosting is a challenging problem to be addressed in this thesis.

4. Preliminary Results

In this section we present some retrieval experiments using the federated approach to ontologies. One possible evaluation of the proposed approach requires a task centered evaluation process (Porzel and Malaka, 2004). Unfortunately at this time there are no existing standard procedures and test sets for ontologies, which forced us to create our own. For this reason we are unable to provide results comparing our system with other similar systems. Similar to the experiment performed by the mentioned authors, we selected the task of type checking, described below.

4.1.1. Type Checking

The task of type checking tries to determine if, given a type T and a concept C , C is of type T . In the case of the federated approach, we can achieve this by using three operators, the *relation* operator, the *parents* operator and the *children* operator, as previously described.

Type checking using federated ontology search can be viewed as the task of finding an *is-a* based path between two concepts. Our approach has the advantage of using indirect paths when no direct path is found. An indirect path consists of partial ordered sub paths that exist in separate ontologies but form one path when combined. Finding an indirect path is possible by simply applying either the *parents* operator or the *children* operator to the source node in one ontology and using the resulting nodes to query for a direct path in another ontology. The resulting path is the combination of these partial paths. Using indirect paths provides a promising way of combining information that by itself would be incomplete and enabling the deduction of previously non-existent paths.

4.1.2. Experimental Setup

A total of 9558 pairs were extracted from results of the Javelin question answering system in TREC QA 2003 [Nyberg et al., 2003]. Each pair consists of the expected answer type or subtype and the candidate answer.

For the purposes of our evaluation we used two of the currently available ontologies, Wordnet and ThoughtTreasure. The purpose of this preliminary evaluation is to contrast the performance of each of the ontologies individually, which would be a typical scenario for a project using one ontology as a knowledge base, with the performance of the set of ontologies using a federated approach.

We have evaluated the recall and precision of the retrieved results..

4.1.3. Results and Analysis

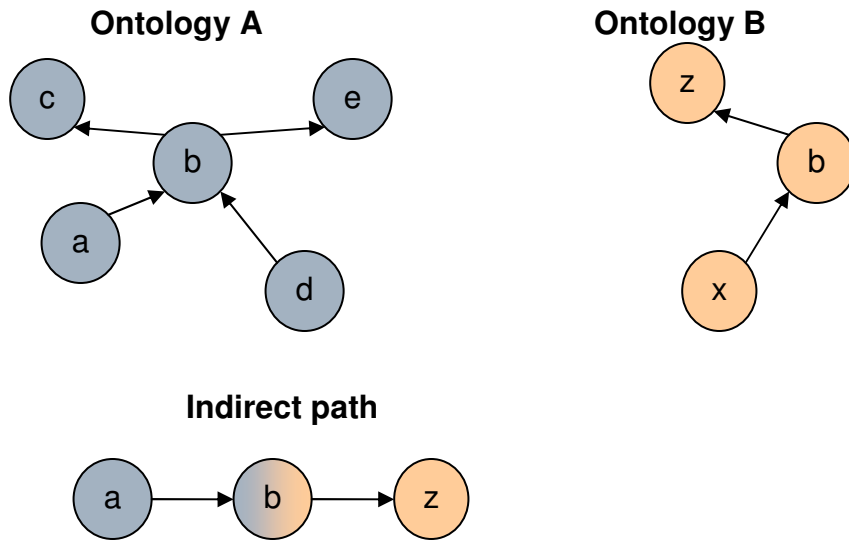
Table 1 shows the recall after running the test set with different configurations.

Configuration	Recall
Wordnet	4278 (44.7%)
ThoughtTreasure	730 (7.6%)
Combined	4686 (49%)
Merge	4686 (49%)
Merging + Indirect	6870 (71.8%)
Test size	9558

Table 1: Recall using different configurations with the full set of pairs

Wordnet and *ThoughtTreasure* were experiments where Wordnet and ThoughtTreasure were used individually. The *Combined* experiment queried each of the ontologies individually, picking only the top ranked result. The recall is lower than the direct sum of the individual results due to knowledge overlap in the ontologies. The *Merge* experiment queries both ontologies but merges the results using the merging algorithm described previously. Finally we use merging as well as indirect path query to perform the last experiment

An indirect path is a path that is comprised of partial paths contained in different ontologies, as shown below.



Although the recall remained the same when applying the merging procedure, the average confidence of the top result, in cases where there was more than one result, increased significantly (28%), as shown in Table 2.

	avg. confidence
Without merging	0.72
With merging	0.93
increase	28.7%

Table 2 – The Increase in the average confidence of the top ranked result due to the merging algorithm.

In order to test the accuracy of the federated approach, we created a gold standard for a subset of the full set of pairs. Using random sampling, we selected 1300 pairs, which we then proceeded to judge manually. For each pair in the gold standard subset we generated a tuple of the form $(type, concept, judgment)$, where judgment reflects if the *concept* is of the type *type*.

We compared the answers of the Federated Search with the gold standard by applying a variation on the result score threshold. If a score is below the threshold then the *concept* is considered not to be of the type *type*.

	Precision	Recall	F1 Measure
Combined (W+T)	0.59	0.49	0.53
FOS (M+I)	0.67	0.71	0.69
Increase			30.18%

Table 3 – Precision and recall of the Federated System using Wordnet and ThoughtTreasure

We obtained a significant increase in performance when using the federated search approach. The optimal threshold for this experiment is $T=0.1$ with a precision of $P = 0.676$. The recall was very close to the one obtained in the full set with a recall of 0.71 (71%). Below we can see the F-Measure of the system.

5. Roadmap

5.1. Thesis Scope

Research Topic	Research Question	Contributions to the field	Possible Answer
Query Description			
Operator Set	What is the set of basic operations in an ontology	General operator set that described the set of basic ontological operations	Creation of a basic operator set through literature review
Resource Description			
Ontology Description	What are the areas of expertise of a given ontology	Set of features for resource description	Creation of set of queries to determine areas of expertise
Resource Selection			
Concept Matching	How to match concepts in the query, described by strings to concepts in the ontology	Methodology for selecting matching concepts in ontologies given a query description	Definition of context for ontology querying
Fitness of an ontology given a query	How to determine how good is an ontology given a certain query	Algorithm for determining ontology fitness given a query	Usage of the determined areas and with conjunction with the context of the query
Result Merging			
Result Similarity	How to determine the similarity between two results	Algorithm for determining result similarity and for merging results	Use the combination of concept and edge similarity with
- Edge similarity	How to determine the similarity between two edges		Use topological similarity, string similarity and property coherence (monotonic, reciprocal, etc..) for
- Concept Similarity	How to determine the similarity		

	between two concepts		determination of similarities
Result Boosting	What is the correct way to boost the results?		Use spread boost taking into account the properties of the ontology itself
Result Scoring			
Adequacy of Result	How well does the result answer the query (quantity of new information, precision of new information, etc)	Algorithm for Scoring results.	Using Utility-based metrics to boost the score of the results
Width vs. Breadth	Is it possible to describe the results in terms of quantity and quality of information?		

5.2. Research Activities

This research is comprised mostly of the four major tasks described in 5.1. We will describe the activities in each task. The conclusion of these set of activities will lead to the creation of a framework for integration and use of ontologies for NLP applications.

5.2.1. Graph Matching

Graph Matching is one of the key components of the proposed research. It heavily influences both term matching and graph merging thus consisting of a fundamental focus of this research. Besides other aspects covered throughout this proposal, one important aspect is the definition of graph matching at three separate levels. At the structural level, string level and meta level.

5.2.1.1. *Structural Level Matching*

Matching at the structural level involves analyzing and comparing the structure of both graphs to be matched. This requires comparing branching factors, connectivity structure of the graphs, edge number and position. This can be done by extrapolating existing matching algorithms fine tuned to the purpose at hand.

5.2.1.2. String Level Matching

String level matching involves comparing information contained within nodes or relations. This requires knowledge of the type of content we are comparing, given that some type of content requires exact matching (e.g. dates, distances) while other requires interval matching (E.g. weight of a whale can be from 5 to 50 tons). Besides string matching using algorithms such as the Q-Grams algorithm, we can use structural matching combined with string matching to identify synonyms.

5.2.1.3. Meta Level Matching

Meta Level matching refers to properties of the graph and its relations that are not structural or string based. Properties like relation exclusivity (E.g. we can only have one mother but we can have many siblings) force certain graphs to be incompatible due to meta knowledge on the information contained in those graphs. We must not only match the direct knowledge contained in the graphs, but also the inferred knowledge that can be extracted from them.

5.2.2. Ontological Metadata

Given the diversity of ontological languages and the capabilities of expression that each language possesses, we must encode ontological metadata information. The purpose of this information is to help model the query performed in that ontology, as well as the results returned. Information like which relations are available, what kinds of logical structures are present and what inference mechanisms are active is necessary to correctly query ontologies and process their results.

Some of the information that is required from the relations present in ontologies includes information about whether a relation is transitive, unique, reflexive and associative, amongst others. This suggests that some sort of relation algebra is required to effectively describe relation interplay accurately.

5.2.3. Query and Resource Description

5.2.3.1. *Creation of an operator set*

Creation of an operator set that corresponds to the basic set of operations usually performed in an ontology. Although the set of operators depends on the evaluation task, it is possible to decompose the typical ontological operations in a set of basic operators, using the current literature as a basis.

5.2.3.2. *Development of a set of ontology queries for Ontology Description*

In order to describe an ontology, a set of queries suitable to querying different types of ontologies must be created. This set of queries should come from various corpus, namely the CNS corpus, the AQUAINT corpus, amongst others. A methodology should be defined such that the extension of this query set becomes an easy task, possibly automatic, given the degree of variability of coverage in ontological resources.

5.2.3.3. *Extension of the operator set*

The current operator set will be extended to include a subset of the operators supported by SIRUP [Ziegler, et al., 2006]. Possible operators include

Operator	Description
getConcept(name)	Given a name, return all the concepts associated with it
getRelationSet(concept)	Retrieve the relations associated with a given concept
getAttributes(concept)	Retrieve the attributes of a given concept
fillGaps(graph g)	g is a graph with empty slots. Return the graph structures that fill those empty slots
getInstances(concept)	Retrieve the instances of a given concept
isConcept(conceptName)	returns true is the name corresponds to at least one concept
isAttribute(attributeName)	Returns true in the attribute name corresponds to at least one attribute
isSimilar(conceptA, conceptB)	Returns true if concept A and concept B

	are similar
getRelatedConcept(conceptA,[relationSet])	Returns the concepts related to conceptA using relations contained in the relationSet

5.2.3.4. Evaluation of query set for Ontology Description

This task consists of the evaluation of the results of the created query set. The evaluation type is described further ahead.

5.2.4.Resource Selection

5.2.4.1. Definition of a query context

This task consists of the definition of what is a query context for an ontological query. This should take into account the structured nature of ontologies such that the context represents not only the set of concepts associated with the query term, but also the relations that exist between the query term and the concepts in the context.

5.2.4.2. Matching the query terms

I propose to address two problems with my research work, synonym terms and homonym terms.

5.2.4.2.1. Synonym Terms

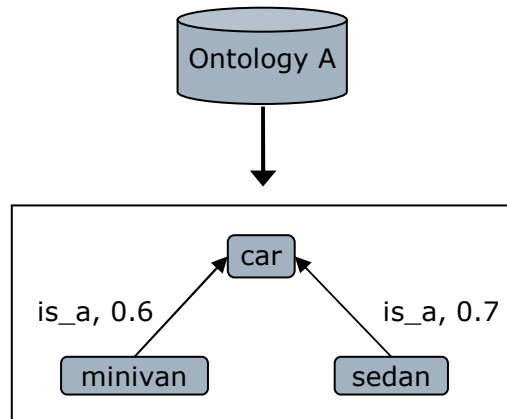
Synonyms terms present a significant problem in term matching. Terms with very similar meaning are frequently represented by different strings, a situation which leads to an unsolved matching problem. We intend to leverage on the fact that we are using a federated approach to address this problem.

The main idea is to use the graph structures of the returned results to query other ontologies and find synonyms. By exploring the structure surrounding terms whose

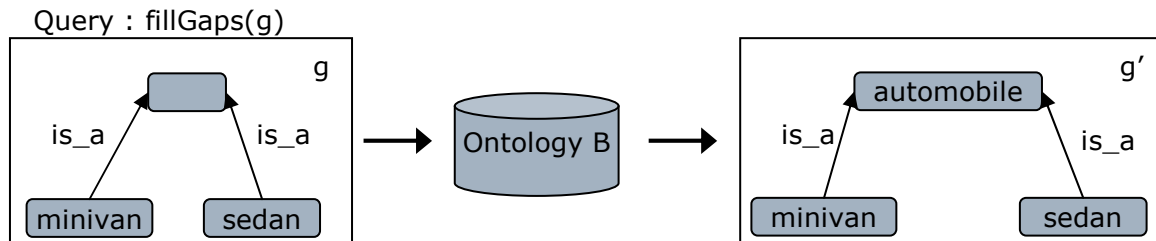
strings can be matched we can identify terms that have different strings but relate with the same surrounding term in the same way. An example is given below.

Let 's consider the following query children(car). Our goal is to match car with both car and automobile.

Query : children(car)



Given the graph that results from querying ontology A, we can use the structure of that result to query ontology B.



5.2.4.2.2. Homonym Terms

Given a query, this task is comprised of using the context defined in 5.2.2.1 to disambiguate between homonym terms. Current possibilities consist of the inclusion in the query of related words or sentences where the terms occur.

5.2.4.3. **Creation of an algorithm for determining ontology fit**

Given a query this task is comprised of the creation of an algorithm for the determining the probability of success of querying an ontology given a query, or $p(O,Q)$.

5.2.4.4. Evaluation of Resource Selection

This task consists of an evaluation of the algorithms for Resource Selection.

5.2.5. Result Merging

5.2.5.1. Experimentation with topological similarity, property coherence and string similarity

This task consists of experiments using topological similarity, string similarity and property coherence for determining the similarity of results.

5.2.5.2. Refinement of current boosting algorithm

The current boosting algorithm is not sensitive to structural differences in similar results; this task consists of refining the current algorithm so that it is more robust and sensitive to this situation.

5.2.5.3. Evaluation of Similarity measures and boosting algorithm

This consists of evaluating the similarity measures and boosting algorithm created in the previous points.

5.2.6. Result Scoring

5.2.6.1. Definition of utility metric for determining the appropriateness of the result given the query

This task consists of the creation of an utility metric that takes into account the information need in the query to determine the goodness of the results in order to influence the ranking.

5.2.6.2. Evaluation of the scoring metric

This task consists of the evaluation of the created scoring metric.

5.3. Evaluation

Different methods have been used to evaluate single ontology systems [Hartmann, et al., 2004; Porzel and Malaka, 2004]. These methods address different ontological aspects, namely coherence, coverage regarding a corpus, redundancy, amongst others. For our case, given that we are not trying to evaluate the ontologies themselves, but their combined use, we believe that a task based evaluation is the most suitable form of evaluation. This means the use of the existing type checking task, but also the creation of a different task for evaluation of the Federated Ontology System. We propose to perform a task based on the TREC QA set of factoid questions as well as at least one other task.

5.3.1. Factoid TREC QA Task

TREC QA set of factoid questions for evaluation of the proposed approach such as suggested in [Lita, et al., 2004]. The reasons for this are the following

- The TREC QA questions have available answers and the methods to judge are automatic and straightforward.
- The relations displayed in the TREC QA factoid questions represent a range of relations whose coverage is wide enough to test large ontologies such as Wordnet and ThoughtTreasure.

The Evaluation will address the precision and recall of the system, as well as key aspects such as the impact that the different methods of merging and scoring have in the results of the system. This will allow us to understand the factors at play in the federated system.

The goal of the evaluation is to take us beyond factoid question answering, since there in lays the challenge to and greater benefit from the use of multiple ontologies.

The evaluation will consist of the following steps:

- 1) Selection of the set of ontologies to use in the evaluation. The set should be comprised of ontologies in the number between 6 and 12.
 - a) Current ontologies considered
 - 1 WordNet

- 2 Cyc Research
 - 3 ThoughtTreasure
 - 4 CNS ontology
 - 5 Scone Ontology
 - 6 SUMO ontology
 - 7 Wikipedia derived ontology
 - 8 OpenMind Ontology
 - 9 MindNet Ontology
 - 10 Omega Ontology
 - 11 MultiNet Ontology
- 2) Creation of the set of necessary operators
 - 3) A/B comparison of type-check task using the researched methods
 - 4) Creation of a test set using TREC QA 2005 factoid questions. This includes queries and answer sets.
 - 5) A/B comparison of factoid QA task using the refined methods
 - a) Comparison of each individual ontology and the federated system
 - b) Comparison of the federated system with or without the merging algorithm
 - c) Comparison of the federated system

5.3.2. Possible evaluation tasks

5.3.2.1. *Tagging Ontological Relations*

This task is similar to the task described in [Porzel and Malaka, 2004]. The goal is to correctly tag the ontological relations that hold between ontologically equivalent marked-up entities. This is similar to the task of identifying correct frames for labeled verbs and arguments, presented in [Gildea and Jurafsky, 2002]. Some of the relevant ontologies relevant to this task are Framenet, VerbNet and Propbank.

5.3.2.2. *Image Labeling*

Given the recent advances in Human Computation [von Ahn and Dabbish, 2004], it is likely that we will now have millions of labeled images. It would be interesting to further label these images with ontological relations in order to increase the relational density of the images and the quantity of labels. This way, we would be able to query for animals and get images of dogs, cats, etc.

In this case the purpose would to leverage on the large number of ontologies that exist in ontology repositories and, although loosing some language diversity, benefit from an experiment that is both orthogonal to the factoid QA and uses a significantly larger number of ontologies.

5.3.2.3. *Ontology Clustering*

The goal of Ontology clustering is to cluster ontologies based on topics. This would allow, for example, determining commonsense regarding a concept using a set of ontologies. In this task a list of topics is given and the goal is to associate ontologies with each topic.

5.3.3. Condition of Success

We will have achieved success if the set of results can be sufficiently explained while providing better task results than using an independent set of ontologies in the proposed evaluation task.

5.4. Timetable

A tentative timetable is provided here. The expected completion of this thesis is December of 2007

TASK	Duration of task
Operator Set definition	15 days
Query Set Creation	1 month
Definition of a query context	15 days
Algorithms for concept matching	2 months
Evaluation of concept matching algorithms	1 month
Experiment with topological merging	1 month
Refinement of boosting algorithm	1 month
Evaluation of merging algorithm	1 month
Utility Based Ranking Algorithm	1 month
Evaluation of Ranking Algorithm	1 month
Final Task Based Evaluation	2 months
Thesis write up	3 months

6. References

- [Alani and Brewster, 2005] H. Alani and C. Brewster, "Ontology ranking based on the analysis of concept structures," *Proceedings of the 3rd international conference on Knowledge capture*, pp. 51-58, 2005.
- [Baker, et al., 1998] C. F. Baker, C. J. Fillmore, and J. B. Lowe, "The Berkeley FrameNet project," *Proceedings of COLING-ACL*, vol. 98, 1998.
- [Barwise and Seligman, 1997] J. Barwise and J. Seligman, *Information flow: the logic of distributed systems*: Cambridge University Press, 1997.
- [Bemers-Lee, et al., 2001] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, vol. 284, pp. 34-43, 2001.
- [Boag, et al., 2002] S. Boag, D. Chamberlin, M. F. Fernandez, D. Florescu, J. Robie, and J. Simeon, "XQuery 1.0: An XML Query Language," *W3C Working Draft*, vol. 15, 2002.
- [Brank, et al., 2005] J. Brank, M. Grobelnik, D. Mladenic, and B. Fortuna, "A survey of ontology evaluation techniques," in *Conference on Data Mining and Data Warehouses (SiKDD 2005)* Ljubljana, Slovenia, 2005.
- [Brewster, et al., 2004] C. Brewster, H. Alani, S. Dasmahapatra, and Y. Wilks, "Data Driven Ontology Evaluation," *International Conference on Language Resources and Evaluation, Lisbon, Portugal, 2004*.
- [Callan, 2000] J. Callan, "Distributed information retrieval. Advances in information retrieval," Kluwer Academic Publishers, 2000.
- [Chalupsky, 2000] H. Chalupsky, "OntoMorph: A Translation System for Symbolic Knowledge," *Principles of Knowledge Representation and Reasoning*, vol. 185, 2000.
- [Ding, et al., 2004] L. Ding, T. Finin, A. Joshi, R. Pan, R. S. Cost, Y. Peng, P. Reddivari, V. Doshi, and J. Sachs, "Swoogle: a search and metadata engine for the semantic web," *Proceedings of the Thirteenth ACM conference on Information and knowledge management*, pp. 652-659, 2004.
- [Doan, et al., 2004] A. Doan, J. Madhavan, P. Domingos, and A. Halevy, "Ontology matching: A machine learning approach," *Handbook on Ontologies in Information Systems*, pp. 397-416, 2004.
- [Fahlman, 2005] S. E. Fahlman, "Scone user's manual," 2005.
- [Fryer, 2004] D. Fryer, "Federated search engines," *Online(Weston, CT)*, vol. 28, pp. 16-19, 2004.
- [Ganter and Wille, 1997] B. Ganter and R. Wille, *Formal Concept Analysis: Mathematical Foundations*: Springer-Verlag New York, Inc. Secaucus, NJ, USA, 1997.
- [Gildea and Jurafsky, 2002] D. Gildea and D. Jurafsky, "Automatic labeling of semantic roles," *Computational Linguistics*, vol. 28, pp. 245-288, 2002.
- [Grosso, et al., 1999] W. E. Grosso, H. Eriksson, R. W. Ferguson, J. H. Gennari, S. W. Tu, and M. A. Musen, "Knowledge Modeling at the Millennium (The Design and Evolution of Protege-2000)," *Proceedings of the Twelfth Workshop on Knowledge Acquisition, Modeling and Management (KAW99)*, pp. 16-21, 1999.
- [Guarino, 1998] N. Guarino, *Formal ontology in information systems*: IOS Press, 1998.

- [Halevy, et al., 2003] A. Y. Halevy, Z. G. Ives, P. Mork, and I. Tatarinov, "Piazza: data management infrastructure for semantic web applications," *Proceedings of the twelfth international conference on World Wide Web*, pp. 556-567, 2003.
- [Hartmann, et al., 2004] J. Hartmann, P. Spyns, D. Maynard, R. Cuel, M. C. S. de Figueroa, and Y. Sure, "Methods for Ontology Evaluation," *KnowledgeWeb Deliverable# D*, vol. 1, December 2004.
- [Heflin, et al., 2003] J. Heflin, J. A. Hendler, and S. Luke, "SHOE: A blueprint for the semantic web," *Spinning the Semantic Web*, pp. 29-63, 2003.
- [Hovy, et al., 2003] E. H. Hovy, M. Fleischman, and A. Philpot, "The Omega Ontology," prep, 2003.
- [Jiang and Conrath, 1997] J. J. Jiang and D. W. Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy," *Proceedings of International Conference on Research in Computational Linguistics*, pp. 19-33, 1997.
- [Kalfoglou and Schorlemmer, 2002] Y. Kalfoglou and M. Schorlemmer, "Information-flow-based ontology mapping," *On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE*, pp. 1132-1151, 2002.
- [Kalfoglou and Schorlemmer, 2003] Y. Kalfoglou and M. Schorlemmer, "Ontology mapping: the state of the art," *The Knowledge Engineering Review*, vol. 18, pp. 1-31, 2003.
- [Kingsbury and Palmer, 2002] P. Kingsbury and M. Palmer, "From Treebank to PropBank," *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pp. 1989-1993, 2002.
- [Klein, 2001] M. Klein, "Combining and relating ontologies: an analysis of problems and solutions," *Workshop on Ontologies and Information Sharing, IJCAI*, vol. 1, 2001.
- [Ko, et al., 2006] J. Ko, L. Hiyakumoto, and E. Nyberg, "Exploiting multiple semantic resources for answer selection," *Proceedings of of LREC*, 2006.
- [Lacher and Groh, 2001] M. Lacher and G. Groh, "Facilitating the Exchange of Explicit Knowledge through Ontology Mappings," *Proceedings of the 14th International FLAIRS Conference*, May 2001.
- [Lenat, 1995] D. B. Lenat, "CYC: a large-scale investment in knowledge infrastructure," *Communications of the ACM*, vol. 38, pp. 33-38, 1995.
- [Lita, et al., 2004] L. V. Lita, W. Hunt, and E. Nyberg, "Resource analysis for question answering," *Association for Computational Linguistics Conference (ACL)*, 2004.
- [Lozano-Tello, et al., 2004] A. Lozano-Tello, A. Gomez-Perez, A. Lozano-Tello, and A. Gomez-Perez, "ONTOMETRIC: A method to choose the appropriate ontology," *Journal of Database Management*, vol. 15, pp. 1-18, 2004.
- [MacGregor, et al., 1997] R. MacGregor, H. Chalupsky, and E. R. Melz, "PowerLoom Manual," *ISI, University of South California*, 1997.
- [Maedche and Staab, 2002] A. Maedche and S. Staab, "Measuring similarity between ontologies," *Proceedings of the European Conference on Knowledge Acquisition and Management (EKAW)*, pp. 251-263, 2002.
- [McGuinness, et al., 2000] D. L. McGuinness, R. Fikes, J. Rice, and S. Wilder, "The Chimaera Ontology Environment," *Proceedings of the 17th National Conference on Artificial Intelligence*, 2000.

- [Mena, et al., 2000] E. Mena, A. Illarramendi, V. Kashyap, and A. P. Sheth, "OBSERVER: An Approach for Query Processing in Global Information Systems Based on Interoperation Across Pre-Existing Ontologies," *Distributed and Parallel Databases*, vol. 8, pp. 223-271, 2000.
- [Miller, 1995] G. A. Miller, "WordNet: A Lexical Database for English," *COMMUNICATIONS OF THE ACM*, vol. 38, pp. 11-39, 1995.
- [Mueller, 1997] E. T. Mueller, "Natural Language Processing with Thoughttreasure," Signiform, 1997.
- [Niles and Pease, 2001] I. Niles and A. Pease, "Towards a standard upper ontology," *Proceedings of the international conference on Formal Ontology in Information Systems-Volume 2001*, pp. 2-9, 2001.
- [Noy, 2004] N. F. Noy, "Semantic integration: a survey of ontology-based approaches," *ACM SIGMOD Record*, vol. 33, pp. 65-70, 2004.
- [Noy and Musen, 1999] N. F. Noy and M. A. Musen, "SMART: Automated Support for Ontology Merging and Alignment," *Twelfth Workshop on Knowledge Acquisition, Modeling, and Management, Banff, Canada*, 1999.
- [Noy and Musen, 2000] N. F. Noy and M. A. Musen, "PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment," *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2000.
- [Noy and Musen, 2002] N. F. Noy and M. A. Musen, "PromptDiff: A fixed-point algorithm for comparing ontology versions," *18th National Conference on Artificial Intelligence (AAAI-2002)*, 2002.
- [Nyberg, et al., 2003] E. Nyberg, T. Mitamura, J. Callan, J. Carbonell, R. Frederking, K. Collins-Thompson, L. Hiyakumoto, Y. Huang, C. Huttenhower, and S. Judy, "The javelin question-answering system at trec 2003: A multi-strategy approach with dynamic planning," *Proceedings of the Twelfth Text REtrieval Conference (TREC2003)*, 2003.
- [Patel, et al., 2003] C. Patel, K. Supekar, Y. Lee, and E. K. Park, "OntoKhoj: a semantic web portal for ontology searching, ranking and classification," *Proceedings of the fifth ACM international workshop on Web information and data management*, pp. 58-61, 2003.
- [Philpot, et al., 2003] A. Philpot, M. Fleischman, and E. H. Hovy, "Semi-Automatic Construction of a General Purpose Ontology," *Proceedings of the International Lisp Conference. New York, NY. Invited*, 2003.
- [Porzel and Malaka, 2004] R. Porzel and R. Malaka, "A Task-based Approach for Ontology Evaluation," *ECAI Workshop on Ontology Learning and Population, Valencia, Spain*, 2004.
- [Prasad, et al., 2002] S. Prasad, Y. Peng, and T. Finin, "Using explicit information to map between two ontologies," *Proceedings of the AAMAS 2002 Workshop on Ontologies in Agent Systems (OAS'02)*, pp. 52-57, 2002.
- [Quillian, 1967] M. R. Quillian, "Word concepts: a theory and simulation of some basic semantic capabilities," *Behav Sci*, vol. 12, pp. 410-30, 1967.
- [Raymond, et al., 2002] J. W. Raymond, E. J. Gardiner, and P. Willett, "RASCAL: Calculation of Graph Similarity using Maximum Common Edge Subgraphs," *The Computer Journal*, vol. 45, pp. 631-644, 2002.

- [Reed and Lenat, 2002] S. Reed and D. Lenat, "Mapping ontologies into cyc," *AAAI 2002 Conference Workshop on Ontologies For The Semantic Web, Edmonton, Canada, July, 2002*.
- [Resnik, 1995] P. Resnik, "Using information content to evaluate semantic similarity in a taxonomy," *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, vol. 1, pp. 448-453, 1995.
- [Ross, 1976] S. M. Ross, *A first course in probability*: Macmillan, 1976.
- [Schuler, 2003] K. K. Schuler, "VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon," Ph. D. thesis proposal, University of Pennsylvania, 2003.
- [Serafini and Taminin, 2005] L. Serafini and A. Taminin, "Drago: Distributed reasoning architecture for the semantic web," *Proc. of the Second European Semantic Web Conference (ESWC'05)*, vol. 3532, pp. 361-376, 2005.
- [Si and Callan, 2005] L. Si and J. Callan, "Modeling search engine effectiveness for federated search," *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 83-90, 2005.
- [Stumme and Maedche, 2001] G. Stumme and A. Maedche, "FCA-Merge: Bottom-up merging of ontologies," *7th Intl. Conf. on Artificial Intelligence (IJCAI'01)*, pp. 225-230, 2001.
- [von Ahn and Dabbish, 2004] L. von Ahn and L. Dabbish, "Labeling images with a computer game," *Proceedings of the 2004 conference on Human factors in computing systems*, pp. 319-326, 2004.
- [Voorhees, 2003] E. M. Voorhees, "Overview of the TREC 2003 Question Answering Track," *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*, 2003.
- [Zhang, et al., 2004] Y. Zhang, W. Vasconcelos, and D. Sleeman, "Ontosearch: An ontology search engine," *Proc. 24th SGAI Int. Conf. on Innovative Techniques and Applications of Artificial Intelligence*, 2004.
- [Ziegler, et al., 2006] P. Ziegler, C. Sturm, and K. R. Dittrich, "The SIRUP Ontology Query API in Action," *LECTURE NOTES IN COMPUTER SCIENCE*, vol. 3896, p. 1172, 2006.