HyKSS: A Multiple Ontology Approach to Hybrid Search

by

Andrew Zitzelberger

A thesis proposal submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computer Science

Brigham Young University

June 2010

Abstract. The rapid production of digital information has made the task of locating relevant information increasingly difficult. Keyword search alleviates this difficulty by retrieving documents containing keywords of interest, but suffers from issues such as ambiguity and synonymity. Semantic search resolves these issues, but is limited by the quality of semantic annotations that are likely to be imprecise and incomplete. In this proposal we introduce HyKSS, a *Hy*brid *K*eyword and *S*emantic *S*earch system. HyKSS mitigates the weaknesses of these two approaches by employing both techniques in a hybrid ranking process in order to retrieve more relevant results. Further, HyKSS is not limited to queries over a single domain, but can employ multiple ontologies representing several domains from its customizable library in the ranking process. We expect to be able to show that the proposed hybrid system outperforms keyword search and semantic search.

1 Introduction

Digital information is being produced at a prodigious rate. The Web alone is estimated to contain billions of digital documents [GS05] and is still growing.[1] This unprecedented access to information, however, has made the task of locating relevant information increasingly difficult.

Keyword search methods alleviate the problem by retrieving documents that contain words specified by users. The retrieved documents are likely to contain relevant information because of the presence of words of interest. However, keyword search suffers from a number of issues. Ambigous keywords, which have more than one possible meaning, often result in the retrieval of irrelevant documents. Alternatively, relevant documents may contain terms different from, but similar in meaning to, specified keywords. These documents will be missed in the retrieval process. Further, keyword searches are ineffective for specifying comparison-based constraints on information, such as "less than 14 grand," because they are only capable of matching keywords. In this case a user is not interested in finding the words

[1]In November of 2008 Google announced the indexing of 1 trillion unique URLs with a growth rate of several billion per day. See http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html.

contained in the constraint, but rather is interested in finding information that satisfies the constraint.

Semantic search resolves these issues by considering the semantics of information in relation to a user's query. For purposes of this proposal, we consider semantics to be the classification of information according to some schema. These semantics are provided using machine-readable annotations. Semantic search is the process of interpreting a user's query with respect to underlying semantic annotations in order to provide relevant results. These results can be the relevant annotations themselves or the documents that contain them.

However, producing semantic annotations is not a trivial task. Manual hand annotation is time consuming and human annotators may disagree on which annotation is most appropriate. Alternatively, semi-automatic or automatic tools can extract information and produce annotations. While great progress has been made on such tools, even the best miss critical information or provide inaccurate annotations. Searching over these annotations alone may miss relevant information that is not properly annotated.

In this proposal we introduce HyKSS (pronounced "hikes"), a *Hy*brid *K*eyword and *S*emantic *S*earch system. HyKSS mitigates the weaknesses of keyword and semantic search by employing both techniques in a hybrid search process. Semantic search helps overcome the ambiguity, synonymity, and constraint problems; keyword search helps overcome annotations that are likely imprecise and incomplete. HyKSS is fundamentally an information retrieval engine concerned with returning links to documents that are relevant to users' textual free-form queries. However, the inclusion of semantic search allows us to augment document links with relevant annotations and provide highlighting of relevant information within the document.

At the heart of the semantic search process is a library of semantic models called extraction ontologies. An extraction ontology is a means of tying linguistic information to a formal conceptual model [EZ10]. These extraction ontologies serve a dual role in HyKSS. First, they extract information from documents and produce semantic annotations. Second, they interpret possible semantics of user queries. Upon query execution, the interpretations and annotations they provide produce a semantic score for each document. HyKSS combines this semantic score with a keyword score to produce a final score and ranking for each document.

By default, HyKSS provides extraction ontologies for a number of simple concepts. While large ontologies can model an entire domain, HyKSS is specifically designed to work across multiple smaller ontologies targeting concepts of interest. The query processing mechanism allows queries to use multiple, possibly unrelated, ontologies in computing document relevance. Additionally, the library of ontologies is customizable. Users can tune the semantic understanding of the system to meet their needs through customization or by using models provided by others.

To demonstrate the HyKSS system, consider a query such as "red honda 'no dings' orem under 14 grand." A simple *Car* ontology such as the one in Figure 1 recognizes "red" as a *Color*, "honda" as a *Make*, and "under 14 grand" as a constraint on *Price*. A *Location* ontology will pick up "orem" as a *City*. However, neither model understands the phrase "no dings." Using semantic search alone for this query would require the user to manually search through documents to determine if dings are present. Alternatively, keyword search attempts to retrieve documents based on keywords in the query. However, terms such as "under", "14", and "grand" are unlikely to appear in relevant documents. Rather, the user is likely interested in finding a car with a price that satisfies the constraint.

HyKSS leverages the strengths of both keyword and semantic search by combining them in a hybrid search process. Semantic search gives a high score to documents with information that satisfies the understood semantic constraints, and keyword search gives a high score to documents with the specified keywords (keywords from comparison-based constraints, such as numerical constraints, are removed before keyword processing). The final results of hybrid processing combines the scores from both processes to produce a final ranking in which high-scoring documents meet semantic constraints and contain relevant keywords. Further, HyKSS will display results in a manner that enables users to quickly see keywords and relevant annotations that appear in the document.

We expect to be able to show that HyKSS outperforms both keyword and semantic search in terms of mean average precision [MRS08]. We will also demonstrate that extraction ontologies can be used to drive both the annotation and semantic search mechanisms of such a hybrid search system.

2 Related Work

Many researchers have invested effort in using semantics to provide enhanced search capabilities over document collections. Many of these efforts rely on information extraction and integration methods to produce a structured database of facts (e.g. [Caf09], [CRS$^+$07], [CBC$^+$07]). The focus of these systems is to provide structured querying capabilities over document contents. However, performing structured queries requires some user knowledge of schematic structures and structured query languages. While some systems do allow the use of search queries in addition to structured queries, they process each independently rather than in a hybrid manner.

Other systems allow free-form textual queries over semantic annotations by translating unstructured queries into structured queries. Approaches such as Avatar [KKR$^+$06], SPARK [ZWX$^+$07], and the work of Tran, et al. [TCRS07] accept keyword queries and return ranked structured queries that may meet users' information needs. Alternatively, the QBK approach [TCL09] returns a structured query template the user can complete in order to find relevant information. Some structured query languages allow for the specification of keywords but interpret them in a Boolean manner within the structured query rather than processing them as a separate keyword process. Other approaches accept textual queries and return answers directly from annotations but require the use of full natural-language queries rather than keyword queries (e.g. [LPM05], [KBZ06]). While these approaches mitigate the need to understand underlying schematic structures and structured queries, they are a form of semantic search rather than hybrid search. As such, they are limited by the quality of the semantic annotations.

Several systems employ ranking processes that are hybrid in nature. OntoSearch [JT06] accepts an initial keyword query and uses keyword search to retrieve an initial document set. It then employs a spread activation-inference process over an underlying semantic network to rank documents. This process may add documents to or remove documents from the initially-retrieved document set as it computes the final document rankings. Rinaldi [Rin09] presents a system that uses WordNet[2] and lexical chain processing to rank documents according to the semantic similarity to the original query. This system does not require semantic annotations, but the focus on linguistic processing prohibits the successful completion of comparison-based queries.

Fazzinga, et al. [FGGL10] present a method for performing ontological conjunctive queries using a Datalog-like query language. These researchers designed and built their system on top of existing search engines in order to provide ranking scores. However, they assume the existence of annotations. MELISA [AG00] uses an underlying ontology to model the Medline[3] database. This system uses a form-based interface but also allows users to include keywords of interest. This interface is similar to the advanced search in HyKSS, but HyKSS works across multiple domains.

Bhagdev, et al. [BCC$^+$08] present a hybrid architecture similar to HyKSS. The researchers provide a formalization of a hybrid search concept that includes both keyword and semantic search. They also provide a reference implementation, K-Search. The primary differences between HyKSS and K-Search is that HyKSS allows queries to use multiple ontologies and provides a textual rather than a purely form-based interface. Wang, et al. [WTL08] propose a scalable architecture for this type of hybrid search. Castells, et al. [CFV07] also present a hybrid search system similar to HyKSS. They present an extension to the traditional vector space model to allow for ontology-based retrieval. This system

[2]http://wordnet.princeton.edu/
[3]http://www.nlm.nih.gov/databases/databases_medline.html

provides semi-automatic extraction using ontologies that meet minimal specifications. However, users must submit structured queries that are translated into keyword queries instead of vice versa. All of these hybrid systems have shown improvement over both keyword and semantic search across various metrics.

Our work includes an extension of AskOntos [Vic06] and SerFR [AM07]. While serving different purposes, the premise of both systems is to interpret free-form textual queries in order to query semantic annotations. Both AskOntos and SerFR use ontologies to extract semantic annotations and drive the query process. HyKSS augments this functionality with keyword search and semantic ranking. HyKSS also provides additions that allow queries to be run across multiple ontologies rather than a single ontology at a time.

Our work on HyKSS can be seen as a partial implementation of a dataspace. A dataspace is an abstraction that takes the *data everywhere* approach: all and any resources can be queried using keyword search, but the search capabilities of the system can be improved through annotation and integration [FHM05, HFM06]. As proposed, HyKSS is currently limited to HTML and PDF documents, but has the potential to be expanded to other resource types in the future. Although our system does not make recommendations to the user about improvements based on usage, we do allow users to modify the extraction-ontology library to tune the semantics of the system in a *pay-as-you-go* manner.

It is important to recognize that popular search engines such as Google already employ some means of hybrid search. Google's base PageRank [BP98] algorithm considers the structure of links between documents in addition to the presence of keywords. Additionally, Google has since augmented their search algorithms to include features such as rich snippets[4] to take advantage of semantic annotations and social search[5] to take advantage of known user social contacts. Google recently announced that they also now use Google Squared[6] technology to provide semantic results to searches directly on the results page.[7] Each of

[4]http://googlewebmastercentral.blogspot.com/2009/05/introducing-rich-snippets.html
[5]http://googleblog.blogspot.com/2009/10/introducing-google-social-search-i.html
[6]http://www.google.com/squared
[7]http://googleblog.blogspot.com/2010/05/understanding-web-to-find-short-answers.html

these, and other tools by Google and mainstream search engines, are likely to employ hybrid search in some form, but it is unclear how and to what extent.

3 Thesis Statement

HyKSS, our proposed *Hy*brid *K*eyword and *S*emantic *S*earch system, outperforms both keyword search and semantic search over imprecise and incomplete annotations in terms of mean average precision. We also demonstrate that HyKSS can perform queries using multiple disparate ontologies and that its semantic search capabilities can improve through pay-as-you-go enhancements to the semantic library.

4 Project Description

This section discusses the proposed hybrid document ranking system in detail. We begin with a brief introduction to extraction ontologies and the data frame components that make extraction possible. We then present the overall architecture of the system and discuss each component. We conclude with a discussion of the query processing step.

4.1 Extraction Ontologies

An extraction ontology is a conceptual model augmented with linguistic information to enable information extraction over text. The model primarily consists of object sets, relationships sets, and constraints. Figure 1 shows an extraction ontology in its conceptual-model form. Each rectangular box represents an object set. A box with a dashed border indicates that the object-set concept has associated linguistic information allowing for instance recognition in text. A box with a solid border, alternatively, indicates that instances of the concept are not found in text but rather are generated when sufficient evidence appears in related lexical concepts and in keywords or keyword phrases that indicate the presence of a nonlexical object set. Line segments between object sets denote relationship sets. Next to the endpoint of each relationship set is a participation constraint in *min*:*max* format. The

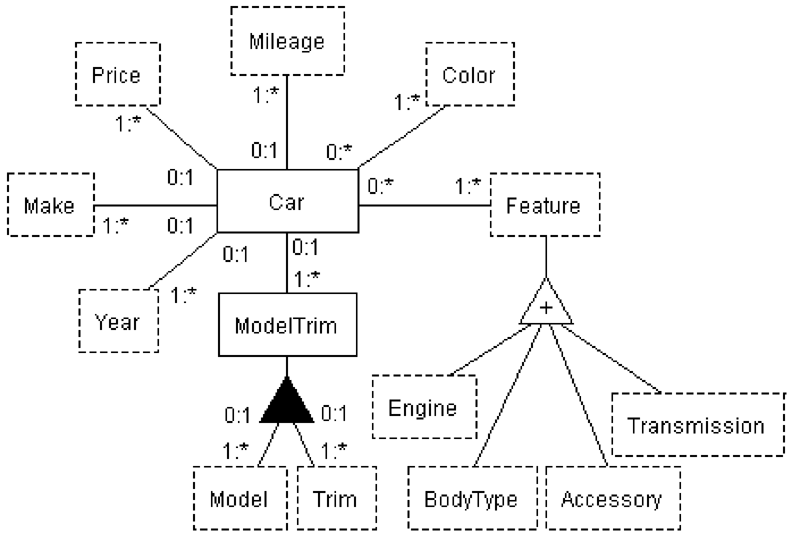


Figure 1: An example extraction ontology for the car concept.

"∗" symbol indicates unlimited cardinality. The black triangle and the white triangle specify aggregation and generalization/specialization respectively, where the concept connected to the apex of the triangle is the holonym or hypernym concept, respectively.

Linguistic information is tied to lexical concepts using data frames. A data frame uses a series of regular expressions to capture the essential properties of concept instances [Emb80]. Figure 2 shows part of a sample data frame for *Price*. The *internal representation* indicates how the system should store extracted values internally. *External representations* consist of a series of regular expressions specifying how instances might appear in text. The distance of matches from *context keywords* helps determine which match to choose for potentially ambiguous functional concepts. A number, for example, could be interpreted as a *Number* or a *Price*, but would be interpreted as a *Price* when words such as *asking* or *negotiable* appear near it.

Units, the *canonicalization method*, and *comparison methods* allow for semantic comparisons over extracted values. *Units* express units of measure or value qualifications that help quantify extracted values. In Figure 2, *K* indicates multiplication by 1,000 and *dollars* specifies a type of currency. A *canonicalization method* converts an extracted value and

```
Price
  internal representation: Double
  external representations: \$[1-9]\d{0,2},?\d{3} | \d?\d [Gg]rand | ...
  context keywords: price|asking|obo|neg(\.|otiable)| ...
  ...
  units: dollars|[Kk] ...
  canonicalization method: toUSDollars
  comparison methods:
   LessThan(p1: Price, p2: Price) returns (Boolean)
   external representation: (less than | < | under | ...)\s*{p2} | ...
   ...
  output method: toUSDollarsFormat
  ...
end
```

Figure 2: Sample data frame for price.

units (if any) to a unified internal representation. Once in this representation, *comparison methods* can compare values extracted from different documents despite being represented in different ways. These methods can correctly confirm, for example, that "$4,500" is less than "5 grand." The *output method* is responsible for displaying internally-stored values to the user in a readable format. This comes into play when HyKSS displays search results to users.

OntoES, our *Onto*logy *E*xtraction *S*ystem, applies extraction ontologies to web pages to extract information and produce annotations. The extraction processes uses the linguistic information provided by data frames and the constraints of the model structure under the guidance of heuristics to perform information extraction tasks. Past work shows that OntoES performs well in terms of precision and recall for the extraction task when documents are rich in recognizable constants and narrow in ontological breadth [ECJ+99].

4.2 Architecture

The proposed HyKSS architecture consists of two disparate index repositories for storing and processing document representations: the keyword index and the semantic index. HyKSS

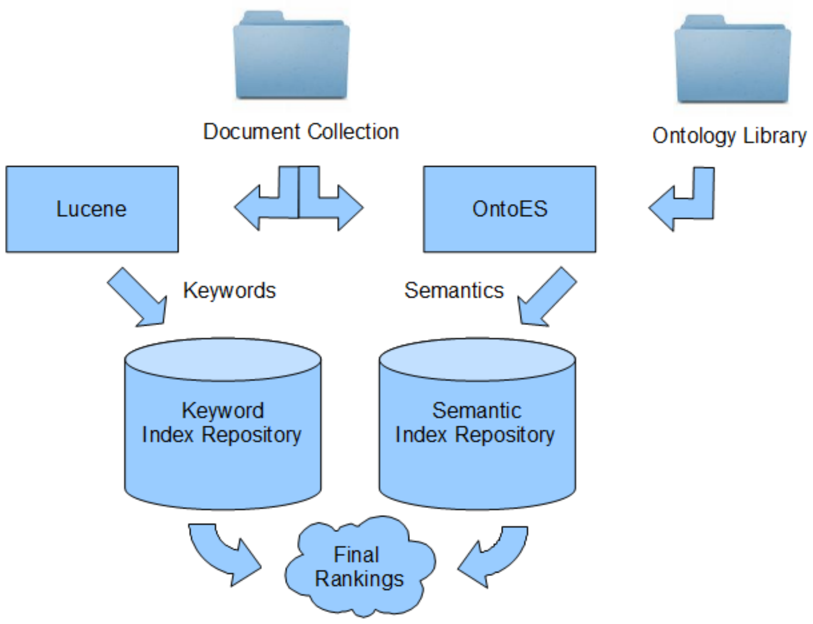


Figure 3: Architecture of the proposed system.

produces both of these index repositories from a document collection. Any change to the document collection requires a reindexing of the modified portion of the collection.

We plan to use Lucene's[8] full text indexing capabilities to create the keyword index and store it using a series of binary files. We construct the semantic index repository using the library of extraction ontologies. HyKSS applies each extraction ontology in the library to each document and uses an indexed RDF triple database to store extracted annotations. We are considering AllegroGraph[9] and Sesame[10] as viable triple databases for this purpose.

By default the library comes with pre-built extraction ontologies for simple concepts such as number, price, distance, and height. Users can modify the semantics available to the system by adding new ontologies, removing existing ontologies, or modifying existing ontologies. These types of modifications to the ontology library require a reindexing of the document collection with the modified portion of the ontology library.

[8]http://lucene.apache.org/
[9]http://www.franz.com/agraph/allegrograph/
[10]http://www.openrdf.org/

4.3 Query Processing

The primary user interface of HyKSS accepts free-form textual queries from users. These queries can be as simple as a single keyword and as complex as a natural language conjunctive query. We expect typical queries to consist of several keywords and contain small phrases specifying any needed comparison-based constraints. The advanced search option provides a form-based interface that allows for the addition of disjuncts and negations that are difficult to parse from free-form text. The query processing unit produces similarity scores by comparing user queries with both keyword-index and semantic-index repositories and combining the scores.

While HyKSS can essentially compute the two scores in parallel, it requires a minimum amount of semantic processing to detect and remove comparison-based constraints before proceeding with keyword processing. Constraints such as “less than 10 grand” or “rated very good or higher” do not specify actual terms the user is interested in finding in relevant documents, but rather constraints on information found in the documents. As such, they are likely to add noise to keyword search, and they should be removed. HyKSS uses the modified query (the original query with phrases specifying comparison-based constraints removed) to produce a keyword score using a combined Boolean and Vector Space Model as provided by Lucene. This score will always be between 0 and 1 inclusive where a larger number indicates greater similarity.

The semantic score consists of two subscores: the ontology match score and the document match score. The ontology match score measures the similarity between a textual user query and an ontology or set of ontologies. HyKSS first produces a similarity score for each ontology by extracting concepts from the user query to determine similarity. The exact means of computing this score is one aspect of this research. One simple method is simply to count the number of concept matches where a larger number indicates greater similarity. While this approach may favor larger ontologies, normalizing by the number of concepts may favor smaller ontologies. We plan on experimenting with a variety of approaches.

After HyKSS scores each individual ontology, it considers sets of ontologies. Each ontology whose score is above some minimum similarity threshold can be combined with any other ontology or ontology set, so long as one of the ontologies is not completely subsumed in the extraction. Consider the running example query "red honda 'no dings' orem under 14 grand." The *Car* ontology in Figure 1 recognizes "red" as a *Color*, "honda" as a *Make*, and "under 14 grand" as a constraint on *Price*. Suppose that a *Location* ontology recognizes "orem" as a *City*. These ontologies can be combined into an ontology set because neither ontology subsumes the other in their extractions. In this case the combined scores of the individual ontologies indicate that the set of ontologies better matches the query than either ontology individually.

Alternatively, suppose a *Clothing* ontology picks up "red" as a *Color*. In this case HyKSS would not combine the *Clothing* ontology with the *Car* ontology to create an ontology set because the extractions from the *Car* ontology subsume the *Clothing* ontology extraction. HyKSS could combine the *Clothing* and *Location* ontologies but this combination would rank lower than the *Car* and *Location* combination because fewer relevant values are extracted for the set. Further, the *Clothing* and *Location* ontology set does not need to be considered for query processing because the extractions of the *Car* and *Location* ontology set subsume its extractions. If ontologies only overlap for some extractions, the ontologies can still be combined into an ontology set for consideration.

The document match score refers to how well the extracted annotations for a document match the user's initial query. Continuing with the query in the running example, a document that explicitly states that a Honda is red should be ranked higher than a document that does not specify the color. The method for computing this document match score is a focus of this research. A simple approach might be to sum the constraints satisfied and subtract the constraints violated. Experimentation might show that one aspect of this formula should be weighted more heavily than another. We plan on having a scoring approach that does not penalize documents for constraints that are neither satisfied nor violated.

HyKSS computes the final score for any document by combining the keyword score and the semantic score. HyKSS uses the following formula for this computation:

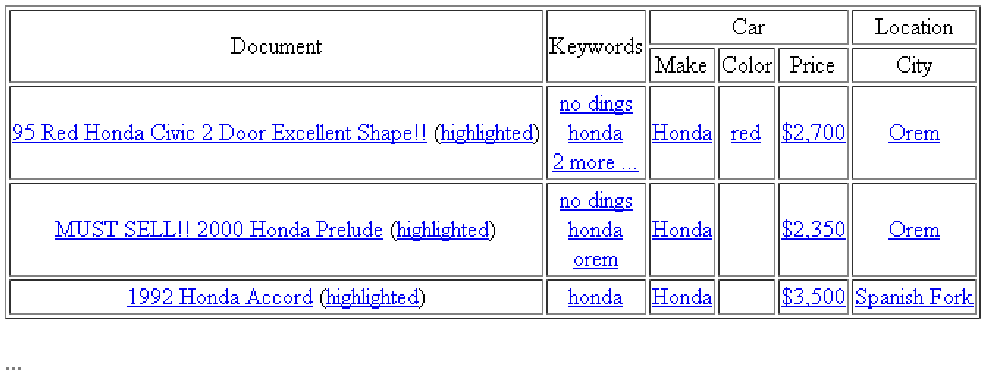
$$\alpha k \cdot \beta(\gamma o + \delta d)$$

where k is the keyword score, o is the ontology match score, and d is the document match score. While the keyword score is guaranteed to be between 0 and 1 inclusive, the semantic score may not be, depending on the method used for calculation. Semantic score normalization may therefore be necessary. The parameters α, β, γ, and δ are adjustable parameters for varying the weights for each score. Determining good values for these parameters is part of this research project as well.

Advanced Search

In addition to processing free-form textual queries HyKSS also offers an advanced search option. Similar to traditional search engines, the advanced search allows users to enter simple Boolean queries. The difference, however, is that because HyKSS has a semantic foundation, these Boolean queries can involve semantic annotations rather than just keywords.

With the advanced search option, a user still begins by entering a textual query. After HyKSS ranks each ontology and ontology set with respect to the query, HyKSS displays the best ontology or ontology set in a form layout. Users can then see what the system "understood" from the query and can make modifications to the query. The form allows users to add disjuncts and negations. The form also allows for entry of keyword Boolean search queries over documents and for the specification of traditionally-processed keywords. After clicking submit, the query processing functions as usual.



Document	Keywords	Car			Location
		Make	Color	Price	City
95 Red Honda Civic 2 Door Excellent Shape!! (highlighted)	no dings honda 2 more ...	Honda	red	$2,700	Orem
MUST SELL!! 2000 Honda Prelude (highlighted)	no dings honda orem	Honda		$2,350	Orem
1992 Honda Accord (highlighted)	honda	Honda		$3,500	Spanish Fork

...

Figure 4: Example results page.

Result Page

The output of HyKSS is similar to other search engines, but HyKSS augments the results with additional information. HyKSS displays results as a list of tables. The first column of any non-header row in a table includes a link to the original document and a link to a script that highlights relevant values in the document. These relevant values consist of the keywords and extracted annotations displayed in the other columns. The second column is a list of relevant keywords found in the document. The remaining columns display extracted annotations that are relevant to the query. HyKSS displays these annotations under a heading that names the ontology that extracts them. Each of the listed keywords and annotations are links to a script that highlights the listed value in the document. Figure 4 shows example results for our running query example. The first table shows that *Make*, *Color*, and *Price* were extracted using the *Car* ontology while *City* was extracted using the *Location* ontology.

Results are displayed in order of relevance according to the computed match scores. For readability, we restrict table columns to containing three keyword or annotation lines. If more than three occur in the document, the third line will consist of a link specifying "n more ...", where n is the number of additional keywords or annotations. This link activates an additional page displaying the remaining relevant results. Although HyKSS can generate a table header for each document, to maintain readability, HyKSS adds results to the previous table header when possible.

5 Validation

To validate our claim that hybrid search outperforms traditional keyword search and semantic querying over imprecise and incomplete annotations, we will provide a comparison of the results for each of these approaches. We will pseudo-randomly generate 100 queries using the underlying ontologies and relevant dictionary files. We say pseudo-randomly because we will design the system to only create queries that make sense and will eliminate queries that do not. The query generator will not attempt to generate natural-language queries, but rather queries that include keyword terms and randomly inserted comparison-based constraints from ontologies. This generated query format is the same format we expect from typical users. We will run each query using keyword processing (without removing comparison-based constraints) in isolation, using semantic search processing in isolation, and using the full hybrid system.

The document collection for testing will consist of pages from Wikipedia,[11] advertisements from craigslist,[12] and historical documents from Ancestory.com.[13] We will partition the document collection into two sets. The first set will be used for development testing of HyKSS and its ontology library. The second set will be a blind-test set reserved for the final experiments. We will compare each approach over the document collection using mean average precision (MAP). MAP combines precision and recall into a single-value metric that puts an emphasis on the ranking of documents. MAP is computed using the following formula:

$$MAP(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{M_j} \sum_{k=1}^{M_j} Precision(R_{jk})$$

[11]http://www.wikipedia.org/
[12]http://www.craigslist.org
[13]http://www.ancestry.com/

where Q is the set of queries, M_j is the number of relevant documents in the collection for query j, and $Precision(R_{jk})$ is the precision of a relevant document R for query j and recall level k. MAP is an approximation of the area under an average precision vs. recall curve. For visualization of our results, we will include these curves as well. We expect that the hybrid approach will outperform both keyword search and semantic search over the MAP metric.

We also demonstrate the ability to perform queries across multiple ontologies and to customize system semantics using proof by construction. Additionally, we will report on the size of both the keyword and semantic indices used for experimentation, as well as the size of the source file set used to generate them. We will also report on the query processing times of our system using a single processor machine.

6 Thesis Schedule

Design and Coding	June 2010 - October 2010
Experiments	November 2010 - December 2010
Chapter 2 - HyKSS Architecture and Query Processing	January 2011
Chapter 3 - Experimental Results	February 2011
Chapter 4 - Related Work	February 2011
Chapters 1 and 5 - Introduction and Conclusion	February 2011
Thesis Revision and Defense	March 2011

References

[AG00] Jose María Abasolo and Mario Gómez. MELISA. An Ontology-Based Agent for Information Retrieval in Medicine. In *Proceedings of the First International Workshop on the Semantic Web (SemWeb2000)*, pages 73–82, Libson, Potrugal, 2000.

A form based interface for searching the Medline database that also allows for the entry of keywords. MELISA also employs a hybrid architecture but is limited to a single ontology and single domain.

[AM07] Muhammed J. Al-Muhammed. *Ontology Aware Software Service Agents: Meeting Ordinary User Needs on the Semantic Web.* PhD thesis, Brigham Young University, Provo, Utah, August 2007.

SerFR uses underlying ontologies to interpret user textual queries and assists them in scheduling appointments. Our process of interpreting user textual queries grows out of the ideas used for SerFR but we expand it to allow for queries across multiple ontologies.

[BCC$^+$08] Ravish Bhagdev, Sam Chapman, Fabio Ciravegna, Vitaveska Lanfranchi, and Daniela Petrelli. Hybrid Search: Effectively Combining Keywords and Ontology-Based Searches. In *Proceedings of the 5th European Semantic Web Conference (ESWC'08)*, pages 554–568, Tenerife, Canary Islands, Spain, June 2008.

Presents a formilization of the hybrid search concept as well as an example implemenation, K-Search. Our system differs from the proposed reference implementation in that our system allows for queries to cross the boundries of potentially unrelated ontologies. Further, we use a textual query approach rather than a form based query approach for the primary interface.

[BP98] Sergey Brin and Lawrence Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. In *Proceedings of the 7th International World Wide Web Conference (WWW'98)*, pages 107–117, Brisbane, Australia, April 1998.

Presents the PageRank algorithm that considers hyperlink structure of web document in addition to the words in the document. Google, whos ranking algorithm began with PageRank, has since added more semantic processing to the ranking process. Its not clear to what degree Google employs hybrid search.

[Caf09] Michael J. Cafarella. Extracting and Querying a Comprehensive Web Database. In *Proceedings of the 4th Conference on Innovative Data Systems Research (CIDR'09)*, Asilomar, California, January 2009.

Presents a system that uses multiple domain-independent extractors in an ensemble like manner and then provides the ability to query over the extracted facts. This approach allows for querying of facts across multiple web documents but is not a hybrid search system and is limited by the quality of the extracted annotations.

[CBC+07] Eric Chu, Akanksha Baid, Ting Chen, AnHai Doan, and Jeffrey Naughton. A Relational Approach to Incrementally Extracting and Querying Structure in Unstructured Data. In *Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB'07)*, pages 1045–1056, Vienna, Austria, September 2007.

Presents a workbench that allows administrators to apply extract, integrate, and cluster operations to documents in order to produce high quality annotations that can be queried. This strategy allows for pay-as-you-go creation of a search engine that takes sdvantage of available semantics. However this system uses keyword and semantic search capabilities separately rather than together in a hybrid search process.

[CFV07] Pablo Castells, Miriam Fernandez, and David Vallet. An Adaptation of the Vector-Space Model for Ontology-Based Information Retrieval. *IEEE Transactions on Knowedge and Data Engineering*, 19(2):261–272, February 2007.

Presents an adaption of the traditional vector space model to allow for ontology based retrieval. This system uses a hybrid approach to retrieving documents but requires a structured query that is translated to a keyword query instead of vice versa. The system does not appear to handle queries that cross ontologies.

[CRS+07] Michael J. Cafarella, Christopher Ré, Dan Suciu, Oren Etzioni, and Michele Banko. Structured Querying of Web Text: A Technical Challenge. In *Proceedings of the 3rd Conference on Innovative Data Systems Research (CIDR'07)*, pages 225–234, Asilomar, California, USA, January 2007.

Stores extracted information in a probablistic web database to account for the inexact nature of the extraction task and allows querying over the extracted facts. This system allows for ranking of results based on the probability that a fact is correct. However, this system is a form of semantic search rather than a hybrid search system.

[ECJ$^+$99] David W. Embley, Douglas M. Campbell, Yuan S. Jiang, Stephen W. Liddle, Deryle W. Lonsdale, Yiu-Kai Ng, and Randy D. Smith. Conceptual-Model-Based Data Extraction from Multiple-Record Web Pages. *Data & Knowledge Engineering*, 31(3):227–251, November 1999.

Provides information about the extraction capabilities of OntoES, our in house ontology extraction system. OntoES is used to apply extraction ontologies to web pages and create useful semantic annotations in the proposed system.

[Emb80] David W. Embley. Programming with Data Frames for Everyday Data Items. In *Proceedings of the AFIPS National Computer Conference (AFIPS'80)*, pages 301–305, Anaheim, California, May 1980.

Original paper intoducing the concept of data frames as a means to tie linquistic information to everday objects. Data frames provide the means by which our ontologies become information extraction ontologies, meaning they can recognize instances from text.

[EZ10] David W. Embley and Andrew Zitzelberger. Theoretical Foundations for Enabling a Web of Knowledge. In *Proceedings of the Sixth International Symposium on Foundations of Information and Knowledge Systems (FoIKS'10)*, pages 211–229, Sofia, Bulgaria, February 2010.

Includes a formilization of extraction ontologies and data frames and how they can be used to build a web of knowledge. These components are the essential building blocks driving both the extraction and semantic query processes, as well as providing the ability for customization.

[FGGL10] Bettina Fazzinga, Giorgio Gianforme, Georg Gottlob, and Thomas Lukasiewicz. Semantic Web Search Based on Ontological Conjunctive Queries. In *Proceedings of the Sixth International Symposium on Foundations of Information and Knowledge Systems (FoIKS'10)*, pages 153–172, Sofia, Bulgaria, February 2010.

Presents a generalization of the PageRank algoirthm that considers both documents and annotations when returning relevant information to the user. This system assumes the existence of annotations and is designed to be built upon current search engines. This system is a hybrid search system but requires users to submit structured queries.

[FHM05] Michael Franklin, Alon Halevy, and David Maier. From Databases to Dataspaces: A New Abstraction for Information Management. *SIGMOD Record*, 34(4):27–33, December 2005.

Intoduces data spaces as new abstraction for data management. Data spaces must be able to handle any type of data and offer incremental improvement of features through extraction and integration. Our system is not a data space but does allow for pay-as-you-go improvements through extraction ontology modification.

[GS05] Antonio Gulli and Alessio Signorini. The Indexable Web is More than 11.5 billion Pages. In *Proceedings of the 14th International World Wide Web Conference (WWW'05)*, pages 902–903, Chiba, Japan, May 2005.

Provides evidence as to the size of the web in terms of indexable pages. This helps motivate the need for enhanced search tools.

[HFM06] Alon Halevy, Michael Franklin, and David Maier. Principles of Dataspace Systems. In *Proceedings of the 25th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'06)*, pages 1–9, Chicago, Illinois, June 2006.

Provides an explanation of the principles necessary to produce a dataspace service provider. These principles include things such as handling all resources, providing an integrated means of searching, querying, updating, and administrating, and returning best effort results. Our system does not claim to be a dataspace service provider, but we do claim to implement several features such as providing search over unannotated resources and offering pay-as-you-go improvements to the system.

[JT06] Xing Jiang and Ah-Hwee Tan. OntoSearch: A Full-Text Search Engine for the Semantic Web. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI'06)*, pages 1325–1330, Boston, Massachusetts, July 2006.

Presents a hybrid search system in which a keyword search query is used to obtain an initial document set and a spread activation algorithm is used to re-rank and discover new documents based on the underlying semantic network. Our system uses disparate ontologies rather than a single semantic network and we allow queries to cross ontology boundries.

[KBZ06] Esther Kaufmann, Abraham Bernstein, and Renato Zumstein. Querix: A Natural Language Interface to Query Ontologies Based on Clarification Dialogs. In *5th International Semantic Web Conference (ISWC'06)*, pages 980–981, Athens, Georgia, November 2006.

Querix is a completely protable question-answer interface for the Semantic Web that relies on a reduced set of NLP tools and the WordNet ontology. Querix is an example of a natural language interfaces for semantic search and is not a hybrid search engine.

[KKR+06] Eser Kandogan, Rajasekar Krishnamurthy, Sriram Raghavan, Shivakumar Vaithyanathan, and Huaiyu Zhu. Avatar Semantic Search: A Database Approach to Information Retrieval. In *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data (SIGMOD'06)*, pages 790–792, Chicago, Illinois, June 2006.

Avatar allows users to submit a keyword query and then returns a set of ranked semantic query interpretations. It is similar to the propsed system in that semi-automatic means are used to produce the underlying annotations that the system semantically interprets user textual queries. However, Avatar is not a hybrid search system but rather a system for using keywords to search and query for semantics.

[LPM05] Vanessa Lopez, Michele Pasin, and Enrico Motta. AquaLog: An Ontology-Portable Question Answering System for the Semantic Web. In *Proceedings of 2nd European Semantic Web Conference (ESWC'05)*, pages 546–562, Heraklion, Crete, Greece, May/June 2005.

AquaLog is a natural language interface to performing semantic search and is not a hybrid search system.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, July 2008.

Book covering topics in modern information retrieval. Includes a discussion of the mean average precision metric we use to evaluate our system.

[Rin09] Antonio M. Rinaldi. An Ontology-Driven Approach for Semantic Information Retrieval on the Web. *ACM Transactions on Internet Technology*, 9(3):1–24, July 2009.

Uses the WordNet ontology and lexical chain processing to perform hybrid search. The system dynamically produces ontologies at run time but it is limited to lexical processing and cannot handle numerical constraints.

[TCL09] Aditya Telang, Sharma Chakravarthy, and Chengkai Li. Query-By-Keywords (QBK): Query Formulation Using Semantics and Feedback. In *Proceedings of the 28th International Conference on Conceptual Modeling (ER'09)*, pages 191–204, Gramado, Brazil, November 2009.

Translates user keyword queries into query templates that the user can fill out to obtain relevant results. This is a form of semantic search rather than hybrid search.

[TCRS07] Thanh Tran, Philipp Cimiano, Sebastian Rudolph, and Rudi Studer. Ontology-Based Interpretation of Keywords for Semantic Search. In *Proceedings of the 6th International and 2nd Asian Semantic Web Conference (ISWC/ASWC'07)*, pages 523–536, Busan, Korea, November 2007.

Translates user keyword queries into a ranked set of structured queries the user can choose and see results from. This is an example of a semantic search system rather than a hybrid search system.

[Vic06] Mark Vickers. Ontology-Based Free-From Query Processing for the Semantic Web. Master's thesis, Brigham Young University, Provo, UT, June 2006.

The semantic query component of the proposed system grows out of AskOntos query processing. We expand the system to allow for queries to cross multiple ontologies.

[WTL08] Haofen Wang, Thanh Tran, and Chang Liu. CE^2: Towards a Large Scale Hybrid Search Engine with Integrated Ranking Support. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM'08)*, pages 1323–1324, Napa Valley, California, October 2008.

Present an architecture for performing hybrid search with a reasonable runtime. We are also presenting a possible architecture to maintain reasonable runtimes although our architecture is specifically designed to allow for customization of extraction ontologies for future work.

[ZWX$^+$07] Qi Zhou, Chong Wang, Miao Xiong, Haofen Wang, and Yong Yu. SPARK: Adapting Keyword Query to Semantic Search. In *Proceedings of the 6th International and 2nd Asian Semantic Web Conference (ISWC/ASWC'07)*, pages 694–707, Busan, Korea, November 2007.

SPARK translates keyword queries into ranked SPARQL queries for the user to consider and obtain results from. SPARK is an example of a semantic search system rather than a hybrid search system.

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE PAPER

of a thesis proposal submitted by

Andrew Zitzelberger

This thesis proposal has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

Date	David W. Embley, Chair
Date	Stephen W. Liddle
Date	Tony R. Martinez
Date	Kent E. Seamons, Graduate Coordinator