Automatic pure anchor-based taxonomy generation from the world wide web.

Joseph Paul Elliott
University of Louisville

Follow this and additional works at: http://ir.library.louisville.edu/etd

Recommended Citation
https://doi.org/10.18297/etd/399
AUTOMATIC PURE ANCHOR-BASED TAXONOMY GENERATION FROM THE WORLD WIDE WEB

By
Joseph Paul Elliott
B.S., University of Louisville, 2003

A Thesis
Submitted to the Faculty of the
Graduate School of the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Master of Engineering

Department of Computer Engineering and Computer Science
University of Louisville
Louisville, Kentucky

May 2007
AUTOMATIC PURE ANCHOR-BASED TAXONOMY GENERATION FROM THE WORLD WIDE WEB

By
Joseph Paul Elliott
B.S., University of Louisville, 2003

A Thesis Approved on
May 2, 2007

By the following Thesis Committee:

__________________________
Dr. Antonio Badia, Thesis Director

__________________________
Dr. Olfä Nasraoui

__________________________
Dr. Suraj Alexander
ABSTRACT

AUTOMATIC PURE ANCHOR BASED TAXONOMY GENERATION FROM THE WORLD WIDE WEB

Joseph Paul Elliott

May 3, 2007

This thesis proposes a new method of automatic taxonomy generation using the link structure of Webpages. Taxonomy is a hierarchy of concepts where each child concept is said to be encompassed by its parent concept. Techniques have previously been developed to extract taxonomies from a traditional text corpus, but this thesis relies exclusively on the links between documents in the corpus, as opposed to the text of the corpus itself.

A series of algorithms were designed and implemented to realize the objectives of this thesis. These programs perform comparably to other techniques using the text in the documents and have shown that there is information available in the link structure of Webpages when creating concept taxonomies.
ACKNOWLEDGEMENTS

I would like to thank Dr. Antonio Badia for all of his guidance over the course of the year. Without his help this thesis would not have been possible. His knowledge and perspective helped make this thesis what it is. I would also like to thank Dr. Olfa Nasraoui for providing valuable suggestions and recommendations on how to improve this thesis.

Finally, I would like to thank my girlfriend, Anne, for putting up with me and making me countless frozen dinners while working on this thesis.
# TABLE OF CONTENTS

ABSTRACT....................................................................................................................... iii  
ACKNOWLEDGEMENTS............................................................................................... iv  
LIST OF FIGURES.......................................................................................................... vii  
LIST OF TABLES............................................................................................................. ix  

1. INTRODUCTION ...................................................................................................... 1  
   1.1. Background ........................................................................................................ 1  
   1.2. Organization of this Thesis ................................................................................. 4  

2. LITERATURE REVIEW ........................................................................................... 5  
   2.1. Bottleneck Minimization/Process Definition ...................................................... 5  
   2.2. Syntactic Analysis ............................................................................................... 7  
   2.3. Statistical Analysis ............................................................................................ 10  

3. THEORY AND DESIGN ......................................................................................... 12  
   3.1. Introduction ....................................................................................................... 12  
   3.2. Notation ............................................................................................................. 13  
   3.3. Creating the Relationships ................................................................................ 16  
   3.4. Analyzing the Relationship to Create the Term Hierarchy ......................... 24  

4. IMPLEMENTATION ............................................................................................... 33  
   4.1. SiteMap ............................................................................................................. 34  
   4.2. SiteGraph .......................................................................................................... 38  
   4.3. Miscellaneous tools ......................................................................................... 45  
   4.4. Implementation Issues ...................................................................................... 46  

5. EXPERIMENTAL RESULTS ................................................................................. 49
LIST OF FIGURES

Figure 1-1 Example term taxonomy ................................................................................................. 3

Figure 3-1 Overview of taxonomy creation ..................................................................................... 17

Figure 3-2 Three pages in an example website ................................................................................. 20

Figure 3-3 Example webpage from CNN.com (March, 2007) hosted in the Firefox web browser .......................................................................................................................... 28

Figure 4-1 Overview of Implementation ......................................................................................... 34

Figure 4-2 Screenshot of Sitemap as used to map websites .............................................................. 34

Figure 4-3 Screenshot of Sitemap used to insert data into the database ......................................... 37

Figure 4-4 Screenshot of the Word Graph tab of SiteGraph ........................................................................... 39

Figure 4-5 Screenshot of a taxonomy node context menu in SiteGraph ........................................ 42

Figure 4-6 Screenshot of the Stats dialog in SiteGraph ............................................................................ 42

Figure 4-7 Screenshot of the Hierarchy dialog in SiteGraph ................................................................. 43

Figure 4-8 Screenshot of the Word Frequency tab in SiteGraph ........................................................... 44

Figure 4-9 Screenshot of the Word Management tab in SiteGraph ...................................................... 45

Figure 5-1 Percent of Output that fulfills Subsumption Measure vs. Input parameters for Computer Hardware domain .................................................................................................................. 53

Figure 5-2 Average Generalization/Specialization quality vs. Input parameters for Computer Hardware domain ........................................................................................................................................... 54

Figure 5-3 Percent of Output that fulfills Subsumption Measure vs. Input parameters for Sports domain .............................................................................................................................................. 54

Figure 5-4 Average Generalization/Specialization quality vs. Input parameters for Sports domain .............................................................................................................................................. 55
Figure 5-5 Percent of Output that fulfills Subsumption Measure vs. Input parameters for News domain ................................................................. 56

Figure 5-6 Average Generalization/Specialization quality vs. Input parameters for News domain ........................................................................... 57

Figure 5-7 Screenshot of SiteJudge .......................................................................................................................... 58
LIST OF TABLES

Table 3-1 Example child counts before normalization..................................................... 23
Table 3-2 Example child counts after normalization........................................................ 24
Table 3-3 Example children demonstrating the use of the instance-cutoff parameter ..... 26
Table 3-4 Example children sorted by topic_rank............................................................ 28
Table 3-5 Co-occurrence example ..................................................................................... 30
Table 3-6 Crosstopic trimming example........................................................................... 32
Table 5-1 Output Measures for the taxonomies submitted for human judging ............ 58
Table 5-2 Average and standard deviation of relationship ratings by topic and totals..... 59
1. INTRODUCTION

1.1. Background

The World Wide Web is an ever changing, yet vast source of information. New websites are constantly appearing and old ones disappearing. Two pages may be connected via direct link today and not tomorrow. Also, web pages are created using a variety of technologies (HTML, CSS, JavaScript, Flash, etc.) and presented in as many human languages as one can name. However, in spite of the difficulty of extracting data from the Web, researchers continue to try, due to the abundant amount of available information on any conceivable subject.

One such set of attempts is the Semantic Web [13], an effort directed toward a more complete markup language that the creators of Webpages can use to identify pieces of data on their websites. Therefore, if items on a webpage are appropriately marked, software agents can know what kind of data a text string represents, instead of just displaying it to the user. The Semantic Web uses the Resource Description Framework (RDF), a markup language designed to describe objects, and to present information in an organized and consistent form. It also uses the Web Ontology Language (OWL) to describe objects, properties of those objects, and their relationships. Using both RDF and OWL web designers can describe the data in their Webpages and how it relates to both itself and other pieces of data in a form more easily manageable by software. This approach to extract data effectively and accurately from the internet puts the burden on
the designer to create their pages in a manner that can be easily parsed and understood and ties them to rigid standards in the way they encode their content.

Researchers have also attempted to automatically extract information from the World Wide Web by applying traditional Information Extraction techniques to the Web’s documents, Webpages [1,3,4,5,8]. For instance, in [3], PANKOW is developed to search for lexico-syntactic patterns on the internet using Google in order to discover concept relationships. Lexico-syntactic patterns were originally developed to be used on a traditional text corpus, but are now applied to the Web. Also, [4] discusses how to represent terms as vectors extracted from webpages and calculate statistical similarities between the vectors to determine concept relationships. Similar approaches, developed for Information Extraction from a standard text corpus, are being tried on the Web.

Due to the difficulty of creating taxonomies (from either a traditional text corpus or the World Wide Web), generally an ontology engineer, a human judge with topic specific knowledge, is required to validate, trim and add to the automatically generated ontology. This is known as the Knowledge Acquisition Bottleneck. Because of this bottleneck, heuristics that create more accurate ontologies are very valuable as they reduce, and may one day eliminate, the time required for the ontology engineer to review the created ontology.

This thesis outlines a new method of taxonomy construction built from data extracted from the link structure of the internet. The taxonomy created displays perceived relationships between terms, and is displayed as a rooted tree. That is to say, it is a graph with no cycles and has a designated root node. Each node represents a term and the edges from one node to the next represent a parent-child relationship. The
concept represented by the term in the child node should be more specific and subsumed by the concept represented by the parent node. The algorithms in this thesis also identify topic and instance nodes/terms. A *topic node* is more generic and is able to be expanded into more specific concepts. An *instance node* is specific and is a single occurrence of a particular topic node. An example taxonomy is shown in Figure 1-1.

![Figure 1-1 Example term taxonomy](image)

Currently there are no “perfect” taxonomy creation algorithms. Every algorithm has its strengths and drawbacks and use different pieces of information for generation. The heuristics proposed in this thesis are unique in that they use specifically the link structure of Webpages and the anchor text in links to automatically generate taxonomies.

The software developed in this thesis was able to create viable taxonomies from three different topic domains, SPORTS, COMPUTER HARDWARE, and NEWS. Each of these taxonomies were judged by humans and found to perform favorably based on developed statistical metrics that evaluate the consistency of a taxonomy. Additionally, while most existing techniques use the *entire document text*, this thesis only uses a fraction of the document that is limited to the *anchor text*. Therefore this thesis shows that there is valuable and currently ignored information inherent in the link structure of webpages for automatic taxonomy generation.
1.2. Organization of this Thesis

Chapter Two presents a detailed literature review of existing automatic taxonomy generation techniques. Chapter Three discusses the algorithms used to generate taxonomies from links in Webpages. Chapter Four presents the software developed for this thesis to implement the ideas in chapter two. Chapter Five presents the experiments designed to test the performance of our approach, then shows the results obtained, and discusses ways to improve the algorithms discussed in this thesis. Finally, Chapter 6 discusses the conclusions reached from performing the research in this thesis.
2. LITERATURE REVIEW

The Web is a very large source of potentially valuable, unorganized information. Because it is such an impressive source of information, a substantial effort has gone into extracting information and knowledge from, and organizing its contents.

2.1. Bottleneck Minimization/Process Definition

One set of efforts has been directed at creating concept hierarchies to act as a backbone for organizing other sets of information. These techniques generally suffer from a knowledge acquisition bottleneck, the process of a human knowledge engineer pruning a tree created from one of a variety of algorithms. Tools have been developed to lower the time and resources spent pruning these hierarchies [1,5,9].

[1] details a tool developed to compare a variety of clustering techniques in order to more easily determine which methods are the most effective in ontology building. The developed workbench provides methods for choosing what grammatical relations indicate a relationship between concepts and what pruning threshold to use. It uses a standard vector based distance measure to determine if two concepts are related.

The Mo’K workbench is an excellent tool for comparing how grammatical relationships and pruning parameters affect ontology building. It facilitates a deeper insight and understanding these parameters contribute to the final ontology.
The Mo'K workbench is similar to the SiteGraph tool developed in this thesis. They are both graphical utilities that allow the user to modify factors controlling the taxonomy and see how it affects the relationships therein.

[5] uses K-Means clustering and Latent Semantic Indexing to extract topics from a corpus. These techniques were combined into a software suite that proposes topics and relationships to an ontology engineer in a concise, manageable way who makes the final decisions about the topic relationships.

The authors outline an interface that the engineer could use to facilitate the process of pruning an existing ontology created using a variety of other approaches. It provides an impressive and complete interface to allow quick and easy ontology editing.

The Ontology Editor developed in this paper is similar to the SiteGraph application developed in this thesis. However, while they both provide graphical representations of the created taxonomies, Ontology Editor allows the user to delete relationships in order to refine the taxonomy. It is a tool for directly editing taxonomies while SiteGraph is a tool for viewing and manipulating taxonomies created by the ideas in this thesis.

[9] outlines a process for semi-automatic ontology construction. It identifies four steps for ontology learning used by their Ontology maintenance application, OntoEdit. The Import/Reuse step discusses the methods for which already defined ontologies can be merged with an existing one. Extraction deals with the actual ontology creation from a text corpus. Various standard ontology extraction techniques are discussed such as Hierarchical Concept Clustering and Lexical Entry Extraction. Hierarchical Concept Clustering uses the similarity of items to create a hierarchy by grouping those items
which are most similar. Similarity can be calculated using measures such as adjacency or syntactic relationships. Lexical Entry Extraction is the process of extracting N-grams from a document set based on statistical frequencies of co-occurrence. The third step, pruning an ontology, includes the balancing act of removing those concepts and relationships that are most likely invalid from the ontology. The final step is ontology refinement. During the refinement step the ontology pieces of the ontology are extracted and fine tuned for use with a specific application. The authors provide a good overview of various techniques used in Information Extraction and Ontology Building. They also suggest a solid, robust framework for creating an ontology building system and addresses the challenges and benefits to automatic ontology creation. However, it does not significantly add to the study of ontology building and only serves as summary or introductory work. This paper discusses very broad ideas regarding an entire life cycle of ontology maintenance. However, the ideas in this thesis are a very specific method that would fall under the single step, Extraction, discussed in this paper.

2.2. **Syntactic Analysis**

Many techniques have been used to create the taxonomies to be evaluated by a knowledge engineer including traditional knowledge extraction techniques such as collocation measures and syntactic pattern matching to the large text corpus that the internet provides [3, 4, 7]. These measures have enjoyed varying degrees of success, but still require human interaction to create viable term hierarchies.

[3] discusses a method of using Google to expand a text corpus to find instance of and subclass relationships. Previous attempts to use text patterns to find these
relationships suffered from a lack of text to search. The method takes a given Instance <I> and Concept <C>, and searches for the Hearst Patterns [14], e.g. <C>s such as <I> and such <C>s as <I>. It also uses Definite, Apposition and Copula patterns. Each of these pattern sets exploits commonly used syntactic patterns to extract term relationships. Using this method a given instance can be compared with a large number of concepts to see which concept it matches up with most successfully.

Although this method provides impressively reliable results it requires a rather narrow set of instances and concepts to be effective. Since every combination of every concept and instance and phrase must be searched this technique would require a large amount of time and internet resources to process even a small set of data. If only 10 concepts and 100 instances and all 8 mentioned patterns are used it would require 10 * 100 * 8 = 8000 Google searches on a very small set of data. Another weakness to this approach is that it requires a set of concepts and instances to start with as opposed to generating its own from the corpus. For this reason it would be a powerful, albeit expensive, pruning algorithm as opposed to an ontology builder itself.

These methods would be a great addition to other ontology building algorithms because it can reinforce or refute an existing set of relationships. For instance, if these methods were to be integrated it into the ideas in this thesis, they could be used to check all created relationships to see if they appear a significant number of times on the internet. This would be a very powerful pruning step if the resources were available.

[7] proposes a new method for extracting concepts and relationships from text documents and compares it to another approach employed by Text-To-Onto. Text-To-Onto is another automatic taxonomy generation program which uses the tf/idf measure to
compute similarity. *tf/idf* (term frequency–inverse document frequency) is a measure used in information retrieval to determine the importance of a word by using its frequency of appearances. CRCTOL (Concept Relation Concept Tuple based Ontology Learning) is a system that given a parsed and tagged text input and a domain lexicon creates an ontology. This system uses TIM-DRM scores to create an initial list of terms and then uses a second process to pull commonly used single terms out of multi-term phrases found using TIM-DRM. TIM-DRM is a statistical measure, which uses *tf* and *idf* and adds syntactic relationships, to create a more meaningful weighting parameter.

Semantic Relations are extracted using syntactical relations of the form `<Noun, Verb, Noun>`. The two nouns have a relationship through the verb.

In a given domain this technique outperformed the Text-To-Onto in both concept and relationship extraction using the measures of recall and precision.

The techniques discussed in this paper are similar to the ideas in this thesis in that they both automatically attempt to generate taxonomies. However, CRCTOL is based on syntactic analysis of a set of documents while this thesis only uses the link structure of a set Webpages to determine concept relationships.

[6] generalized to the case of digital libraries, the syntactic hierarchy based similarity originally proposed in [19] to compare URLs for the purpose of clustering user sessions. The idea behind this similarity is that URLs are considered closer if they share in common a larger proportion of the path from the root of the hierarchy to each of the URLs.
2.3. Statistical Analysis

Clustering algorithms have also been applied to sets of Webpages in order to measure similarity and extract hierarchies [2, 8]. These clusters have also been compared to a site’s link structure as a measure of the cohesiveness and organization of the site [10].

[10] attempts to use page clustering and link structure to make suggestions on how to improve a website. The authors suggest a method of slowly increasing the number of clusters to create a tree of which the root node contains all of the pages and each level groups the pages into an increasing number of clusters. The number of clusters to be used in this process is determined experimentally for each site. Then, using usage information, the most frequently visited clusters are identified. The authors then conclude that for “good” website design, clusters of pages should be connected together and those most visited clusters should have links from the main page. While this approach does use the link structure of the site to determine information about the site, it does not attempt to build taxonomies of terms from this structure as does this thesis. However, the paper uses this information, compared to usage statistics and page similarity measures to try to determine if a site is organized well.

[8] gives an overview of the current state and accepted practices in retrieving taxonomies from the web and then proposes a combination of syntactic and statistical methods for creating taxonomic relations. It states that the solution to more intelligent and functional web services is semantic descriptions of objects that will facilitate a “new level of Web Intelligence”. The paper concludes that automatic ontology generation and object classification is the key to the Semantic Web and, therefore, research in this field is valuable. Unfortunately, due to the complexity of information available on the web
this is also a very difficult task. The author begins by classifying automatic ontology
generation into two main categories. The first is a symbolic approach which relies on
matching lexico-syntactic patterns to create term relationships. While this technique is
powerful it suffers from being language dependent and not as scalable as its alternative, a
statistic based approach. The statistics based clustering approach uses a similarity
measure and a computation strategy to create clusters of similar words. It has the
strength of being scalable, however, it suffers because it doesn’t make use of language
constructions which have been shown to be very powerful. The author then goes on to
propose a new method combining the advantages of the statistical and symbolic
approaches. Given an existing taxonomy and a new word to insert, the authors propose
considering both the words semantic and statistic relationships to the words already in the
hierarchy. Although, it was concluded that this method is not statistically advantageous
over the previous methods, it was expected that further study of combined approaches
will yield better results.

The ideas proposed in this thesis could be considered a combination of syntactic
and symbolic approaches as well. However, instead of using lexico-syntactic patterns to
create relationships between words, the actual link structure of the web pages themselves
is used. Also, collocation statistics by appearance in links is used to determine some
relationships between sibling terms.
3. THEORY AND DESIGN

3.1. Introduction

This thesis focuses on using only the link structure and anchor text in a set of domain specific sites in order to extract a hierarchy. It does not propose to create complete or perfect hierarchies, but rather to show that the largely ignored link structure of a site contains valuable information for taxonomy building. Generally website designers attempt to organize their sites for easy human access and navigation to the topics in their site. Therefore, there is information inherent in the way the pages in a site are linked and by examining the link structure, anchor text and URL itself, a taxonomy can be created. Using this information only, this thesis proposes methods to map a site and organize the pages from “most generic” to “most specific”. Then the relationships in this tree are examined to create a taxonomic tree.

The methods discussed in this thesis do not attempt to create complete taxonomies from the topic domains in question. These methods do, however, show that using the link structure of sites is a valuable addition to the variety of techniques already employed in automatic taxonomy creation. Two applications were developed, SiteGraph and SiteMap, in order to create taxonomies using the ideas discussed in this thesis. These applications generated taxonomies from three different topic domains, NEWS, SPORTS, and COMPUTER HARDWARE, that were evaluated using several different measures including human judges. Theses taxonomies compared favorably with taxonomies generated using other means. Therefore, this thesis shows that the information contained
in the link structure of webpages can and should be used in automatic taxonomy generation.

### 3.2. Notation

#### Terms

A term, defined as a string of characters, will be referred to with a lowercase $t$ or variant thereof ($t'$ for instance).

#### Link

A link, defined as the `<a>` tag in an HTML document, will be referred to with a lowercase $l$ or variant thereof ($l'$ for instance). A link has the following properties:

| $l.text$ | The text, or anchor text, is the set of terms, $t$, that falls between the opening `<a>` tag and the closing `</a>` tag. The text is filtered according to a procedure to be described later. |
| --- |
| $\{l_1, l_2, l_3 \ldots l_n\}$ | Given a set of links $[]$ returns the set of terms from the text of each link. $\{l_1, l_2, l_3 \ldots l_n\} = l_1.text \cup l_2.text \cup l_3.text \cup \ldots \cup l_n.text$ |

#### Webpage

A Webpage, defined as a single document as shown by a normal web browser, will be referred to with a lowercase $p$ or variant thereof ($p'$ for instance). A page has the following properties:

| $p.url$ | The URL for this web page. |
| --- |
| $p.domain$ | The fully qualified domain name from the URL. For instance, the domain of the URL http://www.google.com/index.html is www.google.com. |
| $p.depthrating$ | The depth rating is a natural number between 0 and 1 which gives a relative indication of how general or specific p’s contents are. See Section 3 for a specific description of how a depth rating is calculated. |
| $p.linksout$ | The set of links contained in webpage p. |
The D function takes two pages and returns the minimum link distance between the two pages. Since this is a real metric, it has the following properties:

\[
D(p_1, p_2) = 0
\]

\[
D(p_1, p_2) = D(p_2, p_1)
\]

\[
D(p_1, p_2) \leq D(p_1, p_3) + D(p_3, p_2)
\]

The distance from a page to itself is always zero. The direction of the links does not affect distance and the distance from page to another is always the minimum link distance.

**WebSite**

A website is defined as a set of Webpages that are a certain link distance from the “root” page and belong to the same domain. A website will be referred to with a lowercase \( w \) or variant thereof (\( w' \) for example). The set of all websites used in an algorithm will be represented by a capital \( W \).

<table>
<thead>
<tr>
<th>( \text{w.root} )</th>
<th>The “root” page of the website. This is not necessarily the “home” page or what might be considered the root page by a human viewer of the website. See Section 3 for more information on how the root pages are obtained.</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{w.pages} )</td>
<td>The set of pages whose domain match ( \text{w.root} ) and are reachable within ( \delta ) links or less. Formally, ( \text{w.pages}(\delta) = { p</td>
</tr>
<tr>
<td>( \text{pages}(W) )</td>
<td>The ( \text{pages()} ) function takes a set of Websites, ( W ), and returns the union of their ( \text{pages} ) set. Formally, ( \text{pages}(w) = w_1.\text{pages}(\delta) \cup w_2.\text{pages}(\delta) \cup ... \cup w_n.\text{pages}(\delta) )</td>
</tr>
</tbody>
</table>

**Notation Refinement**

Now that all of our concepts are defined we can expand the definition of the Link and Webpage.

A Link, \( l \), also has the following characteristics:

| \( l.to \) | The webpage pointed to by the link indicated by the contents of the \text{href} attribute. |
A Webpage, $p$, also has the following characteristics:

<table>
<thead>
<tr>
<th>$p$</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p$.website</td>
<td>The website that this page belongs to. Note that the website’s domain matches $p$.domain. Formally, $p$.website $= w : w.root.domain = p$.domain</td>
</tr>
<tr>
<td>$p$.pagesto</td>
<td>The set of all pages that the links in $p$ point to and that belong in the same website as $p$. Formally, $p$.pagesto $= { p'</td>
</tr>
<tr>
<td>$p$.linksin</td>
<td>The set of all links pointing to $p$ by other pages in the same website as $p$. Formally, $p$.linksin $= { l</td>
</tr>
<tr>
<td>$p$.pagesfrom</td>
<td>The set of all pages that contain links that point to $p$ and belong to the same website as $p$. $p$.pagesfrom $= { p'</td>
</tr>
<tr>
<td>$p$.terms</td>
<td>The set of terms used in anchor text for anchors that link to $p$. Formally, $p$.terms $= { t</td>
</tr>
<tr>
<td>count($p$, $t$)</td>
<td>Given a page $p$, and a term $t$, this is the total number of times $t$ appears in $p$.linksin. This considers the text in anchors only and not the rest of the page content.</td>
</tr>
</tbody>
</table>

**Relationships**

The terms from the anchor text and link structure of the Website are used to create a relationship with the following properties:

A relationship is a set of tuples with three attributes:

$(parent-term, child-term, count)$

<table>
<thead>
<tr>
<th>$parent-term$</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$child-term$</td>
<td>The child term or more specific term in the taxonomy. There is only one tuple for each child-term and parent-term combination. If a tuple appears then the following is true about the parent-term and child-term: $\exists p\exists p' : p \in W \land p' \in W \land p' \in p$.pagesto $\land r.parentterm \in p$.terms $\land r.childterm \in p'$.terms</td>
</tr>
</tbody>
</table>

15
**count**

A natural number from 0 to infinity indicating the number of times the parent-term and child-term appear in the above relationship. The contribution of the pages is normalized by the number of terms in the page.

**parent_term_support**

Given a tuple \( r \) and a relationship \( R \) where \( r \in R \), the parent_term_support is a number between 0 and 1 that represents the percentage of all tuples \( r' \) with the same parent-term as \( r \), \( r\.parent-term \). Formally,

\[
parent_term_support(r, R) = \frac{r\.count}{\sum_{r' \in R | r'.parentterm = r\.parentterm} r'\.count}
\]

In other words, this is the tuple’s count divided by the sum of the counts of the tuples where the parent term matches \( r\.parent-term \).

**add**

Given a tuple \( r \) and a relationship \( R \), if there already exists a tuple in \( R \) with a matching parent-term and child-term then \( r\.count \) is added to the existing tuple. Otherwise \( r \) is added to the relationship \( R \). Formally,

\[
\text{add}(r, R): \begin{cases} 
\text{if } \exists r': r' \in R \\
\land r'.parentterm = r\.parentterm \\
\land r'.childterm = r\.childterm \\
r'.count += r\.count \\
\text{else } R = R \cup \{r\}
\end{cases}
\]

### 3.3. Creating the Relationships

All websites that contain a significant amount of content require a link structure to allow navigation from the highest level documents, the site’s root or home page, to the lowest level, topic specific documents. For example, a site with information about the C programming language may have a home page, topic pages about keywords, function libraries, etc. and then specific pages detailing one keyword, one function etc. Organizing the page from most generic to the most specific sets up the tree from which the term hierarchy is eventually derived. This functionality is encapsulated by the SiteMap application discussed in Section 4.1.

The process for developing the taxonomy is shown in Figure 3-1.
3.3.1. Crawling the Pages

First a set of websites that define a certain domain are chosen and downloaded. The set of pages retrieved by the crawler from each WebSite is formally defined as:

\[ w \text{.pages}(\delta) = \{ p \mid D(p, w \text{.root}) \leq \delta \land p \text{.domain} = w \text{.root.domain} \} \]

Recall that the function \( D() \) is defined in Section 3.2 as being the shortest link distance between two pages. The value \( \delta \) is set experimentally.

A crawler is used to navigate each site and store information regarding the links, pages, and anchor text. For each webpage, all of the incoming and outgoing links with their anchor text is recorded. In order to limit the amount of a website the crawler explores and to limit the amount of time it ran, several rules are employed, as explained below.

Rule 1: The crawler limits itself only to the domain of the root webpage. In an attempt to keep the crawler obtaining pages in the domain we are interested in creating a hierarchy for, the simplest solution is to keep it inside of the website it is traversing.
Rule 2: In order to keep the crawler from spending too much time on a single website, only pages found in the first $\delta$ levels of the link structure from the root are used. Because the sites used tended to be very large, crawling a single one would take too long to allow for efficient use of time. Also, even though only $\delta$ levels are used, most sites still return thousands of Webpages.

Rule 3: To keep the crawler focused on pages that contain domain specific content and away from pages that are specific to site structure or maintenance, links with the following text in the URL or anchor are not followed:

- forum
- site
- newsletter
- search
- archive
- blog
- rss

After we have mapped each website, the crawled Webpages are organized from most general to most specific in a hierarchy. The webpage hierarchy will in turn be used to create the term hierarchy as described below.

### 3.3.2. Organizing the Pages

Next, we will reorganize the graph of Webpages created by the site’s link structure into a tree. This will require an algorithm that will attempt to organize the pages by content (from most general at the root to most specific at the leaves). After all of the pages are downloaded and stored, the first step is to calculate a depth rating for each page which gives an indication of how deep in the tree it should appear. The depth rating for each page is calculated as follows:

\[
p_.depthrating = \alpha \* \left( \frac{p.url.length}{\max_{p'\in_p.website.pages} \ p'.length} \right) + (1 - \alpha) \* \left( 1 - \left( \frac{|p.links|}{\max_{p'\in_p.website.pages} \ |p'.links|} \right) \right)
\]
where length is a function that returns the number of characters in the URL. In other words, the above formula returns the sum of the length of the page’s URL divided by the longest URL in the website and the page’s number of incoming links divided by the number of incoming links of the page with the most incoming links. These two factors that make up the depth rating are also weighted using the parameter $\alpha$ whose value is determined experimentally.

Then, after the depth rating for each page is calculated, the pages are organized (using the depth rating) in a tree structure where each node represents one Webpage. We start with the node that represents the root of the Website. All of the Webpages that the root Webpage links to are added as children of the root node in the tree. Then the tree is traversed and every Webpage node in the tree that doesn’t yet have children has the Webpages it links to added as children in the tree. This process is repeated until all of the Webpages in the site end up in the tree.

We note, however, that a webpage can only exist once in the tree. Therefore, when a webpage is reached that already exists in the tree, the algorithm uses the depth rating to see if it should move the webpage node or leave it where it is. The page is moved only if two conditions are satisfied. First, the page has to be moving farther down the tree, towards the leaves, away from the root. Second, the depth rating of the webpage to be moved has to be significantly larger than the depth rating of the webpage node it would be moved under.

For example, consider a Website about Computer Hardware which contains three Webpages as shown in Figure 3-2. The example below is contrived and the URLs involved are not known to exist and should not be followed:
First, we need to calculate the depth rating of each page.

\[
\begin{align*}
\text{max}(|\text{w.pages.linksin}|) &= 20 \\
\text{max}(\text{w.pages.url.length}) &= 38 \\
p1.\text{ deeprating} &= .6 \times \frac{20}{38} + .4 \times (1 - \frac{20}{20}) = .6 \times .526 + .4 \times 0 = .316 \\
p2.\text{ deeprating} &= .6 \times \frac{31}{38} + .4 \times (1 - \frac{15}{20}) = .6 \times .816 + .4 \times .25 = .589 \\
p3.\text{ deeprating} &= .6 \times \frac{38}{38} + .4 \times (1 - \frac{4}{20}) = .6 \times 1 + .4 \times .8 = .920
\end{align*}
\]

Next we need the standard deviation of the depth ratings:

\[\text{DepthStdDev} = .303\]
Now we will execute the logic in the tree building algorithm designed to build a tree from the most general webpage to the most specific webpage:

Step 1:
\[
Q = \{p_1\} \\
T = \{p_1\}
\]
for each Webpage \( p_{\text{Child}} \) in \( p_1 \)
- \( p_1 \) does not exist in the tree
  - therefore it is made a child of \( p_1 \) and placed in the queue
- \( p_2 \) does not exist in the tree
  - therefore it is made a child of \( p_1 \) and placed in the queue

Step 2:
\[
Q = \{p_2, p_3\} \\
T = \{p_2, p_3\}
\]
for each Webpage \( p_{\text{Child}} \) in \( p_2 \)
- \( p_2 \) does exist in the tree
  - \( p_2 \).depthrating + DepthStdDev < \( p_1 \).depthrating
    - \( .589 + .303 < .316 \) (FALSE)
- \( p_3 \) does exist in the tree
  - \( p_3 \).depthrating + DepthStdDev < \( p_1 \).depthrating
  - \( .920 + .303 < .920 \) (TRUE)
  - therefore it is removed from \( p_1 \) and made a child of \( p_2 \)

Step 3:
\[
Q = \{p_1\} \\
T = \{p_1\}
\]
for each Webpage \( p_{\text{Child}} \) in \( p_1 \)
- \( p_1 \) does exist in the tree
  - \( p_1 \).depthrating + DepthStdDev < \( p_1 \).depthrating
    - \( .920 + .303 < .316 \) (FALSE)
- \( p_2 \) does exist in the tree
  - \( p_2 \).depthrating + DepthStdDev < \( p_1 \).depthrating
  - \( .920 + .589 < .589 \) (FALSE)

Completed Tree:
Due to the length of the url and the number of incoming links, the webpage about Radeon brand video cards is moved down under the webpage about video cards in general. Although this example is simplistic, it demonstrates how the depth rating is calculated and used to organize the Webpages of websites from most general to most specific.

If, however, \( p_3 \) had a larger than expected number of incoming links, it would be feasible that \( p_3 \) would not be moved down below \( p_2 \). For example, if \( p_3 \) had 40 incoming links:

\[
\begin{align*}
\text{max}(|w\text{.pages.linksin}|) &= 40 \\
\text{max}(w\text{.pages.url.length}) &= 38 \\
p_1\text{.depthrating} &= .6 \times (20 / 38) + .4 \times (1 - (20 / 40)) = .6 \times .526 + .4 \times .5 = .516 \\
p_2\text{.depthrating} &= .6 \times (31 / 38) + .4 \times (1 - (15 / 40)) = .6 \times .816 + .4 \times .625 = .739 \\
p_3\text{.depthrating} &= .6 \times (38 / 38) + .4 \times (1 - (40 / 40)) = .6 \times 1 + .4 \times 0 = .600 \\
\text{DepthStdDev} &= .113
\end{align*}
\]

In this case, even though \( p_3 \) should be under \( p_2 \), the situation would be reversed and \( p_2 \) would be under \( p_3 \) because the \text{DepthStdDev} is less than the difference in their \text{depthratings}. However, in this case, it requires that \( p_3 \) have nearly three times the number of incoming links which would be unlikely if the content in \( p_3 \) truly was more specific.

3.3.3. Calculating Term Relationships

The final step in creating the term relationships is to process the tree created in the above step and, using the Webpage \text{terms} set (defined in Section 3.2), create a set of
relationships. The terms are the actual strings of characters representing a word found in an anchor link referring to this webpage.

The tree is traversed and for each parent-child connection in the tree, a set of tuples is generated to be added to the relationship. A tuple is created for every combination of terms in the parent webpage’s terms set and child webpages terms set. These sets contain every term in any anchor text that is pointing to each page. Therefore, if a parent page has the terms \{computer, pc\} and has two child webpages with the term sets \{motherboard, memory\} and \{sound, card\} then tuples are created with the following parent and child terms:

- computer-motherboard
- computer-memory
- computer-sound
- computer-card
- pc-motherboard
- pc-memory
- pc-sound
- pc-card

As the terms are added, they are normalized so that the contribution of each child webpage was weighed equally. This simply means that the count of each term was divided by the total number of terms that refer to this webpage. For instance, if a treenode has two child treenodes with pages that contain the list of terms and counts in Table 3-1,

<table>
<thead>
<tr>
<th>Child Webpage 1</th>
<th>Child Webpage 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>child.term</td>
<td>child.count</td>
</tr>
<tr>
<td>motherboard</td>
<td>24</td>
</tr>
<tr>
<td>memory</td>
<td>10</td>
</tr>
<tr>
<td>child.term</td>
<td>child.count</td>
</tr>
<tr>
<td>sound</td>
<td>3</td>
</tr>
<tr>
<td>card</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 3-1 Example child counts before normalization

then the first webpage would contribute significantly more than the second webpage simply because it had more links referencing it. Therefore, we normalize the contribution by dividing the first Webpage’s counts by \(24 + 10 = 34\) and the second Webpage’s counts by \(3 + 1 = 4\). This yields:
Therefore, both child Webpages contribute equally to the terms of the webpage of the current treenode.

When creating a relationship (set of tuples described in Section 3.2), each website contributes using the above process. The pages in the site are first organized using the depth rating, and then the tree is traversed and has the relationships added using the process described in this section. When a website generates a tuple, it is added to the relationship using the \( \text{add}(r, R) \) described in Section 3.2. Therefore, webpages contribute to the overall taxonomy by adding individual tuples to the relationship. If two webpages generate the same tuple, then it is strengthened by adding the counts together. In this fashion, the relationship is created from all of the contributing websites to be analyzed in the following sections.

### 3.4. Analyzing the Relationship to Create the Term Hierarchy

Now that the relationships have been created, the next step is to create the term hierarchy. A variety of parameters that directly influence its size and shape have to be adjusted to provide meaningful term hierarchies. These parameters control tree depth, trimming, allowed terms, the root term, and a large number of other factors. All of these parameters and this functionality are encapsulated in the SiteGraph application discussed in Section 4.2.

<table>
<thead>
<tr>
<th>Child Webpage 1</th>
<th>Child Webpage 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>motherboard</td>
<td>sound</td>
</tr>
<tr>
<td>0.705882</td>
<td>0.75</td>
</tr>
<tr>
<td>memory</td>
<td>card</td>
</tr>
<tr>
<td>0.294118</td>
<td>0.25</td>
</tr>
</tbody>
</table>

Table 3-2 Example child counts after normalization
3.4.1. Terminology

term_anchor_support(t): The term anchor support is simply the proportion of all pages that a given term appears in (anchor text only). Given a term $t$:

$$\left| \left\{ p \mid p \in \text{pages}(W) \land t \in [p.linksin] \right\} \right| \left\| \text{pages}(W) \right\|$$

Note that a capital $W$ is used indicating the set of all pages of all websites crawled to create the relationship $R$.

**Instance Terms:** Instance terms (sometimes referred to as instance children when a specific child/parent relationship is being discussed) are terms that use only a relationship percentage value compared to a cutoff value to determine if they are valid. *An instance term intends to represent a specific instance of a broader subject.* For example, in the SPORTS domain a player’s name such as “BRYANT” could be considered an instance term.

**Topic Terms:** Topic terms (sometimes referred to as topic children when a specific child/parent relationship is being discussed) are terms that use both a relationship percentage value and the term_anchor_support as measures to determine validity. *A topic term intends to represent a subject that can be broken down into smaller instances.* For instance, in the SPORTS domain a sport such as “BASKETBALL” could be considered a topic term.

**Modifier Terms:** Referring to terms as “modifiers” simply indicates that they co-occur more frequently in the same link’s anchor text than a given threshold. They are searched for and displayed in results simply to give the user more insight into the relationships between terms. For example, in the COMPUTER HARDWARE domain the term “SOUND” could be considered a modifier of the term “CARD”.

25
**topic_rank:** The *topic rank* of a term *t*, given a parent *p*, is the *term_anchor_support* plus the *parent_term_support* (see Section 3.2) where *t* is the child-term and *p* is the parent-term. It is used in generating topic terms. Formally,

\[
\text{topic_rank}(p, t) = \text{term_anchor_support}(t) + \text{parent_term_support}(R, r)
\]

where *R* is the relationship created from the webpages and *r* is the tuple containing *t* as the child-term and *p* as the parent-term.

### 3.4.2. Input Parameters

**root-term** - This is the term chosen to be the root of the hierarchy.

**max-levels** - The maximum number of levels keeps the algorithm from continuing infinitely. Generally, the other trimming parameters will keep the hierarchy from reaching the maximum number of levels. However, this value is included as a fail safe.

**instance-cutoff** - The instance-cutoff is used when searching for child instance terms for a parent term. They are generated by comparing the *instance-cutoff* value with the *parent_term_support* value. For instance, if the term “CARD” appears as the parent-term in the tuples in *R*, listed in Table 3-3.

<table>
<thead>
<tr>
<th>parent-term</th>
<th>child-term</th>
<th>count</th>
<th>parent_term_support</th>
</tr>
</thead>
<tbody>
<tr>
<td>CARD</td>
<td>VIDEO</td>
<td>0.52</td>
<td>0.07</td>
</tr>
<tr>
<td>CARD</td>
<td>AUDIO</td>
<td>2.88</td>
<td>0.41</td>
</tr>
<tr>
<td>CARD</td>
<td>RADEON</td>
<td>1.77</td>
<td>0.25</td>
</tr>
<tr>
<td>CARD</td>
<td>GEFORCE</td>
<td>0.31</td>
<td>0.04</td>
</tr>
<tr>
<td>CARD</td>
<td>PCI</td>
<td>1.5</td>
<td>0.21</td>
</tr>
</tbody>
</table>

Table 3-3 Example children demonstrating the use of the instance-cutoff parameter

and the *topic-cutoff* is 0.2 then PCI, AUDIO, and RADEON would be retrieved as child instance terms.
**top-topic-percent** - This parameter controls what percent of all potential topics actually become topics. It defaults to ten percent. See below for more information.

**topic-cutoff** - The topic cutoff is used when searching for child topic terms for a parent term. They are generated by comparing the topic-cutoff value with the `parent_term_support` value. The values are compared exactly like the `instance-cutoff` value, except the relationships are also ordered by `topic_rank` and only the `top-topic-percent` of terms are returned. Because only the `top-topic-percent` of candidate topics are retained the `topic-cutoff` tends to be an order of magnitude less than the `instance-cutoff`.

Sorting the pages in this manner causes those terms appearing in the largest percentage of pages to be favored for topics. This method was used (and works) because the links that tend to contain “topic” terms are those that appear as navigational links on every page in a site. For instance, when navigating http://www.cnn.com (as of March, 2007) there is a blue bar appearing at the top of every page allowing instant navigation to the main topics of the site: World, U.S., Business, Sports, etc. Refer to figure 3-3.
Because this design structure is used so frequently, the term\_anchor\_support of a term is a very powerful indicator of whether or not this term is a good topic. Returning to our example, Table 3-4 shows the child terms with the term\_anchor\_support added and sorted by a topic\_rank:

<table>
<thead>
<tr>
<th>parent-term</th>
<th>child-term</th>
<th>count</th>
<th>parent_term_support</th>
<th>term_anchor_support</th>
<th>topic_rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>CARD</td>
<td>AUDIO</td>
<td>2.88</td>
<td>0.41</td>
<td>0.48</td>
<td>0.89</td>
</tr>
<tr>
<td>CARD</td>
<td>VIDEO</td>
<td>0.52</td>
<td>0.07</td>
<td>0.52</td>
<td>0.59</td>
</tr>
<tr>
<td>CARD</td>
<td>PCI</td>
<td>1.5</td>
<td>0.21</td>
<td>0.23</td>
<td>0.44</td>
</tr>
<tr>
<td>CARD</td>
<td>RADEON</td>
<td>1.77</td>
<td>0.25</td>
<td>0.12</td>
<td>0.37</td>
</tr>
<tr>
<td>CARD</td>
<td>GEFORCE</td>
<td>0.31</td>
<td>0.04</td>
<td>0.09</td>
<td>0.13</td>
</tr>
</tbody>
</table>

Table 3-4 Example children sorted by topic\_rank

Using this methodology AUDIO and VIDEO are the two most likely candidates for topic children of the parent term “CARD” based on the topic\_rank.

**topic\_trim\_cutoff** - The algorithm first attempts to generate topic children for a given term. If, however, during the trimming phase a greater percentage of topics are
removed than the \textit{topic-trim-cutoff}, then all of the topic children are removed and are replaced with instance children using the above methodologies.

\textbf{modifier-cutoff} - This parameter determines what percentage of a term’s total appearances have to co-occur with another term for it to be considered a modifier term. In this case, co-occurrence is defined as appearing in the same anchor text. Creating modifier terms is part of the trimming phase and is designed to reveal more information about the relationships between sibling terms.

Using the above input parameters, the taxonomy is created by building one level at a time and then trimming that level until either \textit{max-levels} is reached or all of the leaf nodes are instance nodes (See Section 3.4.4 to find how instance nodes are created).

\subsection*{3.4.3. Building a Level}

We start by adding the \textit{root-term} as the root node of the tree and a Topic node. Then we build each level by traversing the tree and creating child topic nodes using the \textit{topic-trim-cutoff}. After each level is built it is trimmed using the below process. The trimming process determines if a Topic node makes a significant contribution to the taxonomy. If it does not then the Topic children are removed and replaced with Instance children. Therefore, because the building process does not build off of instance children, the taxonomy creation process will eventually end when there are no more topic leaf nodes.

\subsection*{3.4.4. Trimming a Level}

After a level is built, the algorithm uses a series of trimming rules that remove and move TreeNodes. There are three basic trimming rules that are applied to each level.

\textbf{Topic Trimming}
The first kind of trimming removes topic children, and, if certain conditions are met, it removes all topic children of a node and replaces them with instance children. The hierarchy does not build off of instance children so if a node’s child topic terms are replaced with child instance terms, then that will be the last level of the tree on this branch.

First, the trimming logic checks how many unique topic children a node brings into the tree. If a node fails to bring any new topics to the entire tree, it has its child topic nodes removed. Next, if the topic node provides at least one new topic to the tree, it then removes all topics that already appear in a higher level. If it removes a larger percentage than the topic-trim-cutoff parameter, or only has one topic child left then it will have all of its topic children removed. Then, all of those nodes who had their Topic children removed, have them replaced with Instance children. Instance children are generated as described above.

**Modifier Trimming**

The next kind of trimming, identifies “modifier” nodes among sibling terms and decides if one should be moved over as a modifier of another. Each node on the newly created level is checked against each of its sibling terms to see if its co-occurrence in anchor text with every other term divided by its total appearances is greater than the modifier-cutoff parameter. This is performed for both instance and topic nodes. If this is true then the term is moved over as a “modifier” of the other term.

<table>
<thead>
<tr>
<th></th>
<th>Appearances</th>
</tr>
</thead>
<tbody>
<tr>
<td>power</td>
<td>200</td>
</tr>
<tr>
<td>supply</td>
<td>100</td>
</tr>
<tr>
<td>co-occurrence:</td>
<td>80</td>
</tr>
</tbody>
</table>

Table 3-5 Co-occurrence example
In the example, shown in Table 3-4, supposing the modifier-cutoff is set to .5, then supply would be considered a modifier term of power because 80% of the times supply appeared in anchor text it appeared with power. However, power would not be considered a modifier of supply because only 40% of its appearances in links co-occurred with supply. This is equal to the conditional probability of the terms. If,

\[ p(A | B) > \text{ModifierCutoff} \]

is true then B is considered a modifier of A.

**CrossTopic Trimming**

Sometimes a child term will appear across so many topic terms, that expanding it further would cause huge redundancies in the tree. Therefore terms that appear across a sufficient number of topic nodes will be considered “cross topic”. In order to find cross topics nodes, a given term’s appearances on a given level is compared to the average appearances by the other nodes. If a term is two standard deviations away from the average term appearance at that level it is to be considered a cross topic node because only those terms that were true outliers should be moved up as a cross topic node. During experimentation it was found that a single standard deviation was not restrictive enough and too many nodes were considered cross topic.

For instance, in the tree cross topics are listed in Table 3-6 marked with orange:

<table>
<thead>
<tr>
<th>Level</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>CARD</td>
<td>VIDEO</td>
<td>SOUND</td>
<td>2</td>
</tr>
<tr>
<td>SOUND</td>
<td>USB</td>
<td>ABIT</td>
<td>3</td>
</tr>
<tr>
<td>USB</td>
<td>PCI</td>
<td>NIKON</td>
<td>1</td>
</tr>
<tr>
<td>PCI</td>
<td>MEMORY</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>HARDWARE</td>
<td>MOTHERBOARD</td>
<td>ABIT</td>
<td>PCI</td>
</tr>
<tr>
<td>USB</td>
<td>DIGITAL</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
If we are trimming the third level, then USB would be considered a cross topic term. This is because it appears three times in the level which is greater than the average appearance at that level plus two times the standard deviation.
4. IMPLEMENTATION

A series of programs were developed to implement the ideas described in this thesis regarding taxonomy building and information extraction. These programs were developed in C# using Visual Studio .NET 2005 with SQL Server as a back end to store the recovered data. Several externally developed tools were also used such as the Badger Information Extraction Software (http://www-nlp.cs.umass.edu/software/badger.html) developed by the Center for Intelligent Information Retrieval at the University of Massachusetts.

The first program, SiteMap, downloads and maps a list of websites and inserts them into an SQL Server database. It is responsible for accurately maintaining the links between all pages in a site, and tracking what words are used in the link text to link from page to page. The second program, SiteGraph, reads the data from the database and displays it in a manageable form to the user. It also allows the user to easily modify a variety of constants and threshold values in order to understand what values lead to the best results.
Figure 4-1 Overview of Implementation

4.1. SiteMap

The SiteMap application is responsible for mapping a list of sites, arranging the pages in a hierarchy of each individual site from most general to most specific, extracting link text from each page and inserting all relationships, pages and links into the database. This application was developed using C# in Visual Studio 2005.

![SiteMap application interface]

Figure 4-2 Screenshot of SiteMap as used to map websites

The interface allows the user to add a list of Webpages by entering each site into the URL Entry text box and clicking add. This populates the listbox with each entered URL. When the user clicks “Begin Map” the program will then map each website in order and save the data in XML files. The websites used were determined by searching Google using the topics we intended to create taxonomies for and using the first five results. For a given website the mapping process can take anywhere from half an hour to two or three hours depending on the number of links found.
When the user clicks “Insert Data” the program will open each indicated xml file (created in the Mapping Phase), create a hierarchy of pages, and insert the relationships into the database using the contents of the Root Word text box as the root of each website. It generally takes about half an hour to an hour per site to insert all of its relationships into the database.

4.1.1. Mapping and Parsing

This phase of the program happens when the user clicks the Begin Map button. Each list entry is the root \( w.\text{root} \) of a website \( w \). For each Url in the listbox SiteMap follows all of the mapping specifications explained in Section 3.3.1. This includes the depth restrictions, staying in the same domain as the \( w.\text{root} \) page and not following links that contain specific stopwords.

The anchor text of each link in each webpage is subjected to a cleaning process to extract only the nouns as terms to be added to the XML file using the following steps:

1) Replace common escape sequences with the characters they represent:

\[
\begin{align*}
&\text{&quot} \rightarrow " \\
&\text{&amp} \rightarrow \text{and} \\
&\text{&lt} \rightarrow < \\
&\text{&gt} \rightarrow > \\
&\text{&nbsp} \rightarrow \text{<space>}
\end{align*}
\]

2) Remove any html tags that may fall in the anchor text <*>.

3) Pass the text through Badger Information Extraction Software (http://www-nlp.cs.umass.edu/software/badger.html) and only use the terms identified as nouns by the software.

4) Remove any terms that match the following terms. First remove pronouns:

i, me, mine, you, you, your, he, him, his, she, her, hers, it, its, we, us, our, ours, you, you, yours, they, them, their, theirs
Also, remove artifacts left by badger: %poss%, @@. %poss% is created by Badger to indicate a term is possessive. @@ is created by Badger to indicate a word indicating a time span or position in time. i.e. yesterday, March 13th, 4:00 PM, etc.

5) Remove common endings:

\[\text{ies} \rightarrow \text{y, s (unless the term ends in a double s)}, \text{ing}\]

Originally the nouns were then passed through the Porter Stemming algorithm to remove common endings and combine singular with plural forms. However, it was determined that the Porter Stemming algorithm was too aggressive and combined terms that did not make sense in the context of taxonomy building. For instance, it combines the terms “DIGITAL” and “DIGIT” whose English meaning is very different. Therefore, a simple, custom algorithm was written to only change plural forms to singular and remove the “ing” ending. The set of terms that is retrieved from Badger is now run through this simpler algorithm. This leaves a list of stemmed nouns to be stored as the set of terms for a given link. This set of terms is the same set that is referred to in Section 3.2 and is retrieved using the [] syntax: [p.linksin].

Each website is stored as its own XML file with its Webpages, urls, links, and processed anchor text. There is only one website per file. The XML contains, for each mapped webpage, its URL and an id field. The webpage node has a <linksto> node which contains a list of all pages it links to and the number of times it links to that page. The <linksfrom> node contains a list of all pages that this page is linked from and all of the terms used in each link. The <terms> node contains all of the terms used to refer to this page (obtained from anchor text in the <linksfrom> nodes) and how many times they were used to refer to this page. The <TotalCount> node is the total number of terms used to refer to this page. This XML files is then parsed by the same program in a second run when
the user clicks “Insert Data”. See below for the DTD for the xml file and refer to the appendix for a sample of the file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT Count (#PCDATA)>
<!ELEMENT Term (Text, Count)>
<!ELEMENT Terms (TotalCount, Term*)>
<!ELEMENT Text (#PCDATA)>
<!ELEMENT TotalCount (#PCDATA)>
<!ELEMENT count (#PCDATA)>
<!ELEMENT id (#PCDATA)>
<!ELEMENT link (pointsto, count, terms)>
<!ELEMENT linksfrom (link*)>
<!ELEMENT linksto (link*)>
<!ELEMENT page (url, id, linksto, linksfrom, Terms)>
<!ELEMENT pointsto (#PCDATA)>
<!ELEMENT root (page*)>
<!ELEMENT term (#PCDATA)>
<!ELEMENT terms (term*)>
<!ELEMENT url (#PCDATA)>
```

4.1.2. Inserting Into the Database

After the user has mapped a set of websites and has saved the XML files using SiteMap, they would use the program again to import the XML files into the database. This time the user would enter the XML file names into the listbox and press the “Insert Data” button. The user also indicates the root subject that they are importing:

![Screenshot of Sitemap used to insert data into the database](image)

Figure 4-3 Screenshot of Sitemap used to insert data into the database
When the user clicks “Insert Data” SiteMap imports each XML element one at a time and builds the tree of Webpages from most generic to most specific as detailed in Section 3. Each XML file contains information from only one website.

After the tree of Webpages has been built in memory, SiteMap recursively traverses the tree breadth-first and inserts each relationship (r in Section 3.2) into the SQL Database for later analysis. SiteMap also inserts all links between all Webpages and all processed anchor text into the database in order to help in calculating the term_anchor_support as mentioned previously.

4.2. SiteGraph

Now that the data has been inserted in the database the SiteGraph application provides an interface to examine the data and extract a hierarchy of terms from it based on the link structure of the tree created by SiteMap and inserted into the database.
4.2.1. Word Graph

The Word Graph tab makes use of the Open Source Netron Light control that continues to be developed under the name Netron Reloaded\(^1\). This tab displays the tree built from the database using the algorithm described in Section 3. It allows the user to adjust all of the variables that control the tree structure which helps them determine the settings that optimize the results. The tree diagram displays nodes retrieved as topic terms in red and nodes retrieved as instance terms in blue. The darker the shade the larger a percentage the child word appears in the parent word. It generally takes about a minute to generate a taxonomy from the relationships in the database depending on the options selected.

\(^1\) http://sourceforge.net/projects/netron-reloaded
**Root Term:** This is the term that the user wishes to use as the root node of the tree. Although other terms can be used, the root term specified when the tree is inserted into the database by SiteMap is generally used as it contains the richest and deepest tree. This is the same as the *root-term* parameter in Section 3.4.

**Max No. Levels:** This is the maximum height of the tree. The value is considered a safeguard to keep the program from spending too much time generating the tree and is usually chosen to be larger than the expected number of levels. Most trees generate around three levels of topic terms so a common value for this parameter is four. This is the same as the *max-levels* parameter in Section 3.4.

**Topic Cutoff:** This is the same as the *topic-cutoff* parameter in Section 3.4. It controls how restrictive the algorithm is when finding Topic Terms.

**Instance Cutoff:** This is the same as the *instance-cutoff* parameter in Section 3.4. It controls how restrictive the algorithm is when finding Instance Terms.

**Top Topic Percent:** This is the same as the *top-topic-percent* parameter in Section 3.4. It controls what percentage of potential topics actually become topics.

**Topics Only:** This check box keeps the program from retrieving the instance nodes. This is used to reduce the time to build the tree but still see the main structure. It is always left checked for generating complete taxonomies.

**Find Cross Topic:** This check box controls whether or not “Cross-Topic” nodes are identified. It is always left checked for generating complete taxonomies.

**Topic Trim Cutoff:** This check box and the nearby text box control whether or not the instance trimming rules are applied to each node. If they are, then value indicated in the text box indicates the percentage required to be kept in order for the child nodes to be
considered topic nodes as described in Section 3.4. It is always left checked for generating complete taxonomies. The value in the textbox is used as the \textit{topic-trim-cutoff} value in Section 3.4.

\textbf{Modifier Cutoff:} This check box and the nearby text box control whether or not all of the child nodes are checked against their siblings to see if their collocation is significant enough to be a modifier. It is always left checked for generating complete taxonomies. The value in the textbox is used as the \textit{modifier-cutoff} value in Section 3.4.

\textbf{Export Tree:} The export tree button exports the displayed tree into a comma delimited .csv file. If the tree is:

```
sport, football, 
golf, leaderboard, 
,, tournament
```

then the comma delimited file would be:

```
sport,football, 
,,golf,leaderboard 
,,tournament
```

and it would appear in a spreadsheet editor as:

<table>
<thead>
<tr>
<th>sport</th>
<th>football</th>
</tr>
</thead>
<tbody>
<tr>
<td>golf</td>
<td>leaderboard</td>
</tr>
<tr>
<td></td>
<td>tournament</td>
</tr>
</tbody>
</table>

\textbf{Context Menu}
If the user right clicks on any node in the tree they are given a context menu which gives them tools to further understand and make sense of the underlying data:
Stats Dialog
The stats dialog shows a variety of stats regarding the node and the node’s children. It also provides links allowing the user to navigate between the node’s children, parent, and siblings.
Figure 4-7 Screenshot of the Hierarchy dialog in SiteGraph

Hierarchy Dialog

The hierarchy dialog uses a hierarchical clustering algorithm detailed in [8] on the children of the selected node and displays the results. The algorithm begins by assigning each term to its own cluster. It then repeatedly combines the two “nearest” clusters until there is only one cluster remaining. The algorithm uses the group-average distance between two clusters, i.e. the average cosine distance between all combinations of terms from each cluster to determine the total distance between clusters. The distance between two terms is defined as the cosine distance between vector representations of each term. The vectors used are either constructed from the collocation with all other terms in pages or in links (chosen by the user). Two terms collocate in a link if they both appear in the anchor text of the same link. Two terms collocate in a page if they both appear in the anchor text of links that appear on the same page. For instance, if a term $t$ extracted from the anchor text of a link collocates with $t_0$ 24 times, $t_1$ 17 times, $t_2$ 0 times, etc. then the
first three values in its vector representation would be \([24, 17, 0, \ldots]\). The cosine distance between two vectors is defined as:

\[
\cos (X, Y) = \frac{\sum_{x \in X, y \in Y} xy}{\sqrt{\sum_{x \in X} x^2 \sum_{y \in Y} y^2}}
\]

In this manner the terms are clustered and displayed to the user.

### 4.2.2. Word Frequency

![Figure 4-8 Screenshot of the Word Frequency tab in SiteGraph](image)

The Word Frequency tab uses a charting object developed by .net Charting\(^1\). This tab displays the distribution of term appearances in links and pages (a term is considered to appear in a page if it appears in the anchor text in a link in that page) and helps the user choose the cutoff values for links and pages. This tab is used in conjunction with the

---

\(^1\) [http://www.dotnetcharting.com](http://www.dotnetcharting.com)
Word Management Tab which shows the actual words that are allowed for different thresholds.

### 4.2.3. Word Management

![Figure 4-9 Screenshot of the Word Management tab in SiteGraph](image)

The word management tab allows the user to see the current stop words and allowed words. It also lets them choose the threshold values for percent appearance in pages and links.

If the “Use ‘OR’ relationship” checkbox is checked, the union of the terms allowed by pages and terms is used. However, if it is unchecked, the intersection is used.

### 4.3. Miscellaneous tools

#### 4.3.1. stopwords.vbs

Stopwords.vbs is a vbscript that was written to import the list of stopwords into the database. It opens a text file, stopwords.txt, and inserts all of the words into the stopwords table in the database.
Stopwords.txt was initially comprised of the words found at this site http://www.pagex.com/webtools/stopwords.cfm. Additional words were added such as numbers and various symbols.

4.3.2. wordnet.vbs

Wordnet.vbs is a vbscript written to determine what percentage of the terms that appeared in the SiteMap database also appeared in Wordnet. It is currently unused. Originally Wordnet was going to be used to evaluate the generated taxonomies. However, due to a large number of proper terms (brand names, product names), Wordnet was not recognizing a significant percentage of the terms and could not be used to evaluate the taxonomies. In the Computer Hardware domain it recognizes 3702 out of 8515 total terms: 43.8%.

4.3.3. results.vbs

Results.vbs is a vbscript written to read the .csv files output by the SiteGraph application and generate the conditional entropy and subsumption results discussed in Section 5.

4.4. Implementation Issues

A variety of issues and difficulties came up during the implementation process and many lessons were learned during the process. For instance, originally, only the relationships created while building the pyramid were inserted into the database using the words and relationship tables. This caused issues when the data such as collocation in pages and links became important and so the process was rewritten to add the full link structure in using the link and pages tables. This allowed a considerable amount of new data to make sense of the tree and relationships between the words. The process should
have been written from the ground up to put the lowest level of data (the words, links and urls) into the database and the word relationships should have been built from that. It would have allowed much better flexibility in the relationship building algorithms from the very beginning.

Secondly, an open source crawler should have been used. A custom written crawler was used which had to be updated several times to support more and more tags as they were discovered. Portions of some websites were probably omitted because certain kinds of links are not supported by the currently used crawler. The current crawler only supports the `<a>` tag and the `alt=` attribute for image links. An open source crawler would have been significantly easier, faster, and more complete than the custom written crawler.

Also, a text parser, Badger (http://www-nlp.cs.umass.edu/software/badger.html), was used to extract nouns from the anchor text as that is the only part of speech that was to be included in the hierarchy tree. It was hoped that adjectives, verbs, and other parts of speech would be excluded. Unfortunately, Badger struggled with most anchor text because most anchor text is not complete sentences. Badger does an admirable job of parsing full sentences, however, with phrases, it tends to consider the entire thing as a noun phrase and mark every term as a noun. For instance, this phrase: “Overclocking 9 Value-Priced DDR2-800 Kits” has every word marked as a noun. Future implementations would use a different text parser.

Finally, a C# implementation of the Porter Stemming algorithm was originally used to stem words and combine concepts. However, the stemming tended to be too strict and it was decided that a custom written parser that simply stripped plurals and “ing” endings would appropriately combine terms. The Porter Stemming was combining
terms such as “DIGITAL” and “DIGIT”, concepts that we wanted to remain separate. Therefore a much simpler algorithm was implemented that suited the needs of this project more appropriately.

Many issues were encountered and lessons were learned during the implementation process. Most importantly:

1) Always store the lowest level of data before processing

2) Use already written, proven code where available

3) Understand the strengths and limitations of all of the third party tools used.
5. EXPERIMENTAL RESULTS

5.1. Generation

In order to evaluate the validity of the ideas proposed in this thesis, three topic domains were chosen at random: Computer Hardware, Sports, and News. In order to be impartial, the set of websites \(W\) chosen to represent each topic domain are the first five websites returned by Google when searching for each phrase “Computer Hardware”, “Sports” and “News” (entered without quotes into Google). The first five Webpages for each domain are:

**Computer Hardware:**
- http://www.pricewatch.com
- http://www.geeks.com
- http://www.newegg.com
- http://www.tomshardware.com
- http://www.devhardware.com

**Sports:**
- http://espn.go.com
- http://sports.yahoo.com
- http://msn.foxsports.com
- http://www.sportsline.com

**News:**
- http://www.cnn.com
- http://news.google.com
- http://www.foxnews.com
- http://news.yahoo.com
- http://www.msnbc.com

Each of these URLs was used as a \(w_{\text{root}}\) along with \(\delta = 3\) and \(\alpha = 0.6\) (these values were determined by trial and error) to create each website in the set \(W\). These websites were the top five returned by Google in the fall of 2006.
5.1.1. Human Judging

The above algorithms were used to generate one taxonomy for each set of websites (each topic phrase) using the following settings. The \textit{topic-cutoff} and \textit{topic-trim-cutoff} are not included because they were determined by comparing input combinations versus output measures in Section Error! Reference source not found.. The other settings were used for the taxonomies submitted for human judging and they were determined previously by trial and error.

\textbf{Computer Hardware:} (crawled 2/4/2007)

\textbf{Tree Parameters}
\begin{itemize}
  \item root-term: HARDWARE
  \item max-levels: 4
  \item instance-cutoff: .01
  \item modifier-cutoff: .4
  \item top-topic-percent: 10%
\end{itemize}

\textbf{Term Set Parameters}
\begin{itemize}
  \item Pages: .0007
  \item Links: .0007
  \item Intersection (And relationship)
\end{itemize}


\textbf{Tree Parameters}
\begin{itemize}
  \item root-term: SPORT
  \item max-levels: 4
  \item instance-cutoff: .01
  \item modifier-cutoff: .4
  \item top-topic-percent: 10%
\end{itemize}

\textbf{Term Set Parameters}
\begin{itemize}
  \item Pages: .0008
  \item Links: .0004
  \item Intersection (And relationship)
\end{itemize}

\textbf{News:} (crawled 1/30/2007)

\textbf{Tree Parameters}
\begin{itemize}
  \item root-term: NEW
  \item max-levels: 4
  \item instance-cutoff: .01
  \item modifier-cutoff: .4
  \item top-topic-percent: 10%
\end{itemize}

\textbf{Term Set Parameters}
\begin{itemize}
  \item Pages: .001
The complete taxonomy result sets submitted for human judging are available in the appendix.

5.2. Evaluation

In order to evaluate the taxonomies created from the sites listed above, three methods were chosen. First, the conditional probability of the child and parent term appearing in the same page was used. In our case, since we only used the anchor text of links, this measure is the conditional probability of the child and parent term appearing in the anchor text of two links appearing in the same page. According to this subsumption measure [12, 15, 20], if \( p \) subsumes \( c \) then the following is true:

\[
P(p \mid c) > 0.8 \land P(c \mid p) < P(p \mid c)
\]

The second measure, Generalization/Specialization Quality, is calculated using the conditional entropy of the parent and child terms [18]. Given a child term, \( c \), and a parent term, \( p \), the Generalization/Specialization Quality is calculated as:

\[
- P(p, c) * \log(P(p \mid c))
\]

This measure is designed to “quantify how a hierarchy respects the relation of generalization/specialization between the objects which are in the nodes.” [18] A lower value indicates a better taxonomy.

The third measure used human judges to determine the validity of the taxonomy. Six judges were asked to rate each relationship (parent-term, child-term). This measures the precision of the created taxonomy. Each of the judges was at least moderately familiar with each of the three topic domains, NEWS, SPORTS, and COMPUTER
HARDWARE. The judges were knowledgeable enough to understand the basic concepts in each domain and how they are related to each other.

5.2.1. Effect of Cutoff Parameters

Based on preliminary trial and error experiments, we found that the two input parameters that most affect the content of the tree were the *topic-trim-cutoff* and the *topic-cutoff*. The *topic-cutoff* affects the number of topic nodes allowed into the tree and the *topic-trim-cutoff* affects how liberal the algorithm is when trimming Topic nodes. Both of these parameters have a very large effect on the length and breadth of the tree.

Sixteen combinations of both input parameters for each domain were tried versus the percent of relationships in the output taxonomies that fulfilled both requirements of the subsumption measure. First, $P(\text{parent}|\text{child})$ must be greater than 60% (see rationale for choosing 60% below). Second, $P(\text{parent}|\text{child})$ must be greater than $P(\text{child}|\text{parent})$. The chosen input values for topic-cutoff were: .002, .003, .004, and .005. The chosen input values for topic-trim-cutoff were: .2, .3, .4, and .5. The effect of the parameters on the average Specialization/Generalization quality described in [18] was also studied.

Due to the fact that the subsumption measure is designed for traditional taxonomy building techniques [12, 15, 20], instead of using the recommended 80% guideline we will be using 60%. Traditional techniques use either syntactic pattern matching or statistical collocation both of which require related terms to be near each other in text. Therefore, such measures are good for evaluating taxonomies already based on distance or pattern matching. Because the techniques developed in this thesis rely primarily on link structure, related terms don’t necessarily have to appear in the same document.
Also, the subsumption measure was designed using complete text documents which would contain significantly more text than the anchor text used by these algorithms.

**Computer Hardware**

![Graph showing the percent of output that fulfills the subsumption measure vs. input parameters for the Computer Hardware domain.](image)

<table>
<thead>
<tr>
<th>topic-trim-cutoff</th>
<th>0.003</th>
<th>0.005</th>
<th>0.004</th>
<th>0.002</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>42.42</td>
<td>49.02</td>
<td>63.25</td>
<td>59.35</td>
</tr>
<tr>
<td>0.4</td>
<td>40.00</td>
<td>50.94</td>
<td>64.57</td>
<td>60.00</td>
</tr>
<tr>
<td>0.3</td>
<td>66.67</td>
<td>74.42</td>
<td>61.18</td>
<td>57.14</td>
</tr>
<tr>
<td>0.2</td>
<td>66.67</td>
<td>72.41</td>
<td>71.01</td>
<td>58.46</td>
</tr>
</tbody>
</table>

Figure 5-1 Percent of Output that fulfills Subsumption Measure vs. Input parameters for Computer Hardware domain

![Graph showing the average quality vs. input parameters for the Computer Hardware domain.](image)
Figure 5-2 Average Generalization/Specialization quality vs. Input parameters for Computer Hardware domain

According to the subsumption measure, the Computer Hardware domain was best when the topic-cutoff is 0.004. Also, except for a couple exceptions, it appears as if the more restrictive values for the topic-cutoff and topic-trim-cutoff generate better results. Also, the average Generalization/Specialization Quality agrees with the Subsumption measure. According to both measures the worst taxonomies are created when the topic-cutoff is 0.005 and the topic-trim-cutoff is 0.5 or 0.4.

Sports

Figure 5-3 Percent of Output that fulfills Subsumption Measure vs. Input parameters for Sports domain
The Sports topic domain also shows that more restrictive values of the topic-cutoff and topic-trim-cutoff tend to create better results as measured by the Subsumption measure. Overall the results are the best of all three topic domains and reach as high as 85% of all output relationships according to the Subsumption measure. Also, the Subsumption measure and the Generalization/Specialization quality again agree on the best and worst taxonomies in the Sports domain.
News

Figure 5-5 Percent of Output that fulfills Subsumption Measure vs. Input parameters for News domain

<table>
<thead>
<tr>
<th>topic-cutoff</th>
<th>0.005</th>
<th>0.004</th>
<th>0.003</th>
<th>0.002</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>63.16</td>
<td>52.29</td>
<td>48.39</td>
<td>42.76</td>
</tr>
<tr>
<td>0.4</td>
<td>65.79</td>
<td>54.46</td>
<td>52.03</td>
<td>47.24</td>
</tr>
<tr>
<td>0.3</td>
<td><strong>95.24</strong></td>
<td>54.70</td>
<td>57.14</td>
<td>52.71</td>
</tr>
<tr>
<td>0.2</td>
<td>95.24</td>
<td>54.70</td>
<td>63.41</td>
<td>66.67</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>topic-cutoff</th>
<th>0.005</th>
<th>0.004</th>
<th>0.003</th>
<th>0.002</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.0025</td>
<td>0.0035</td>
<td>0.0028</td>
<td>0.0026</td>
</tr>
<tr>
<td>0.4</td>
<td>0.0025</td>
<td>0.0031</td>
<td>0.0029</td>
<td>0.0025</td>
</tr>
</tbody>
</table>
The News topic domain also shows that more restrictive values of the *topic-cutoff* and *topic-trim-cutoff* tend to create better results as measured by the subsumption measure. The two output taxonomies that had the highest ratings (95%) both created less than twenty relationships in the final taxonomy. For more practically sized taxonomies, the scores in the News topic domain were less than the previous domains. The most impressive score being about 65% of output relationships fulfilling the subsumption measure. Also, the two measures again agree on which taxonomies are the best.

Over all three topic domains there appears to be a correlation between *topic-cutoff* and *topic-trim-cutoff*. Making these values more restrictive tends to create results that score better using the both the Subsumption Measure and the Generalization/Specialization quality.

### 5.2.2. Human Judging

Six taxonomies were generated and submitted for human judging. Each taxonomy was generated using the settings in Section 5.1.1 and a *topic-cutoff* and *topic-trim-cutoff* that generated the best taxonomies in Section *Error! Reference source not found.*. For the SPORTS and COMPUTER HARDWARE domain, two different taxonomies were indicated best by the Subsumption measure and the Generalization/Specialization quality. Therefore, both taxonomies were submitted for human judging. In the NEWS domain the same taxonomy scored the best for both measures. The second best taxonomy from the NEWS domain was also added. The chosen taxonomies for human evaluation are listed in Table 5-1.
### Table 5-1 Output Measures for the taxonomies submitted for human judging

<table>
<thead>
<tr>
<th>Topic Domain</th>
<th>topic-cutoff</th>
<th>topic-trim-cutoff</th>
<th># of relationships</th>
<th>Subsumption Measure</th>
<th>Generalization/Specialization Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer Hardware</td>
<td>0.004</td>
<td>0.3</td>
<td>43</td>
<td>74.42%</td>
<td>0.0013</td>
</tr>
<tr>
<td>Computer Hardware</td>
<td>0.005</td>
<td>0.2</td>
<td>36</td>
<td>66.67%</td>
<td>0.0011</td>
</tr>
<tr>
<td>Sports</td>
<td>0.003</td>
<td>0.2</td>
<td>40</td>
<td>85.00%</td>
<td>0.0043</td>
</tr>
<tr>
<td>Sports</td>
<td>0.003</td>
<td>0.4</td>
<td>56</td>
<td>67.86%</td>
<td>0.0039</td>
</tr>
<tr>
<td>News</td>
<td>0.005</td>
<td>0.2</td>
<td>21</td>
<td>95.24%</td>
<td>0.0008</td>
</tr>
<tr>
<td>News</td>
<td>0.003</td>
<td>0.2</td>
<td>123</td>
<td>63.41%</td>
<td>0.0020</td>
</tr>
</tbody>
</table>

#### 5.2.3. Human Judges

Each of six human judges was presented with the following text:

A taxonomy is a classification in a hierarchical system. A hierarchy is a system of ranking and organizing things, where each element of the system (except for the top element) is subordinate to a single other element. The concept of the subordinate, child element should be more specific than the parent element.

For instance:

In the domain of Musical Instruments
- Parent -> Child
- DRUM -> BONGO

would make sense because a bongo is a type of drum.

You will be presented with pairs of elements (a parent and child) from a taxonomy of terms generated from three topic domains: SPORTS, COMPUTER HARDWARE, and NEWS. For each pair rank them on a scale from 1 (is not meaningful) to 3 (is meaningful) keeping in mind the domain the term pair is coming from.

1: NOT MEANINGFUL
2: SOMEWHAT MEANINGFUL
3: MEANINGFUL

Then each judge used a program, SiteJudge, to evaluate the pairs of terms in the result set. The judge was presented with this screen:

![Screenshot of SiteJudge](image.png)

**Figure 5-7 Screenshot of SiteJudge**

The parent and child relationship was displayed along with the domain and a set of radio buttons to select the rating (1 through 3). Selecting a rating would make the next relationship immediately appear. If the judge felt they had misselected or wanted to review their choices they could use the back button. To see the full set of results refer to the Appendix.
The average ratings and standard deviation of all relationships from all judges for each submitted taxonomy was:

<table>
<thead>
<tr>
<th>Topic Domain</th>
<th>topic-cutoff</th>
<th>topic-trim-cutoff</th>
<th>Average Rating</th>
<th>Percent &gt; 2</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer Hardware</td>
<td>0.004</td>
<td>0.3</td>
<td>2.43</td>
<td>79.07%</td>
<td>0.45</td>
</tr>
<tr>
<td>Computer Hardware</td>
<td>0.005</td>
<td>0.2</td>
<td>2.30</td>
<td>66.67%</td>
<td>0.44</td>
</tr>
<tr>
<td>Sports</td>
<td>0.003</td>
<td>0.2</td>
<td>2.43</td>
<td>90.00%</td>
<td>0.43</td>
</tr>
<tr>
<td>Sports</td>
<td>0.003</td>
<td>0.4</td>
<td>2.35</td>
<td>82.14%</td>
<td>0.49</td>
</tr>
<tr>
<td>News</td>
<td>0.005</td>
<td>0.2</td>
<td>2.30</td>
<td>80.95%</td>
<td>0.49</td>
</tr>
<tr>
<td>News</td>
<td>0.003</td>
<td>0.2</td>
<td>2.24</td>
<td>57.72%</td>
<td>0.45</td>
</tr>
</tbody>
</table>

Table 5-2 Average and standard deviation of relationship ratings by topic and totals

As one can see the results are positive. The Sports domain did the best with 90% of relationships having an average rating across all raters greater than 2, while the News domain was overall the worst. This may be due to the fact that News is, perhaps, the broadest of all three topics. In the case of the Sports and Computer Hardware domains, the taxonomy that scored better using the Subsumption measure also was judged better by humans. Also the standard deviations are all relatively low meaning that, in general, the judges agreed on the meaning of the terms and rated them very closely.
6. CONCLUSION AND FURTHER WORK

This thesis describes a method to extract information from the link structure of a website and proves that this is a viable method for automatic taxonomy generation. Two programs, SiteMap and SiteGraph, were developed in order to demonstrate the information available in the link structure of the website.

The taxonomies created by SiteMap and SiteGraph were evaluated using three different techniques. The Subsumption measure, the Generalization/Specialization Quality and particularly the human judging validate the use of the link structure of Websites to create term taxonomies. The methods used in this thesis do not necessarily create perfect taxonomies, but they definitely prove that the link structure contains unused information that, combined with other techniques, can create more and more accurate term taxonomies with less and less human interaction. This thesis contributes new ideas about using the link structure and text from the World Wide Web, and has an impact on the fields of Information Retrieval and Data Mining. In the future, more information will become available on the web, hyperlinked with existing documents, which means that the use of the links between documents will become a more important clue into data categorization.

Also in the future, the ideas in this thesis could be used in combination with other automatic taxonomy generation algorithms. The link structure could be used in concert with clustering or syntactic analysis in order to create more accurate taxonomies. Also,
the link structure could be used more directly by clustering algorithms by considering the linking terms in collocation measures.

There are still many barriers to automatically creating taxonomies that the ideas in this thesis did not overcome. Accurately parsing natural languages is difficult, but as these techniques improve, taxonomy generation will also improve greatly and vice versa. There are three main shortcomings of natural language processing that impacted the results of this thesis.

First, this thesis attempted to only categorize nouns by using a natural language processor, but it became apparent that a large number of other parts of speech ended up in the taxonomies. Part of speech processing is important in all kinds of taxonomy generation and gives important clues to how words should be placed in a hierarchy.

Second, the accurate identification of n-grams in the document set would have increased the strength of the algorithm. If you pull apart the n-gram “SOUND CARD”, you lose the meaning of the terms in the document. Processing the term “SOUND” separately from “CARD” is completely different from the combined concept.

Finally, more accurately identifying and combining the meaning of synonymous terms and phrases would have strengthened the concepts in the final taxonomies. Some concepts were referred to using many different term sets such as: “COLLEGE BBALL”, “BASKETBALL”, “NCAA BASKETBALL” or from another domain: “MOTHERBOARD”, “MOBO” or “MB”. If these concepts were identified as being similar while parsing the documents, and their meanings combined, the final taxonomies would have been significantly stronger.
As computers become more “intelligent” and require the ability to classify larger amounts of information, taxonomy generation becomes more important. The ability to process new data and place it appropriately into a hierarchy of known data is important for both humans and computers. Early research, such as the work in this thesis, on taxonomy generation and data classification can be considered an important step in the field of Artificial Intelligence and Semantic Web Mining.
REFERENCES


APPENDIX A (CODE)

This appendix contains code detailing the most important parts of the SiteGraph and SiteMap applications. The code is not reproduced here in full due to its length.

This first code section describes the way the Webpages are organized into a tree hierarchy using the depth rating.

Assume:
+Q is a Queue
   Enqueue(): Adds an object to the tail of the queue.
   Dequeue(): Returns and removes the object at the head of the queue.
   length : Returns the number of objects in the queue.
+T is a Tree
   root   : The root treenode
+W is a Website as defined in the Notation section
+A TreeNode has the following properties and functions
   page   : A Webpage as defined in the Notation section
   nodes  : The set of child nodes of this node.
   AddNode(): Takes a TreeNode and adds it to the nodes set.
   depth  : the distance from the root node.
+FindPageInTree(p) is a function that searches the tree for the Webpage p in the each TreeNode's Page property. It returns a TreeNode if it finds it, otherwise null.
+MoveNodeInTree(tn, tn') is a function that moves the node tn (and all descendants) from its current location to be a child of tn'.
+DepthStdDev is the standard deviation of all depth ratings of Webpages in W.pages
+tChild is a TreeNode
+pChild is a Webpage
+tNew is a newly created TreeNode

fBuildTree()

T.root.page = w.root
Q.Enqueue(T.root)

//while we have nodes left to process
while Q.length > 0
  TreeNode tNode = Q.Dequeue()

  //check each webpage this node points to
  for each Webpage pChild in tNode.page.pagesto
    //check to see if the node already exists
    if (tChild = FindPageInTree(pChild)) is null
      //it hasn't been added, so by default its our child
      tNew.page = pChild
      tNode.AddNode(tNew)
      Q.Enqueue(tNew)
    else
      //it already exists in the tree, check to see if we should move it
      if tNode.page.deprating + DepthStdDev < tChild.page.deprating
        if tNode.depth > tChild.depth
          MoveNodeInTree(tChild, tNode)
        end if
      end if
    end if
  end for
end while
The above algorithm adds each TreeNode with its associated Webpage into a queue, starting with the TreeNode encapsulating \texttt{w.root}, from which they are removed and processed one at a time. As it processes the TreeNode it checks each member of the TreeNode’s Page’s \texttt{pagesto} set. If the Webpage from the \texttt{pagesto} set has not yet been added to the tree then a new TreeNode is created and added as a child of \texttt{tNode} with the Webpage as its \texttt{page} property. If the Webpage has already been added to the tree, it evaluates two conditional expressions to determine if it should move the Webpage to be a child of the current TreeNode.

\begin{align*}
  & t.\text{Node}.\text{page}.\text{deprating} + \text{DepthStdDev} < t.\text{Child}.\text{page}.\text{deprating} \\
  & t.\text{Node}.\text{depth} > t.\text{Child}.\text{depth}
\end{align*}

The first expression checks to see if the depth rating of the Webpage in question is greater than the potential parent Webpage’s depth rating plus one standard deviation. The standard deviation is used to make sure that the Webpage’s depth ratings are significantly different.

The second expression makes sure that we are always moving the TreeNode "down" the tree to a lower level. This check makes sure that a TreeNode that exists at a lower, "more specific" level isn’t moved up or sideways.

This next code section describes the way the hierarchy of Webpages is used to create the Relationship (a set of tuples described in Section 3.2).

Assume:
+ The TreeNode objects are the same as those in the above section (Organizing the Pages) 
+ \texttt{L} is a list 
  \texttt{Contains()}: Returns true if the input string exists in \texttt{L} 
  \texttt{Add()}: Adds the input string to \texttt{L} 
+ \texttt{T} is the same tree as in the above section (Organizing the Pages) 
+ This recursive function is called on \texttt{fCalcRel(T.root, \{\})} 
+ \texttt{AddRel()} is a function defined externally which takes a parent term, child term, and a value indicating the strength of the relationship. These values are stored for later use.

\texttt{fCalculateRelationships(TreeNode tn, List L)
This recursive algorithm descends the tree created out of the Webpages in the website and adds relationships for all of the anchor text used to refer to the current webpage as the “parent” terms (tn.page.terms) and all of the anchor text used to refer to all of its child Webpages as the “child” terms (tn.nodes.page.terms). The fAddRel() function is a wrapper for the add() function described in Section 3.2 above. If a tuple, r, already exists such that r.parent-term = parent.term and r.child-term = child.term then

\[
\text{r.count} += \frac{\text{count}(\text{pChild}, \text{child})}{\sum_{\text{t} \in \text{pChild.terms}} \text{count}(\text{pChild}, \text{t})}
\]

Otherwise, a new tuple is created and added to the relationship R with

\[
\frac{\text{count}(\text{pChild}, \text{child})}{\sum_{\text{t} \in \text{pChild.terms}} \text{count}(\text{pChild}, \text{t})}
\]

as r.count.

The list \( \mathcal{L} \) is used to keep the algorithm from adding children to a term if any ancestor node has already handled that term. Early on during development it was noticed that very often, more generic terms used farther up in the tree would have their child relationships severely altered by more specific uses farther down inside of the tree.
Therefore, a list of already used words was created to keep descendants from contributing to an already defined term.

It should be noted that, as the terms are added, they are normalized so that the contribution of each child webpage was weighed equally:

\[
\text{fAddRel}(\text{parent, child}, \sum_{\text{pChild terms}} \frac{\text{count}(\text{pChild, child})}{\text{count}(	ext{pChild}, t)})
\]

This next code section shows the driving loop that SiteGraph uses to create the final taxonomies using the Relationship.

Assume:
+T is a Tree
  root : The root treenode
+A TreeNode has the following properties and functions
  term : a character string
  nodes : The set of child nodes of this node.
  AddNode(): Takes a TreeNode and adds it to the nodes set.
  depth : the distance from the root node.
  instance: a boolean value indicating if this is an instance term
+root-term and max-levels are input parameters as defined above.

fBuildTaxonomy()

T.root.term = root-term
CurrentLevel = 0
while (CurrentLevel < max-levels)
    BuildLevel(CurrentLevel, T.root)
    TrimLevel(CurrentLevel, T.root)
    CurrentLevel++
end while

This code section shows how the taxonomy is trimmed while it is being created.

Assume:
+L is a list of tree nodes
  Add() : adds the input TreeNode to the list
+AddInstanceChildren(): adds instance terms as described above using the input TreeNode.term as the parent term.
+RemoveChildren(): Removes all child nodes from the TreeNode.nodes set
+CountUniqueChildrenInWholeTree(): Counts the number of children of the input TreeNode whose .term appears nowhere else in the tree.
+CountUniqueChildrenInHigherLevels(): Counts the number of children of the input TreeNode whose .term appears in no nodes whose depth is <= the input TreeNode.depth.
+RemoveChildrenInHigherLevels(): Removes all child nodes of the input TreeNode whose .term appears whose depth is <= the input TreeNode.depth.
+UniqueChildren and TotalChildren are integer values
+topic-trim-cutoff is the cutoff parameter set by the user described above.
+CurrentLevel is input this trimming function

TrimLevel(CurrentLevel, T)

//get all nodes at this level
for each TreeNode tn in T where tn.depth = CurrentLevel
    nUniqueChildren = CountUniqueChildrenInWholeTree(tn)
// and remove children if we either contribute no new children or our
// new children/all children ratio is less than topic-trim-cutoff
if nUniqueChildren = 0
    L.Add(tn)
else
    nUniqueChildren = CountUniqueChildrenInHigherLevels(tn)
nTotalChildren = |tn.nodes|
    if nUniqueChildren / nTotalChildren > topic-trim-cutoff or
        nUniqueChildren = 1
        // add to the remove list
        L.Add(TreeNode)
    end if
end if
end for

// Remove all children from trimmed topic nodes and add instance children
for each TreeNode tn in L
    RemoveChildren(tn)
    AddInstanceChildren(tn)
end for

// now remove children that have already appeared in the tree at a higher lvl
for each TreeNode tn at CurrentLevel
    RemoveChildrenInHigherLevels(tn)
end for

This code section shows how crosstopic terms are discovered which is part of the
trimming process.

Assume:
+ TermCount is an object with the following properties and methods
    count: The number of times a term has appeared
    treenode: The treenode containing the Term
+ L is a list of TermCount objects
    Contains(): takes a string and returns true if one of the TermCount.treenode.terms
        matches it.
    Add(): takes a TreeNode and integer and adds a new TermCount object
[]: takes a string and returns the TermCount object whose treenode.term matches it.
+ CalcAvgAppearance() takes a list of term count objects and returns the average of the
    count property
+ CalcStdDevAppearance() takes a list of term count objects and returns the standard
    deviation of the count property
+ MarkAsCrossTopic() marks the node as a crosstopic node so it’s displayed
    appropriately to the user.
+ Avg and StdDev are real values

for each TreeNode tn in T where tn.Depth = CurrentLevel + 1
    if L.Contains(TreeNode.Term)
        listNodeCounts[TreeNode.Term].count++
    else
        listNodeCounts.Add(TreeNode, 1)
    end if
end for
fFindCrossTopic()

Avg = CalcAvgAppearance(L)
StdDev = CalcStdDevAppearance(L)

for each TermCount tc in L
    // if this term appears more times than the average plus two times the
    // std dev, than this is an outlier and a modifier term.
    if tc.Count > Avg + StdDev * 2 then
        MarkAsCrossTopic(tc.treenode)
    end if
end for
SiteMap.sql

Due to length, only the tables, functions, and procedure stubs are included.

--use sitemap
--create database sitemap
use sitemap
go
create table words
{
    id int NOT NULL,
    word varchar(100) NOT NULL
}
go
create table relationship
{
    parent_id int NOT NULL,
    child_id int NOT NULL,
    count int NOT NULL
}
go
create table pages
{
    id int NOT NULL,
    url varchar(500) NOT NULL
}
go
create table link
{
    id int NOT NULL,
    to_id int NOT NULL,
    from_id int NOT NULL,
    word_id int NOT NULL
}
go
create table stopwords
{
    id int NOT NULL
}
go
CREATE FUNCTION fGetNextID()
RETURNS int
create procedure pInsertWord
    @p_Word varchar(100)
create function fGetID
    (@p_Word varchar(100))
RETURNS int
create procedure pInsertRelationship
    @p_WordParent varchar(100),
    @p_WordChild  varchar(100),
    @p_Count      int
CREATE function fGetParentCount
    (@p_id int)
RETURNS int
CREATE FUNCTION fGetNextPageID()
RETURNS int
CREATE FUNCTION fGetNextLinkID()
RETURNS int
create function fGetPageID
(@p_URL varchar(500))
RETURNS int

create procedure pInsertPage
@p_URL    varchar(500)

create function fGetLinkID
(@p_FromURL varchar(500), @p_ToURL varchar(500))
RETURNS int

create procedure pInsertLinkTerm
@p_FromURL varchar(500),
@p_ToURL   varchar(500),
@p_Term   varchar(100)

create function fGetAllowedTerms
(@p_LinkThreshold real,
@p_PageThreshold real,
@p_Or   bit)
RETURNS @words table

    id    int,
    word  varchar(100),
    in_links  int,
    total_links  int,
    in_pages  int,
    total_pages  int

create function fGetAllowedTermsFromLinks
(@p_LinkThreshold real)
RETURNS @words table

    id    int,
    word  varchar(100),
    inlinks  int,
    totallinks  int,
    percentlinks  real

create function fGetAllowedTermsFromPages
(@p_PageThreshold real)
RETURNS @words table

    id    int,
    word  varchar(100),
    inpages  int,
    totalpages  int,
    percentpages  real

create function fGetChildren
(@p_word    varchar(100),
@p_LinkThreshold real,
@p_PageThreshold real,
@p_Threshold real,
@p_Or   bit)
returns @childwords table

    child_id    int,
    child_word  varchar(100),
    child_count int,
    parent_count int,
    perc_appear real,
    in_links  int,
    total_links  int,
    in_pages  int,
    total_pages  int
create function fGetCollocationByPage
    (@p_word1 varchar(100),
     @p_word2 varchar(100))
returns @collocation table
    (word1total int, word2total int, collocation int)

create function fGetCollocationByLink
    (@p_word1 varchar(100),
     @p_word2 varchar(100))
returns @collocation table
    (word1total int, word2total int, collocation int)
APPENDIX B (XML FORMATS)

This appendix contains the format of two of the XML files mentioned in this thesis.

The first, SiteMap XML, is created when SiteMap maps Webpages and saves the pages.

The second, Term Cluster XML, is created when the user selects Export from the Cluster Dialog in the SiteGraph application.

SiteMap XML

<root>
  <page>
    <url><![CDATA[http://www.foxsports.com:80]]></url>
    <id>0</id>
    <linksto>
      <link>
        <pointsto>1</pointsto>
        <count>1</count>
      </link>
      ... repeat for all links to
    </linksto>
    <linksfrom>
      <link>
        <pointsto>28</pointsto>
        <count>2</count>
        <terms>
          <term><![CDATA[BEN]]></term>
          <term><![CDATA[MALLER]]></term>
          <term><![CDATA[RUMORS]]></term>
          <term><![CDATA[NOTES]]></term>
        </terms>
      </link>
      ... repeat for all links from
    </linksfrom>
    <Terms>
      <TotalCount>4</TotalCount>
      <Term>
        <Text><![CDATA[BEN]]></Text>
        <Count>1</Count>
      </Term>
      ... repeat for all terms
    </Terms>
  </page>
  ... repeat for all pages
</root>

Term Cluster XML

<split>
  <split>
    <word><![CDATA[MIDI]]></word>
    <word><![CDATA[AUDIO]]></word>
  </split>
  <split>
    <word><![CDATA[LAB]]></word>
    <word><![CDATA[CREATIVE]]></word>
  </split>
  <word><![CDATA[SORT]]></word>
</split>
# APPENDIX C (RESULTS)

## Human Judged Taxonomies

### Computer Hardware

*topic-cutoff: 0.004*

*topic-trim-cutoff: 0.3*

<table>
<thead>
<tr>
<th>HARDWARE</th>
<th>CARD:USB</th>
<th>VIDEO</th>
<th>GEFORCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>TV</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>SONY</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>CAMERA</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>HD</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>LITE-ON</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>ATI:RADEON</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>SOUND</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>PCI</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>PC:USB REVIEW</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>SYSTEM SERVER</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>WINDOW</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>CISCO</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>XP</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>PRO</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>LAPTOP</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>APPLE</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>RAM</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>MOTHERBOARD</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>INTEL</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>SOCKET</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>MEMORY</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>BOX</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>512MB</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>DDR</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>OCZ</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>1GB</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>256MB</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>RAT</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>FLASH:USB</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>2GB</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>CPU</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>FAN</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>128MB</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>NETWORK:USB</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>WIRELESS</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>NETGEAR</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>FIREWALL</td>
</tr>
<tr>
<td>HARDWARE CARD:USB</td>
<td>VIDEO:PCI</td>
<td>GEFORCE</td>
<td>MICROSOFT</td>
</tr>
</tbody>
</table>

*topic-cutoff: 0.005*

*topic-trim-cutoff: 0.3*
<table>
<thead>
<tr>
<th>SPORT</th>
<th>NFL:PHOTO</th>
<th>REGISTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATE</td>
<td>OKLAHOMA</td>
<td>SPORTSNATION</td>
</tr>
<tr>
<td>QUICK:HIT</td>
<td>NHL:TEAM</td>
<td>RANK</td>
</tr>
<tr>
<td>ALL-STAR</td>
<td>ROSTER</td>
<td>STAR</td>
</tr>
<tr>
<td>MLB:PHOTO:TEAM</td>
<td>DEAL</td>
<td>BOND</td>
</tr>
<tr>
<td>RED</td>
<td>ODD</td>
<td>SIGN</td>
</tr>
<tr>
<td>HALL</td>
<td>STAT</td>
<td>TRANSACTION</td>
</tr>
<tr>
<td>SPORT</td>
<td>NFL:PLAYER</td>
<td>REGISTER</td>
</tr>
<tr>
<td>----------</td>
<td>------------</td>
<td>----------</td>
</tr>
<tr>
<td>NHL:TEAM</td>
<td>RANK</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALL-STAR</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ROSTER</td>
<td></td>
</tr>
<tr>
<td></td>
<td>STAR</td>
<td></td>
</tr>
<tr>
<td>MLB:PLAYER:TEAM</td>
<td>DEAL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>BOND</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RED</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ODD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SIGN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HALL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>STAT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TRANSACTION</td>
<td></td>
</tr>
<tr>
<td></td>
<td>INJURY</td>
<td></td>
</tr>
<tr>
<td></td>
<td>STAND</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PHOTO</td>
<td></td>
</tr>
<tr>
<td>NBA:PLAYER:TEAM</td>
<td>DRAFT</td>
<td>OKLAHOMA</td>
</tr>
<tr>
<td></td>
<td>STATE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SPORTSNATION</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HISTORY</td>
<td></td>
</tr>
<tr>
<td>NCAA:PLAYER:TEAM</td>
<td>STAT</td>
<td>STATISTIC</td>
</tr>
<tr>
<td></td>
<td>LEADER</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FUTURE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>STAND</td>
<td></td>
</tr>
<tr>
<td></td>
<td>STATE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FLORIDA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MICHIGAN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TEXA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WOMEN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WNBA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>STATE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FLORIDA</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-------</td>
<td>-----------</td>
</tr>
<tr>
<td>ABC</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IRAQ:FULL</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEALTH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIRD:FLU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VIDEO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LIBRARY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEART</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DRUG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NEW WORLD</td>
<td>ROAD</td>
<td>HEALTH:VIDEO</td>
</tr>
<tr>
<td>-----------</td>
<td>------</td>
<td>--------------</td>
</tr>
<tr>
<td></td>
<td>RETIREMENT</td>
<td>BIRD:FLU</td>
</tr>
<tr>
<td></td>
<td>IRAN</td>
<td>LIBRARY</td>
</tr>
<tr>
<td></td>
<td>FORECAST</td>
<td>HEART</td>
</tr>
<tr>
<td></td>
<td>CANADIAN</td>
<td>DRUG</td>
</tr>
<tr>
<td></td>
<td>WINDOW:VISTA</td>
<td>MOM:OLDEST</td>
</tr>
<tr>
<td></td>
<td>TEMPERATURE</td>
<td>CLINIC</td>
</tr>
<tr>
<td></td>
<td>PERSON:OLDEST</td>
<td>LABOR</td>
</tr>
<tr>
<td></td>
<td>BIG</td>
<td>CHINA</td>
</tr>
<tr>
<td></td>
<td>MENTAL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ENTERTAINMENT:VIDEO</td>
<td>TV</td>
</tr>
<tr>
<td></td>
<td>SHOW</td>
<td>QUESTION</td>
</tr>
<tr>
<td></td>
<td>PHOTO</td>
<td></td>
</tr>
<tr>
<td></td>
<td>JOHN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TROUBLE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MISSION</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DOWNLOAD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALAN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>EXCLUSIVE</td>
<td></td>
</tr>
<tr>
<td>BIO</td>
<td>PICK</td>
<td>WALL</td>
</tr>
<tr>
<td>-----</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COVERAGE</td>
<td>CONGRESS</td>
<td>BILL</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HAGEL:CRITICISM</td>
<td>LEAVE</td>
<td></td>
</tr>
<tr>
<td>DENOUNCE:ATTACK</td>
<td>PLEAD:TERROR:SUSPECT</td>
<td>GUILTY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>N.H:GOP:CHAIR</td>
</tr>
<tr>
<td>NASA</td>
<td>CHIEF</td>
<td>LAW</td>
</tr>
<tr>
<td>CHALLENGE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REVIEW</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HILLARY:CLINTON</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WIND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IOWA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HIT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAMPAIGN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WHITE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2008</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REPUBLICAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EARLY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOVE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRAVEL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POLICE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRUISE:EUROPEAN:FAMILY:TREN D:PORT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HOTEL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLUE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GREEN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DESTINATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GUIDE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CARNIVAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POLITIC:VIDEO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROFILE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HOUSE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRESIDENT:BUS H</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPORT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GOLF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOCCER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TENNI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FOOTBALL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FREE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BASKETBALL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HOCKEY:TEAM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BASEBALL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BOX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SI:MEDIA:KIT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>COLLEGE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BUSINESS:VIDEO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RETIREMENT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROFIT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U.S.:VIDEO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATOM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IRAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NUCLEAR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AMERICAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTERACTIVE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Subsumption Measure Results

These are the Subsumption results from the human judged taxonomies. Due to length they are incomplete. The format is:


Computer Hardware

topic-cutoff: 0.004

topic-trim-cutoff: 0.3

<table>
<thead>
<tr>
<th>HARDWARE:CARD</th>
<th>0.5:0.9846154:0.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>CARD:VIDEO:GEFORCE</td>
<td>0.9473684:0.7258064E-02:0.8036926</td>
</tr>
<tr>
<td>VIDEO:TV:0.9333333:0.2258064:0.8361028</td>
<td></td>
</tr>
<tr>
<td>VIDEO:SONY:0.5:0.141129:0.7688823</td>
<td></td>
</tr>
<tr>
<td>VIDEO:CAMERA:0.9615384:0.8064516:0.8009916</td>
<td></td>
</tr>
<tr>
<td>VIDEO:HD:0.8793104:0.2056452:0.81218</td>
<td></td>
</tr>
<tr>
<td>VIDEO:LITE-ON:0.3888889:2.822581E-02:0.7592686</td>
<td></td>
</tr>
<tr>
<td>VIDEO:ATI:0.8695652:8.064516E-02:0.7715237</td>
<td></td>
</tr>
<tr>
<td>CARD:SOUND:0.9949495:0.7695313:0.7938663</td>
<td></td>
</tr>
</tbody>
</table>
| ...

| MEMORY:FLASH | 0.9638554:0.7619048:0.7802655 |
| MEMORY:2GB | 0.8571429:0.1142857:0.7824621 |
| MEMORY:CPU:0.7405064:0.5571429:0.7812966 |
| MEMORY:FAN:0.9580838:0.7619048:0.7860746 |
| MEMORY:128MB | 0.9:0.571429:0.7890727 |
| HARDWARE:NETWORK:0.4401914:0.7076923:0.780127 |
| NETWORK:WIRELESS:0.8478261:0.3732058:0.7818195 |
| WIRELESS:NETGEAR:0.7692308:0.1086956:0.7815124 |
| WIRELESS:FIREFALL:0.9:0.5869565:0.7843335 |
| NETWORK:MICROSOFT:0.3392857:9.090909E-02:0.7739836 |
| Average Prob: 0.7739836 |
| Percent Correct: 74.4186046511628 |

topic-cutoff: 0.005

topic-trim-cutoff: 0.2

<table>
<thead>
<tr>
<th>HARDWARE:CARD</th>
<th>0.5:0.9846154:0.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>CARD:VIDEO:GEFORCE</td>
<td>0.9473684:0.7258064E-02:0.8036926</td>
</tr>
<tr>
<td>VIDEO:TV:0.9333333:0.2258064:0.8361028</td>
<td></td>
</tr>
<tr>
<td>VIDEO:SONY:0.5:0.141129:0.7688823</td>
<td></td>
</tr>
<tr>
<td>VIDEO:CAMERA:0.9615384:0.8064516:0.8009916</td>
<td></td>
</tr>
<tr>
<td>VIDEO:HD:0.8793104:0.2056452:0.81218</td>
<td></td>
</tr>
<tr>
<td>VIDEO:LITE-ON:0.3888889:2.822581E-02:0.7592686</td>
<td></td>
</tr>
<tr>
<td>VIDEO:ATI:0.8695652:8.064516E-02:0.7715237</td>
<td></td>
</tr>
<tr>
<td>VIDEO:128MB:0.8:6.451613E-02:0.7743713</td>
<td></td>
</tr>
<tr>
<td>CARD:SOUND:0.9949495:0.7695313:0.7944239</td>
<td></td>
</tr>
</tbody>
</table>
| ...

| MEMORY:FLASH | 0.9638554:0.7619048:0.7826575 |
| MEMORY:DRIVE:0.8926554:0.9518072:0.7382656 |
| FLASH:2GB:0.7857143:0.1325301:0.7399019 |
| FLASH:DISK:0.7647059:7.831325E-02:0.7407286 |
| FLASH:1GB:0.8529412:0.1746988:0.7443483 |
| FLASH:RAT:1:0.2710843:0.7523375 |
| FLASH:PORTABLE:0.7083333:0.2048193:0.751004 |
| FLASH:KINGSTON:0.718:4.33735E-02:0.7495039 |
| FLASH:512MB:0.7857143:0.1325301:0.7505385 |
| FLASH:256MB:0.7368421:8.433735E-02:0.7501581 |
| Average Prob: 0.7501581 |
| Percent Correct: 66.6666666666667 |
Sports

*topic-cutoff*: 0.003

*topic-trim-cutoff*: 0.2

```
SPORT:NFL:0.6592427:0.9704918:0.6592427
NFL:REGISTER:0.9885057:0.1915368:0.8238742
NFL:STATE:0.9487180:0.1648107:0.8654888
NFL:OKLAHOMA:0.9148936:0.5768398:0.89784
NFL:SPORTSNATION:1.0:0.3608018:0.8928571
NFL:QUICK:0.9:2.004454E-02:0.8914652
SPORT:NHL:0.6607143:0.9704918:0.8674392
NHL:RANK:0.9178082:0.4486607:0.8065677
NHL:ALL-STAR:0.9774011:0.3861607:0.8407344
NHL:ROSTER:0.9473684:4.017857E-02:0.8585067
NHL:STAR:0.8:4.464286E-02:0.8501487
SPORT:MLB:0.6584821:0.9704918:0.8261904
MLB:DEAL:0.9756098:8.925824E-02:0.8547056
MLB:BOND:0.95:4.241071E-02:0.8535132
...```

```
RECAP:CHART:4.347826E-02:3.225806E-02:0.7697395
SCHEDULE:BOX:0.9313725:0.5475504:0.7731069
BOX:SIMMON:0.9938272:0.7892157:0.7776114
BOX:NIRE:0.875:0.7892157:0.7795592
BOX:NIGHT:0.9638554:0.7843137:0.7831728
BOX:CHAMPION:0.4782609:5.392157E-02:0.7773091
BOX:TV:0.8858696:0.7990196:0.7793574
BOX:TITLE:0.3421053:6.372549E-02:0.7712601
BOX:PETER:1:0.7892157:0.7754191
BOX:SCORECARD:0.2753623:9.313726E-02:0.7664895
Average Prob: 0.7664895
Percent Correct: 67.85714285714295
```
Generalization/Specialization Quality Results

These are the Generalization/Specialization Quality results from the human judged taxonomies. The results are incomplete due to length. The format is:

Parent-term:Child-term:running Generalization/Specialization Quality running total

Computer Hardware

topic-cutoff: 0.004
topic-trim-cutoff: 0.3

HARDWARE:CARD:7.12972739711404E-03
CARD:VIDEO:7.83967893079147E-03
VIDEO:GEFORCE:7.9178856194704E-03
VIDEO:TV:8.22836235479488E-03
VIDEO:SONY:1.0177897033697E-02
VIDEO:CAMERA:1.0808249507103E-02
VIDEO:HD:1.1335672153172E-02
VIDEO:LITE-ON:1.18666427975409E-02
VIDEO:ATI:1.2091269823207E-02
CARD:SOUND:1.21714228740842E-02

...
WIRELESS: FIREWALL: 5.63740681631666E-02
NETWORK: MICROSOFT: 5.8024398785758E-02

Generalization/Specialization: 5.8024398785758E-02
Average / 43 : 1.34940557857153E-03

------------------------------------

topic-cutoff: 0.005

topic-trim-cutoff: 0.2

HARDWARE: CARD: 7.12972739711404E-03
CARD: VIDEO: 7.83967893079147E-03
VIDEO: VIDEO: 8.22836283547948E-03
VIDEO: SONY: 1.01778977033697E-02
VIDEO: LITE-ON: 1.1866427975409E-02
VIDEO: ATI: 1.2091669823207E-02
VIDEO: 128MB: 1.23781744079139E-02

MEMORY: FLASH: 3.4909850869634E-02
FLASH: DRIVE: 3.63516330847343E-02
FLASH: 2GB: 3.6779859598559E-02
FLASH: DISK: 0.03708234401599
FLASH: 1GB: 3.7428923127948E-02
FLASH: RAT: 3.7428923127948E-02
FLASH: PORTABLE: 3.83711409058686E-02
FLASH: KINGSTON: 3.87223762740137E-02
FLASH: 512MB: 3.91987291491353E-02
FLASH: 256MB: 3.9522938252601E-02

..................................................

------------------------------------

Generalization/Specialization: 3.95422938252601E-02
Average / 36 : 1.09839705070167E-03

------------------------------------

Sports

topic-cutoff: 0.003

topic-trim-cutoff: 0.2

SPORT: NFL: 2.33138129311759E-02
NFL: REGISTER: 2.3501751483286E-02
NFL: STATE: 2.42381564768176E-02
NFL: OMAHA: 2.49611564064423E-02
NFL: SPORTSNATION: 2.49611564064423E-02
NFL: QUICK: 2.51404053851315E-02
SPORT: NHL: 4.83294614956872E-02
NHL: RANK: 0.051588214670321
NHL: ALL-STAR: 5.23357349798667E-02
NHL: ROSTER: 5.25197030924198E-02

..................................................

------------------------------------

Generalization/Specialization: 0.172163091059452
Average / 40 : 4.30407727648631E-03

------------------------------------

topic-cutoff: 0.003

topic-trim-cutoff: 0.4

SPORT: NFL: 2.33138129311759E-02
NFL: REGISTER: 2.3501751483286E-02
SPORT: NHL: 4.66908112588843E-02
NHL: RANK: 4.99495644335181E-02
NHL: ALL-STAR: 5.06970847430638E-02
NHL: ROSTER: 5.08810528556169E-02
NHL: STAR: 5.17246798532486E-02
SPORT: MLB: 7.50241051102732E-02
MLB: DEAL: 7.52108130458425E-02
MLB: BOND: 7.5395038636583E-02
...
RECAP: CHART: 0.19825867787979
SCHEDULE: BOX: 0.200812710663505
BOX: SIMMON: 0.201000618280672
BOX: WIRE: 0.205064540876345
BOX: NIGHT: 0.206177985303137
BOX: CHAMPION: 0.207711716121927
BOX: TITLE: 0.214081639142079
BOX: PETER: 0.214081639142079
BOX: SCORECARD: 0.218713628631962
------------------------------------
Generalization/Specialization: 0.218713628631962
Average / 56 : 3.90560051128504E-03
------------------------------------

News

**topic-cutoff**: 0.005
**topic-trim-cutoff**: 0.2

NEW: WORLD: 1.10615933828328E-03
WORLD: ROAD: 1.37084361264356E-03
WORLD: RETIREMENT: 2.3295442448661E-03
WORLD: IRAN: 2.3295442448661E-03
WORLD: FORECAST: 2.3295442448661E-03
WORLD: CANADIAN: 2.3295442448661E-03
WORLD: WINDOW: 4.05097131513425E-03
WORLD: TEMPERATURE: 4.05097131513425E-03
WORLD: PERSON: 4.05097131513425E-03
NEW: HEALTH: 4.88144567664206E-03
HEALTH: BIRD: 4.88144567664206E-03
HEALTH: VIDEO: 7.06221545255354E-03
HEALTH: LIBRARY: 7.06221545255354E-03
HEALTH: HEART: 7.06221545255354E-03
HEALTH: DRUG: 7.06221545255354E-03
HEALTH: MOM: 7.06221545255354E-03
HEALTH: CLINIC: 7.06221545255354E-03
HEALTH: LABOR: 7.06221545255354E-03
HEALTH: BIG: 7.06221545255354E-03
HEALTH: CHINA: 1.65996691820336E-02
HEALTH: MENTAL: 1.65996691820336E-02

------------------------------------
Generalization/Specialization: 1.65996691820336E-02
Average / 21 : 7.90460437239696E-04
------------------------------------

**topic-cutoff**: 0.003
**topic-trim-cutoff**: 0.2

NEW: WORLD: 1.10615933828328E-03
WORLD: ROAD: 1.37084361264356E-03
WORLD: RETIREMENT: 2.3295442448661E-03
WORLD: IRAN: 2.3295442448661E-03
WORLD: FORECAST: 2.3295442448661E-03
WORLD: CANADIAN: 2.3295442448661E-03
WORLD: WINDOW: 4.05097131513425E-03
WORLD: TEMPERATURE: 4.05097131513425E-03
WORLD: PERSON: 4.05097131513425E-03
NEW: HEALTH: 4.88144567664206E-03
HEALTH: BIRD: 4.88144567664206E-03
HEALTH: VIDEO: 7.06221545255354E-03
HEALTH: LIBRARY: 7.06221545255354E-03
HEALTH: HEART: 7.06221545255354E-03
HEALTH: DRUG: 7.06221545255354E-03
HEALTH: MOM: 7.06221545255354E-03
HEALTH: CLINIC: 7.06221545255354E-03
HEALTH: LABOR: 7.06221545255354E-03
HEALTH: BIG: 7.06221545255354E-03
HEALTH: CHINA: 1.65996691820336E-02
HEALTH: MENTAL: 1.65996691820336E-02

------------------------------------
Generalization/Specialization: 1.65996691820336E-02
Average / 21 : 7.90460437239696E-04
------------------------------------
Human Judging Results

The human judged results show each judged pair and the average rating from all six human judges.

Computer Hardware

*topic-cutoff: 0.004*
*topic-trim-cutoff: 0.3*

<table>
<thead>
<tr>
<th>HARDWARE</th>
<th>CARD</th>
<th>2.50</th>
<th>HARDWARE</th>
<th>MOTHERBOARD</th>
<th>3.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>CARD</td>
<td>VIDEO</td>
<td>2.67</td>
<td>MOTHERBOARD</td>
<td>INTEL</td>
<td>2.67</td>
</tr>
<tr>
<td>VIDEO</td>
<td>GEFORCE</td>
<td>2.33</td>
<td>MOTHERBOARD</td>
<td>SOCKET</td>
<td>2.17</td>
</tr>
<tr>
<td>VIDEO</td>
<td>TV</td>
<td>2.33</td>
<td>HARDWARE</td>
<td>MEMORY</td>
<td>2.83</td>
</tr>
<tr>
<td>VIDEO</td>
<td>SONY</td>
<td>1.83</td>
<td>MEMORY</td>
<td>BOX</td>
<td>1.67</td>
</tr>
<tr>
<td>VIDEO</td>
<td>CAMERA</td>
<td>2.17</td>
<td>MEMORY</td>
<td>512MB</td>
<td>2.67</td>
</tr>
<tr>
<td>VIDEO</td>
<td>HD</td>
<td>2.17</td>
<td>MEMORY</td>
<td>DDR</td>
<td>3.00</td>
</tr>
<tr>
<td>VIDEO</td>
<td>LITE-ON</td>
<td>1.83</td>
<td>MEMORY</td>
<td>OCZ</td>
<td>2.67</td>
</tr>
<tr>
<td>VIDEO</td>
<td>ATI</td>
<td>2.50</td>
<td>MEMORY</td>
<td>1GB</td>
<td>2.67</td>
</tr>
<tr>
<td>CARD</td>
<td>SOUND</td>
<td>2.50</td>
<td>MEMORY</td>
<td>256MB</td>
<td>2.67</td>
</tr>
<tr>
<td>CARD</td>
<td>PCI</td>
<td>2.67</td>
<td>MEMORY</td>
<td>RATING</td>
<td>2.00</td>
</tr>
<tr>
<td>HARDWARE</td>
<td>PC</td>
<td>2.67</td>
<td>MEMORY</td>
<td>FLASH</td>
<td>3.00</td>
</tr>
<tr>
<td>PC</td>
<td>REVIEW</td>
<td>1.83</td>
<td>MEMORY</td>
<td>2GB</td>
<td>2.67</td>
</tr>
<tr>
<td>PC</td>
<td>SYSTEM</td>
<td>1.83</td>
<td>MEMORY</td>
<td>CPU</td>
<td>1.83</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>SERVER</td>
<td>2.17</td>
<td>MEMORY</td>
<td>FAN</td>
<td>2.00</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>WINDOWS</td>
<td>2.17</td>
<td>MEMORY</td>
<td>128MB</td>
<td>2.67</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>CISCO</td>
<td>2.67</td>
<td>HARDWARE</td>
<td>NETWORKING</td>
<td>2.33</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>XP</td>
<td>2.83</td>
<td>NETWORKING</td>
<td>WIRELESS</td>
<td>3.00</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>PRO</td>
<td>2.00</td>
<td>WIRELESS</td>
<td>NETGEAR</td>
<td>2.67</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>LAPTOP</td>
<td>2.67</td>
<td>WIRELESS</td>
<td>FIREWALL</td>
<td>2.33</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>APPLE</td>
<td>2.83</td>
<td>NETWORKING</td>
<td>MICROSOFT</td>
<td>2.17</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>RAM</td>
<td>2.50</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*topic-cutoff: 0.005*
*topic-trim-cutoff: 0.2*

<table>
<thead>
<tr>
<th>HARDWARE</th>
<th>CARD</th>
<th>2.50</th>
<th>PC</th>
<th>REVIEW</th>
<th>1.83</th>
</tr>
</thead>
<tbody>
<tr>
<td>CARD</td>
<td>VIDEO</td>
<td>2.67</td>
<td>PC</td>
<td>ATHLON</td>
<td>2.50</td>
</tr>
<tr>
<td>VIDEO</td>
<td>GEFORCE</td>
<td>2.33</td>
<td>PC</td>
<td>PENTIUM</td>
<td>2.50</td>
</tr>
<tr>
<td>VIDEO</td>
<td>TV</td>
<td>2.33</td>
<td>HARDWARE</td>
<td>MOTHERBOARD</td>
<td>3.00</td>
</tr>
<tr>
<td>VIDEO</td>
<td>SONY</td>
<td>1.83</td>
<td>MOTHERBOARD</td>
<td>INTEL</td>
<td>2.67</td>
</tr>
<tr>
<td>VIDEO</td>
<td>CAMERA</td>
<td>2.17</td>
<td>MOTHERBOARD</td>
<td>SOCKET</td>
<td>2.17</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------</td>
<td>------</td>
<td>-------------</td>
<td>----------</td>
<td>------</td>
</tr>
<tr>
<td>VIDEO</td>
<td>HD</td>
<td>2.17</td>
<td>HARDWARE</td>
<td>MEMORY</td>
<td>2.83</td>
</tr>
<tr>
<td>VIDEO</td>
<td>LITE-ON</td>
<td>1.83</td>
<td>MEMORY</td>
<td>BOX</td>
<td>1.67</td>
</tr>
<tr>
<td>VIDEO</td>
<td>ATI RADEON</td>
<td>2.50</td>
<td>MEMORY</td>
<td>FLASH</td>
<td>3.00</td>
</tr>
<tr>
<td>VIDEO</td>
<td>128MB</td>
<td>2.17</td>
<td>FLASH</td>
<td>DRIVE</td>
<td>2.00</td>
</tr>
<tr>
<td>CARD</td>
<td>SOUND</td>
<td>2.50</td>
<td>FLASH</td>
<td>2GB</td>
<td>2.83</td>
</tr>
<tr>
<td>SOUND</td>
<td>CREATIVE LAB</td>
<td>2.00</td>
<td>FLASH</td>
<td>DISK</td>
<td>2.00</td>
</tr>
<tr>
<td>SOUND</td>
<td>AUDIO</td>
<td>2.17</td>
<td>FLASH</td>
<td>1GB</td>
<td>2.83</td>
</tr>
<tr>
<td>SOUND</td>
<td>SORT</td>
<td>2.17</td>
<td>FLASH</td>
<td>RATING</td>
<td>1.83</td>
</tr>
<tr>
<td>SOUND</td>
<td>SIIG</td>
<td>1.67</td>
<td>FLASH</td>
<td>PORTABLE</td>
<td>2.33</td>
</tr>
<tr>
<td>SOUND</td>
<td>MIDI</td>
<td>2.00</td>
<td>FLASH</td>
<td>KINGSTON</td>
<td>2.33</td>
</tr>
<tr>
<td>SOUND</td>
<td>GOLD</td>
<td>2.00</td>
<td>FLASH</td>
<td>512MB</td>
<td>2.83</td>
</tr>
<tr>
<td>HARDWARE</td>
<td>PC</td>
<td>1.67</td>
<td>FLASH</td>
<td>256MB</td>
<td>2.83</td>
</tr>
</tbody>
</table>

**News**

*topic-cutoff: 0.005  topic-trim-cutoff: 0.2*

<table>
<thead>
<tr>
<th>NEWS</th>
<th>WORLD</th>
<th>2.50</th>
<th>HEALTH</th>
<th>VIDEO</th>
<th>2.20</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORLD</td>
<td>ROAD</td>
<td>2.00</td>
<td>HEALTH</td>
<td>LIBRARY</td>
<td>2.17</td>
</tr>
<tr>
<td>WORLD</td>
<td>RETIREMENT</td>
<td>2.17</td>
<td>HEALTH</td>
<td>HEART</td>
<td>2.50</td>
</tr>
<tr>
<td>WORLD</td>
<td>IRAN</td>
<td>2.67</td>
<td>HEALTH</td>
<td>DRUG</td>
<td>2.50</td>
</tr>
<tr>
<td>WORLD</td>
<td>FORECAST</td>
<td>2.17</td>
<td>HEALTH</td>
<td>OLDEST MOM</td>
<td>2.17</td>
</tr>
<tr>
<td>WORLD</td>
<td>CANADIAN</td>
<td>2.17</td>
<td>HEALTH</td>
<td>CLINIC</td>
<td>2.33</td>
</tr>
<tr>
<td>WORLD</td>
<td>WINDOWS VISTA</td>
<td>2.00</td>
<td>HEALTH</td>
<td>LABOR</td>
<td>2.50</td>
</tr>
<tr>
<td>WORLD</td>
<td>TEMPERATURE</td>
<td>2.17</td>
<td>HEALTH</td>
<td>BIG</td>
<td>1.67</td>
</tr>
<tr>
<td>WORLD</td>
<td>OLDEST PERSON</td>
<td>2.17</td>
<td>HEALTH</td>
<td>CHINA</td>
<td>1.83</td>
</tr>
<tr>
<td>NEWS</td>
<td>HEALTH</td>
<td>2.83</td>
<td>HEALTH</td>
<td>MENTAL</td>
<td>2.67</td>
</tr>
<tr>
<td>HEALTH</td>
<td>BIRD FLU</td>
<td>3.00</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*topic-cutoff: 0.003  topic-trim-cutoff: 0.2*

<table>
<thead>
<tr>
<th>NEWS</th>
<th>WORLD</th>
<th>2.5</th>
<th>WAR</th>
<th>HIRE GIULIANI</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORLD</td>
<td>ROAD</td>
<td>2</td>
<td>WAR</td>
<td>N.H. GOP CHAIR</td>
<td>2.17</td>
</tr>
<tr>
<td>WORLD</td>
<td>RETIREMENT</td>
<td>2.17</td>
<td>TV</td>
<td>REPORT</td>
<td>2.67</td>
</tr>
<tr>
<td>WORLD</td>
<td>IRAN</td>
<td>2.67</td>
<td>REPORT</td>
<td>EARTH</td>
<td>2.17</td>
</tr>
<tr>
<td>WORLD</td>
<td>FORECAST</td>
<td>2.17</td>
<td>REPORT</td>
<td>PROBE</td>
<td>2.17</td>
</tr>
<tr>
<td>WORLD</td>
<td>CANADIAN</td>
<td>2.17</td>
<td>REPORT</td>
<td>S.D. LAWMAKER ABORTION MEASURE</td>
<td>2</td>
</tr>
<tr>
<td>WORLD</td>
<td>WINDOWS VISTA</td>
<td>2</td>
<td>REPORT</td>
<td>NASA</td>
<td>2.17</td>
</tr>
<tr>
<td>WORLD</td>
<td>TEMPERATURE</td>
<td>2.17</td>
<td>REPORT</td>
<td>CHIEF</td>
<td>2.17</td>
</tr>
<tr>
<td>WORLD</td>
<td>OLDEST PERSON</td>
<td>2.17</td>
<td>REPORT</td>
<td>LAW</td>
<td>2.17</td>
</tr>
<tr>
<td>NEWS</td>
<td>HEALTH</td>
<td>2.83</td>
<td>REPORT</td>
<td>WOMEN</td>
<td>2.17</td>
</tr>
<tr>
<td>HEALTH</td>
<td>BIRD FLU</td>
<td>3</td>
<td>REPORT</td>
<td>CHALLENGE</td>
<td>2</td>
</tr>
<tr>
<td>--------</td>
<td>---------</td>
<td>---</td>
<td>--------</td>
<td>-----------</td>
<td>---</td>
</tr>
<tr>
<td>HEALTH</td>
<td>LIBRARY</td>
<td>2</td>
<td>ENTERTAINMENT</td>
<td>REVIEW</td>
<td>2.5</td>
</tr>
<tr>
<td>HEALTH</td>
<td>HEART</td>
<td>2.5</td>
<td>REVIEW</td>
<td>HILLARY CLINTON</td>
<td>2.17</td>
</tr>
<tr>
<td>HEALTH</td>
<td>DRUG</td>
<td>2.5</td>
<td>REVIEW</td>
<td>WIND</td>
<td>1.83</td>
</tr>
<tr>
<td>HEALTH</td>
<td>OLDEST MOM</td>
<td>2.17</td>
<td>REVIEW</td>
<td>IOWA</td>
<td>1.83</td>
</tr>
<tr>
<td>HEALTH</td>
<td>CLINIC</td>
<td>2.5</td>
<td>REVIEW</td>
<td>HIT</td>
<td>1.83</td>
</tr>
<tr>
<td>HEALTH</td>
<td>LABOR</td>
<td>2.33</td>
<td>REVIEW</td>
<td>CAMPAIGN</td>
<td>2.17</td>
</tr>
<tr>
<td>HEALTH</td>
<td>BIG</td>
<td>1.67</td>
<td>REVIEW</td>
<td>WHITE HOUSE</td>
<td>2.17</td>
</tr>
<tr>
<td>HEALTH</td>
<td>CHINA</td>
<td>1.83</td>
<td>REVIEW</td>
<td>2008</td>
<td>2</td>
</tr>
<tr>
<td>HEALTH</td>
<td>MENTAL</td>
<td>2.83</td>
<td>REVIEW</td>
<td>REPUBLICAN</td>
<td>2.17</td>
</tr>
<tr>
<td>NEWS</td>
<td>ENTERTAINMENT</td>
<td>2.5</td>
<td>REVIEW</td>
<td>EARLY</td>
<td>2</td>
</tr>
<tr>
<td>ENTERTAINMENT</td>
<td>TV</td>
<td>3</td>
<td>REVIEW</td>
<td>MOVE</td>
<td>2</td>
</tr>
<tr>
<td>TV</td>
<td>QUESTION</td>
<td>2</td>
<td>ENTERTAINMENT</td>
<td>POLICE</td>
<td>2</td>
</tr>
<tr>
<td>QUESTION</td>
<td>AP</td>
<td>1.67</td>
<td>NEWS</td>
<td>TRAVEL</td>
<td>2.5</td>
</tr>
<tr>
<td>QUESTION</td>
<td>CHENEY</td>
<td>1.83</td>
<td>TRAVEL</td>
<td>CRUISE</td>
<td>3</td>
</tr>
<tr>
<td>QUESTION</td>
<td>COLTS</td>
<td>1.83</td>
<td>TRAVEL</td>
<td>HOTEL</td>
<td>3</td>
</tr>
<tr>
<td>QUESTION</td>
<td>CASE</td>
<td>1.83</td>
<td>TRAVEL</td>
<td>BLUE</td>
<td>1</td>
</tr>
<tr>
<td>QUESTION</td>
<td>LIVE</td>
<td>1.83</td>
<td>TRAVEL</td>
<td>GREEN</td>
<td>1</td>
</tr>
<tr>
<td>QUESTION</td>
<td>VOTE</td>
<td>2</td>
<td>TRAVEL</td>
<td>DESTINATION</td>
<td>2.83</td>
</tr>
<tr>
<td>QUESTION</td>
<td>PART</td>
<td>1.83</td>
<td>TRAVEL</td>
<td>GUIDE</td>
<td>3</td>
</tr>
<tr>
<td>QUESTION</td>
<td>STAND</td>
<td>1.83</td>
<td>TRAVEL</td>
<td>CARNIVAL</td>
<td>2.5</td>
</tr>
<tr>
<td>QUESTION</td>
<td>TRIAL</td>
<td>2.17</td>
<td>NEWS</td>
<td>POLITICS</td>
<td>3</td>
</tr>
<tr>
<td>TV</td>
<td>SHOW</td>
<td>3</td>
<td>POLITICS</td>
<td>PROFILE</td>
<td>2.17</td>
</tr>
<tr>
<td>SHOW</td>
<td>PHOTO</td>
<td>2</td>
<td>POLITICS</td>
<td>HOUSE</td>
<td>2.67</td>
</tr>
<tr>
<td>SHOW</td>
<td>JOHN</td>
<td>1.67</td>
<td>POLITICS</td>
<td>PRESIDENT</td>
<td>2.83</td>
</tr>
<tr>
<td>SHOW</td>
<td>TROUBLE</td>
<td>1.67</td>
<td>NEWS</td>
<td>SPORTS</td>
<td>2.67</td>
</tr>
<tr>
<td>SHOW</td>
<td>MISSION</td>
<td>1.83</td>
<td>SPORTS</td>
<td>GOLF</td>
<td>3</td>
</tr>
<tr>
<td>SHOW</td>
<td>DOWNLOAD</td>
<td>2</td>
<td>SPORTS</td>
<td>SOCCER</td>
<td>3</td>
</tr>
<tr>
<td>SHOW</td>
<td>ALAN</td>
<td>1.83</td>
<td>SPORTS</td>
<td>TENNIS</td>
<td>3</td>
</tr>
<tr>
<td>SHOW</td>
<td>EXCLUSIVE</td>
<td>1.83</td>
<td>SPORTS</td>
<td>FOOTBALL</td>
<td>3</td>
</tr>
<tr>
<td>SHOW</td>
<td>BIO</td>
<td>2</td>
<td>SPORTS</td>
<td>FREE</td>
<td>1.83</td>
</tr>
<tr>
<td>SHOW</td>
<td>PICK</td>
<td>2</td>
<td>SPORTS</td>
<td>BASKETBALL</td>
<td>3</td>
</tr>
<tr>
<td>SHOW</td>
<td>WALL</td>
<td>2</td>
<td>SPORTS</td>
<td>HOCKEY TEAM</td>
<td>3</td>
</tr>
<tr>
<td>SHOW</td>
<td>FAMOUS</td>
<td>2</td>
<td>SPORTS</td>
<td>BASEBALL</td>
<td>3</td>
</tr>
<tr>
<td>SHOW</td>
<td>FACE</td>
<td>2</td>
<td>SPORTS</td>
<td>BOXING</td>
<td>3</td>
</tr>
<tr>
<td>SHOW</td>
<td>OFFICIAL</td>
<td>2</td>
<td>SPORTS</td>
<td>SI MEDIA KIT</td>
<td>2.17</td>
</tr>
<tr>
<td>SHOW</td>
<td>RULE</td>
<td>1.83</td>
<td>SPORTS</td>
<td>COLLEGE</td>
<td>2.83</td>
</tr>
<tr>
<td>SHOW</td>
<td>TURN</td>
<td>1.83</td>
<td>NEWS</td>
<td>BUSINESS</td>
<td>2.83</td>
</tr>
<tr>
<td>TV</td>
<td>WAR</td>
<td>2</td>
<td>BUSINESS</td>
<td>CAR</td>
<td>2.33</td>
</tr>
<tr>
<td>WAR</td>
<td>FULL</td>
<td>1.33</td>
<td>BUSINESS</td>
<td>RETIREMENT</td>
<td>2.67</td>
</tr>
<tr>
<td>WAR</td>
<td>COVERAGE</td>
<td>BUSINESSES</td>
<td>PLAN</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>----------</td>
<td>------------</td>
<td>------</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>WAR</td>
<td>CONGRESS</td>
<td>BUSINESSES</td>
<td>PROFIT</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>WAR</td>
<td>BILL</td>
<td>NEWS</td>
<td>U.S.</td>
<td>2.33</td>
<td></td>
</tr>
<tr>
<td>WAR</td>
<td>LEVEL</td>
<td>2</td>
<td>U.S.</td>
<td>ATOM</td>
<td>1.5</td>
</tr>
<tr>
<td>WAR</td>
<td>MISUSED ISRAEL BOMB</td>
<td>2.33</td>
<td>U.S.</td>
<td>IRAN</td>
<td>2</td>
</tr>
<tr>
<td>WAR</td>
<td>RESPOND TO CHENEY</td>
<td>2.17</td>
<td>U.S.</td>
<td>NUCLEAR</td>
<td>2.67</td>
</tr>
<tr>
<td>WAR</td>
<td>HAGEL CRITICISM</td>
<td>2</td>
<td>U.S.</td>
<td>AMERICAN</td>
<td>2.83</td>
</tr>
<tr>
<td>WAR</td>
<td>LEAVE</td>
<td>2</td>
<td>U.S.</td>
<td>INTERACTIVE</td>
<td>1.5</td>
</tr>
<tr>
<td>WAR</td>
<td>PASTRY CHEF</td>
<td>1.17</td>
<td>U.S.</td>
<td>MILITARY</td>
<td>2.67</td>
</tr>
<tr>
<td>WAR</td>
<td>DENOUNCE ATTACK</td>
<td>2.17</td>
<td>U.S.</td>
<td>SOLDIER</td>
<td>2.5</td>
</tr>
<tr>
<td>WAR</td>
<td>TERROR SUSPECT PLEAD</td>
<td>2.17</td>
<td>U.S.</td>
<td>WORK</td>
<td>2.17</td>
</tr>
<tr>
<td>WAR</td>
<td>GUILTY</td>
<td>2.17</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Sports**

*topic-cutoff: 0.003*

*topic-trim-cutoff: 0.2*

<table>
<thead>
<tr>
<th>SPORTS</th>
<th>NFL</th>
<th>3.00</th>
<th>MLB</th>
<th>INJURY</th>
<th>2.17</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFL</td>
<td>REGISTER</td>
<td>2.00</td>
<td>MLB</td>
<td>PLAYER</td>
<td>3.00</td>
</tr>
<tr>
<td>NFL</td>
<td>STATE</td>
<td>2.17</td>
<td>MLB</td>
<td>STANDINGS</td>
<td>2.50</td>
</tr>
<tr>
<td>NFL</td>
<td>OKLAHOMA</td>
<td>2.17</td>
<td>SPORTS</td>
<td>NBA</td>
<td>3.00</td>
</tr>
<tr>
<td>NFL</td>
<td>SPORTSNATION</td>
<td>2.17</td>
<td>NBA</td>
<td>DRAFT</td>
<td>2.67</td>
</tr>
<tr>
<td>NFL</td>
<td>QUICK HIT</td>
<td>1.67</td>
<td>NBA</td>
<td>HISTORY</td>
<td>2.50</td>
</tr>
<tr>
<td>SPORTS</td>
<td>NHL</td>
<td>2.67</td>
<td>NBA</td>
<td>TORONTO</td>
<td>2.33</td>
</tr>
<tr>
<td>NHL</td>
<td>RANK</td>
<td>2.17</td>
<td>SPORTS</td>
<td>NCAA</td>
<td>3.00</td>
</tr>
<tr>
<td>NHL</td>
<td>ALL-STAR</td>
<td>2.33</td>
<td>NCAA</td>
<td>STATS</td>
<td>2.83</td>
</tr>
<tr>
<td>NHL</td>
<td>ROSTER</td>
<td>2.33</td>
<td>NCAA</td>
<td>STANDINGS</td>
<td>2.50</td>
</tr>
<tr>
<td>NHL</td>
<td>STARS</td>
<td>2.33</td>
<td>STANDINGS</td>
<td>FLORIDA</td>
<td>2.67</td>
</tr>
<tr>
<td>SPORTS</td>
<td>MLB</td>
<td>2.83</td>
<td>STANDINGS</td>
<td>MICHIGAN</td>
<td>2.50</td>
</tr>
<tr>
<td>MLB</td>
<td>DEAL</td>
<td>2.17</td>
<td>STANDINGS</td>
<td>TEXAS</td>
<td>2.50</td>
</tr>
<tr>
<td>MLB</td>
<td>BONDS</td>
<td>2.17</td>
<td>NCAA</td>
<td>PLAYER</td>
<td>3.00</td>
</tr>
<tr>
<td>MLB</td>
<td>REDS</td>
<td>2.50</td>
<td>NCAA</td>
<td>WOMEN</td>
<td>2.67</td>
</tr>
<tr>
<td>MLB</td>
<td>ODDS</td>
<td>2.50</td>
<td>SPORTS</td>
<td>SCHEDULE</td>
<td>2.67</td>
</tr>
<tr>
<td>MLB</td>
<td>SIGN</td>
<td>2.17</td>
<td>SCHEDULE</td>
<td>RECAP</td>
<td>2.67</td>
</tr>
<tr>
<td>MLB</td>
<td>HALL</td>
<td>2.17</td>
<td>SCHEDULE</td>
<td>BOXING</td>
<td>2.67</td>
</tr>
<tr>
<td>MLB</td>
<td>STATS</td>
<td>2.83</td>
<td>SCHEDULE</td>
<td>VIDEO</td>
<td>1.50</td>
</tr>
<tr>
<td>MLB</td>
<td>TRANSACTION</td>
<td>2.17</td>
<td>SCHEDULE</td>
<td>HUNTING</td>
<td>1.50</td>
</tr>
</tbody>
</table>

*topic-cutoff: 0.003*

*topic-trim-cutoff: 0.4*

<table>
<thead>
<tr>
<th>SPORTS</th>
<th>NFL</th>
<th>3.00</th>
<th>STATS</th>
<th>LEADER</th>
<th>2.83</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFL</td>
<td>REGISTER</td>
<td>2.00</td>
<td>STATS</td>
<td>FUTURE</td>
<td>2.33</td>
</tr>
<tr>
<td>SPORTS</td>
<td>NHL</td>
<td>2.67</td>
<td>NCAA</td>
<td>STANDINGS</td>
<td>2.50</td>
</tr>
<tr>
<td>NHL</td>
<td>RANK</td>
<td>2.17</td>
<td>STANDINGS</td>
<td>STATE</td>
<td>2.33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>---------------</td>
<td>-------</td>
<td>---------------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>NHL</td>
<td>ALL-STAR 2.33</td>
<td>STANDINGS</td>
<td>FLORIDA 2.67</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NHL</td>
<td>ROSTER 2.33</td>
<td>STANDINGS</td>
<td>MICHIGAN 2.50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NHL</td>
<td>STARS 2.33</td>
<td>STANDINGS</td>
<td>TEXAS 2.50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPORTS</td>
<td>MLB 2.83</td>
<td>NCAA</td>
<td>WOMEN 2.67</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLB</td>
<td>DEAL 2.17</td>
<td>WOMEN</td>
<td>WNBA 2.83</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLB</td>
<td>BONDS 2.17</td>
<td>WOMEN</td>
<td>STATE 2.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLB</td>
<td>REDS 2.50</td>
<td>WOMEN</td>
<td>FLORIDA 1.83</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLB</td>
<td>ODDS 2.50</td>
<td>WOMEN</td>
<td>TV 2.17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLB</td>
<td>SIGN 2.17</td>
<td>WOMEN</td>
<td>SCOREBOARD 2.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLB</td>
<td>HALL 2.17</td>
<td>SPORTS</td>
<td>SCHEDULE 2.67</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLB</td>
<td>STATS 2.83</td>
<td>SCHEDULE</td>
<td>RECAP 2.67</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLB</td>
<td>TRANSACTION 2.17</td>
<td>RECAP</td>
<td>JONES 2.17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLB</td>
<td>INJURY 2.17</td>
<td>RECAP</td>
<td>FULL 2.17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLB</td>
<td>STANDINGS 2.50</td>
<td>RECAP</td>
<td>SHOT 2.17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLB</td>
<td>PHOTO 2.50</td>
<td>RECAP</td>
<td>CHART 2.17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPORTS</td>
<td>NBA 3.00</td>
<td>SCHEDULE</td>
<td>BOXING 2.67</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NBA</td>
<td>DRAFT 2.67</td>
<td>BOXING</td>
<td>SIMMON 2.17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DRAFT</td>
<td>OKLAHOMA 2.20</td>
<td>BOXING</td>
<td>WIRE 1.83</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DRAFT</td>
<td>STATE 2.17</td>
<td>BOXING</td>
<td>NIGHT 2.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DRAFT</td>
<td>SPORTSNATION 1.17</td>
<td>BOXING</td>
<td>CHAMPION 2.33</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DRAFT</td>
<td>HISTORY 1.83</td>
<td>BOXING</td>
<td>TV 2.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPORTS</td>
<td>NCAA 3.00</td>
<td>BOXING</td>
<td>TITLE 2.33</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NCAA</td>
<td>STATS 2.67</td>
<td>BOXING</td>
<td>PETER 1.83</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STATS</td>
<td>STATISTIC 2.83</td>
<td>BOXING</td>
<td>SCORECARD 2.33</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CURRICULUM VITAE

Joseph Paul Elliott

8204 Vaughn Mill Road
Louisville, KY 40228
jpelli01@louisville.edu
(502)-439-3311

EDUCATION

University of Louisville, Louisville, Kentucky
Thesis: Automatic Pure Anchor-Based Taxonomy Generation from the World Wide Web
Advisor: Dr. Antonio Badia

Bachelor of Science, Computer Engineering and Computer Science, April 2003
University of Louisville, Louisville, Kentucky

RESEARCH INTERESTS

- Artificial Intelligence
- Data Mining
- Information Sciences

WORK EXPERIENCE

Software Developer, 2001 - Present
Yum Brands Incorporated, Louisville, Kentucky