Thesaurus Entry Extraction from an On-line Dictionary *

Jan Jannink
Computer Science Department
Stanford University
Stanford CA, 94305, U.S.A.
jan@db.stanford.edu

Abstract ***The diversity and availability of information sources on the World Wide Web has set the stage for integration and reuse at an unparalleled scale. There remain significant hurdles to exploiting the extent of the Web's resources in a consistent, scalable and maintainable fashion. The autonomy and volatility of Web sources complicates maintaining wrappers consistent with the requirements of the data's target application. This paper describes the ArcRank model of relationships between nodes in a directed labeled graph, such as hypertext. The paper presents a ranking algorithm for directed arcs, and the algorithm for extraction of hierarchical relationships between words in a dictionary. Using ArcRank we create a thesaurus style tool to aid in the integration of texts and databases whose content is similar but whose terms are different. These algorithms complement handcrafted thesauri, by determining more complete relationships between words, although they are less specific. Exploiting hierarchies of relationships between words paves the way for broadening and related term queries in web-based repositories.***

Keywords: relationship rank, semantic heterogeneity, thesaurus, extraction

* This work was supported by a grant from the Air Force Office of Scientific Research (AFOSR).

1 Introduction

The principal obstacle in integrating information from multiple sources is their semantic heterogeneity. The most easily recognized form of heterogeneity is when different terms are used to mean the same thing: lexical heterogeneity. Even so, there is no algorithmic procedure to authoritatively resolve problems of lexical heterogeneity. However, we still desire assistance in determining semantically related terms.

Our experiments use an on-line version of the 1913 Webster's dictionary that is available through the Gutenberg Project [1]. The original dictionary is a corpus of over 50 MB containing some 112,000 terms, and over 2,000,000 words in the definitions alone. We have been working on the problem of automatically extracting thesaurus entries, using the following graph structure: each head word and definition grouping is a node, each word in a definition node is an arc to the node having that head word.

After accounting for the most common problems in constructing the graph, a naive script mis-assigns over five percent of the words, because of differences between the actual data in the dictionary and its assumed structure. Errors in the computation of the graph would affect any subsequent computation of related terms for the thesaurus application. Therefore, we set a goal of 99% accuracy in the conversion of the dictionary data to a graph structure.

Using a novel algebraic extraction technique we were able to generate such a graph structure and then use it to create thesaurus entries for all words defined in the structure including *stop words* such as 'the', 'a', 'and' that most sytems specifically list so as to ignore. The thesaurus engine, based on our relationship ranking technique, constructs more complete repositories than manually constructed thesauri, although they are less specific. It is a potentially important tool for systems integration experts.

1.1 Related work

Some early work on constructing taxonomies[2] and extracting *semantic primitives* [3] used a graph generated from the dictionary definitions. Examples of *lexical knowledge bases* that relate terms according to some two dozen relationships, are the handcrafted WordNet [4], and MindNet [5]. MindNet is generated by phrase parsing in the dictionary.

PageRank[6] is the algorithm that underlies the material in this paper. Algorithms that operate on a matrix representation of word graphs include LSI [7] and *hubs and authorities* [8]. WHIRL [9] attempts database integration using novel IR based textual similarity queries.

1.2 Motivation

The starting point for this work is the hypothesis that structural relationships between terms are relevant to their meaning. These relationships become interesting when all items in the domain of interest contain them, and are organized according to them. Dictionary definitions form a closed domain in the sense that the set of words used in definitions are defined elsewhere in the dictionary. This property leads to a directed labeled graph representation of the dictionary. Nodes of the graph model definitions, head words are labels for the nodes, and a word in a definition represents an arc to the node having that word as a label. Notable collections which are not closed include encyclopedias, which cover a set of terms equivalent to the dictionary nouns, and search engines, which return documents for all but stop words.

At first glance, the PageRank model of Web structure does not lend itself to direct application in non-hypertextual domains. However, we have found that a related model, which we call ArcRank, is useful for extracting relationships between words in a dictionary. This model expresses the importance of a word when used in the definition of another. The attraction of using the dictionary as a structuring tool is precisely that head words are distinguished terms for the definition text. This extra information allows types of analysis that are not currently performed in traditional data mining, and IR, where no term is assigned as 'head word' of a document. Interestingly, we now find that this new analysis may also be applied to document classification and the ranking of results of mining queries.

2 Background

In this section we present the basis of our dictionary structuring techniques. Before presenting the ArcRank measure, we present the PageRank algorithm, and the variants we have used in our experiments.

2.1 Graph Extraction

Substantial manipulation is required to bring the dictionary data into a format ready for generating a graph [10]. Head words and definitions are in a many to many relationship since head words have variant spellings and definitions have multiple differing senses. Other problems in the transformation process are listed below.

- syllable and accent markers in head words
- misspelled head words
- accents and special characters
- mis-tagged fields
- common abbreviations in definitions (etc.)
- stemming and irregular verbs (Hopelessness)
- multi-word head words (Water Buffalo)
- undefined words with common prefixes (Un-)
- undefined hyphenated and compound words (Sea-dog)

Table 1: PageRank

input: directed graph, *output*: scored node list

1. Make adjacency list representation of directed graph
2. Make rank array of size $|n|$ for graph nodes
3. Set (round 0) rank $p_{0s} = 1/n$ for all nodes s
4. While `rankchange` $>$ `threshold` (round i)
5. For nodes s in $\{1 \cdots |n|\}$ (ranking step)
6. For arcs $a_{s,t}$ in s's adjacency list a_s
7. Transfer rank $p_{is}/|a_s|$ from source s to target t
8. For nodes s in $\{1 \cdots |n|\}$ (adjustment step)
9. Normalize, if needed, rank p_{is} wrt to total rank
10. Compute `rankchange` from previous iteration
11. Return final values from rank array

For example, when a conjugated verb form appears as a head word we use it for generating graph arcs. Otherwise we stem definition words until we find a head word that matches. Also, whenever we find instances of a multi-word head word in the definitions, we prefer it over the individual words for generating a graph arc. Since words often appear multiple times in a single definition we allow multiple arcs between graph nodes. Dealing with undefined terms and spelling errors is the most complex issue in the graph generation, and accounts for the quasi-totality of the structural errors in the graph. In the following we define the algorithms that run on the graph structure.

2.2 PageRank

The PageRank algorithm forms the basis of the ranking technique described in this paper, and is important to define before discussing the ranking of arcs. Table 1 below is a pseudocode description of the algorithm:

This algorithm is a flow algorithm which assumes no capacity constraints on the arcs between nodes. All nodes begin with an initial ranking, in our case a constant $1/|n|$, where $|n|$ is the number of nodes in the graph. At each iteration, nodes distribute their rank to their neighbors on outgoing arcs, and receive rank from neighbors on incoming arcs. The total outgoing flow from a node is never greater than its rank, $\sum_t a_{s,t} \leq p_s$, nor is any individual $a_{s,t}$ ever less than zero. The intuition behind the flow is that more richly connected areas of the graph carry larger capacity, and therefore nodes in these areas maintain a higher rank. The rank flow of nodes in strongly connected aperiodic graphs is shown to converge to a steady state [11]. Steady state flow is desirable, because it allows us to assert stable relationships between nodes in the graph. In practice, we accept variability in the flow between nodes, so long as the total variability over the entire graph lies below a threshold.

In general graphs, nodes and clusters of nodes with only outgoing arcs act as sources which lose all of their rank. Likewise, nodes with incoming arcs only act as sinks for the rank of their neighbors. The dictionary graph contains both source and sink nodes: sources are words which are never used in other words' definitions, sinks are words whose definitions are not found in the dictionary. In our application sinks consist of misspellings, proper nouns such as geographical and Latin species names, and scientific formulae, which we do not consider. In PageRank the rank of sources, sinks and weakly connected clusters do not reflect their structural differences well. In our algorithm the final rank of a node should be defined in such a way that when any two nodes have a distinct pattern of connections, then their rank will differ. We adapt the algorithm from Table 1 in one of the following three ways so that sources and weakly connected clusters preserve some rank at each iteration.

1. redistribute $b\%(b/100)$ of total graph rank before each iteration
2. limit rank transfer to a fraction $1/c$ of a node's rank
3. add a self-arc $a_{t,t}$ (node t is both source and target) to nodes

By selecting a non zero threshold for termination of PageRank, and one of the above adaptations, we ensure that all graph nodes preserve a non zero rank. We show here that, given a node t, at iteration i with rank $p_{i t}$, the following holds:

Theorem 1 $\boxed{\forall t \in G, p_{i t} > 0}$

Proceeding by induction, we have: by definition, at the initial iteration, $p_{0 t} = 1/n > 0$*. Assuming the property holds at iteration* i*, the following holds:*

$$p_{i+1 t} = \{b/100, 1/c, p_{it}/(|a_t| + 1)\} + \sum_{v \neq t} a_{v,t}$$

Since, by definition all quantities on the right hand side are positive and greater than zero, $p_{i+1 t}$ *is greater than zero. As indicated by the equation, this property holds for each PageRank variant enumerated above.*

We see that PageRank for dictionary terms represents the transitive contribution of each term to the definitions of all of the dictionary terms. We capitalize on this property to compute the relative importance of terms with respect to each other. This measure is a feature of the arcs between nodes, or equivalently in the dictionary, the usage of terms in the definitions of others.

2.3 Relative Arc Importance

In the dictionary application, PageRank suffers from some inherent limitations. First of all, PageRank is inherently a node oriented algorithm. The top ranked nodes are the common conjunctions and prepositions, which convey little conceptual meaning, and are commonly considered stop words by other applications. It is clear that on its own, PageRank is insufficient to conceptually organize the dictionary structure. We may consider an extension to PageRank which assigns to each arc the amount of rank that flows across it at each iteration. As an absolute measure, this extension is also unsatisfactory, because it favors flows between the most highly ranked terms, that is, between stop words. Besides this obvious extension, there appears to be no self-evident technique to extract an absolute arc-based measure from PageRank.

However, our original goal is to identify the most important arcs for a given individual node. By casting our ranking problem in terms of our original goal we see that rather than an absolute measure, a relative measure between nodes is preferable. For any term in the dictionary, the words that signify the most in their definition should correspond to the arcs in the graph which are most significant in a ranking of arcs. Hence we arrive at the relative measure of arc relevance. Given an edge e, having source node s with rank p_s, target node t with rank p_t, and given $|a_s|$ outgoing arcs from s, the arc relevance r for e is defined as:

$$r_e = \frac{p_s/|a_s|}{p_t}$$

When s and t share several (m) edges $e_1 \ldots e_m$, we sum the arc ranks to compute the importance of t in the definition of s:

$$r_{s,t} = \sum_{e=1}^{m} \frac{p_s/|a_s|}{p_t}$$

$r_{s,t}$ measures the relative contribution of the rank of s to the rank of t which we show has desirable properties, such as:

Theorem 2 $\boxed{0 < r_{s,t} \leq 1}$

This follows directly from Theorem 1 and the definition of p_t*, since both numerator and denominator must be positive and* $p_t = \sum_v p_v/|a_v| = p_s/|a_s| + \sum_{v \neq s} p_v/|a_v| \Rightarrow p_t \geq p_s/|a_s|$*.*

Note that the arc importance measure is an indicator valid only in the immediate local vicinity of the end points of the arc. There is no reason to expect it to be globally commensurate. Having established an arc importance measure we are ready to present the ArcRank algorithm, and walk through a hierarchical set of relationships the algorithm uncovers.

3 ArcRank

In the previous section, we have computed a relative measure of arc importance. Here we show how to rank it with respect to both the source and target nodes, to promote arcs which are important to both endpoints. We discuss the repository we construct using ArcRank, and compare it to other systems.

3.1 ArcRank Algorithm overview

The ranking of an arc according to the arc importance metric defined above is typically different at the source and the target node. Indeed, it is possible for the highest arc importance value of arcs from a source node to be the lowest value for arcs coming into the target node. ArcRank, defined in Table 2 below, computes a mean of the ranked importance of arcs, so as to promote arcs which are important both to the source nodes and to the target nodes.

Table 2: ArcRank

input: triples (source s, target t, arc importance $v_{s,t}$)

1. given source s and target t nodes
2. at s, sort v_{s,t_j} and rank arcs $r_s(v_{s,t_j})$
3. at t, sort $v_{s_i,t}$ and rank arcs $r_t(v_{s_i,t})$
4. compute ArcRank: mean$(r_s(v_{s,t}), r_t(v_{s,t}))$
5. Rank Arcs *input*: sorted arc importance
   - sample values $\{0.9, 0.75, 0.75, 0.75, 0.6, 0.5, \ldots, 0.1\}$
   - equal values take same rank $\{1, \mathbf{2}, \mathbf{2}, \mathbf{2}, \ldots\}$
   - number ranks consecutively $\{1, 2, 2, 2, \mathbf{3}, \ldots\}$

Other rank numbering techniques resulted in skewed output. Competition style ranking, which counts equal values equally, but orders subsequent values differently, disadvantages arcs to nodes with many in-arcs. Given the same sample values from the above, the boldface value in the list here shows where this ranking differs: $\{1, 2, 2, 2, \mathbf{5}, 6, \ldots\}$. Also, computing rank as a fraction of the total number of ranks: $\{1/n, 2/n, \ldots, \mathbf{n/n}\}$ favors arcs to nodes with a larger number of distinct ranks.

The ArcRank algorithm is more space intensive than PageRank, because it is arc oriented, but is fast and easily made into a disk based version. It essentially requires two passes through the data, and storage for twice the number of arcs. In the course of developing ArcRank, we derived a further extension to PageRank. The idea is to vary according to the arc importance ratio the amount of a source node's rank transfered to the targets. Tuning this optimization properly strengthens strong relationships, weakens less important ones. The additional cost is minimal, and requires ranking arcs and summing ranks per node, before pushing value across arcs.

3.2 The Webster's Repository

The repository we have built [12] has a very general structure, and it is defined by usage alone. There are no preimposed limitations, based on grammatical models, as to how terms relate. As it is very general, the structure also sidesteps problems of parsing the part of speech for each term and handling general negation. This repository is the only one which does not exclude stop words, and as a result we are able to find that stop words most strongly relate to each other. On the down side, the type of relationship expressed in the repository is not always self evident, especially since many definitions and terms are now obsolete. Also, the accuracy of the ArkRank measure increases with the amount of data, and much of the dictionary contains very sparse definitions. Due to this sparseness we often find that ArcRank will promote arcs to lower ranked targets. Also, misleadingly, the sparseness of data makes a simple metric of ranking sources by the paucity of arcs work well, when it would otherwise fail.

3.3 Comparison to Other Systems

MindNet is not publicly available, but its scale is 159,000 head words and 713,000 relationships between head words. Its development began in 1992, and it supports 24 different relationships between terms. It appears that it suffers from problems, both in terms of accuracy and completeness of extraction.

WordNet has been in development since 1990, and its design has been elaborated since 1986. Its current revision, **WordNet 1.6** was released in 1998, and includes four principal data files, and a number of executables to aid in searching and displaying the data. Of the existing electronic lexical tools WordNet is the one that most closely resembles the Webster's repository.

The relationships WordNet defines between terms are more precise, as they were manually entered, however there are necessarily fewer of them, and they are far from exhaustive. Also, since the design of WordNet long preceded its implementation, artificial concepts, such as non-existant words, and artificial categorizations, , such as non-conforming adjectives, were introduced when the repository was built. These constructs are a valid ad hoc approach to make the terms conform to the design, but they do not arise out of the usage of the language. WordNet carefully distinguishes between senses of a term, and separates a term into multiple entries when it may be used as different parts of speech, i.e., to *run* vs. a computer *run* vs. a *run* salmon. The Webster's repository only distinguishes senses of a term based on usage, not on grammar. Another significant difference between the two structures is that the data in WordNet is separated by lexical categories, whereas the Webster's repository allows any relationship between terms to exist. Table 3 makes some simple numerical comparisons between the two systems.

Having compared the repositories numerically, it is necessary to illustrate with an example what the Webster's repository provides. Specifically, it relates terms without defining the type of relationship, just the importance of the relationship. The following section gives an example of terms relating to transportation.

4 Word Relationships

In this section we examine some subgraphs that emerge from the repository data after applying the ArcRank measure. For lack of space we can not cover the full array of relationships present in the dictionary, which extend even to stop words for the other repositories.

4.1 Browsing the Webster's Repository

It is instructive to browse through the repository to get an idea of how it organizes the dictionary terms. The example below is prompted by an interest in developing a transportation ontology to support logistics applications. We start at the term **Transport** as shown below in Figure 1. The general form of graphs generated using the repository, such as Figure 5, frame a term by terms used in its definition above and terms that use it in their definition below. These terms are placed from left to right in order of their ArcRank measure. No more than the two dozen most significant associated terms are displayed: the label for the central term contains a count of incoming and outgoing arcs of the form <*outgoing, incoming*>. In addition to the ArcRank measure on arcs, each term has an associated PageRank value. Arcs and Term borders are dotted when the arc's direction is the reverse of the PageRank ordering of its end points.

In Figure 1, which has been further pruned for clarity, we see that the term **Convey** is used in transport's definition. When we next examine the term graph for convey, Figure 2, we find transport, along with transported, and cargo which are also significant for the logistics ontology. Other terms in the set illustrate the more general nature of convey as compared to transport.

Further browsing in the repository takes us to the graph for **Carry** in Figure 3. Note how

Table 3: Comparison of Webster Repository and WordNet 1.6

Name	Size	Comment
Webster	96,800 terms	∼ four man months of effort
	112,897 distinct words	(including variant spellings)
		error rates
		<1% of original input (spelling errors, etc.)
		<0.05% incorrect arcs (hyphenation)
		<0.05% incorrect terms (spelling)
		0% artificial terms
WordNet 1.6	99,642 terms	2 profs, students, volunteers, 8-12 years
	173,941 word senses	(including numbers, repetition of terms)
	66,025 nouns	**disjoint files**
	12,127 verbs	
	17,915 adj.	
	3,575 adv.	
		error rates
		∼0.1% inappropriate classifications
		∼1-10% artificial & repeated terms

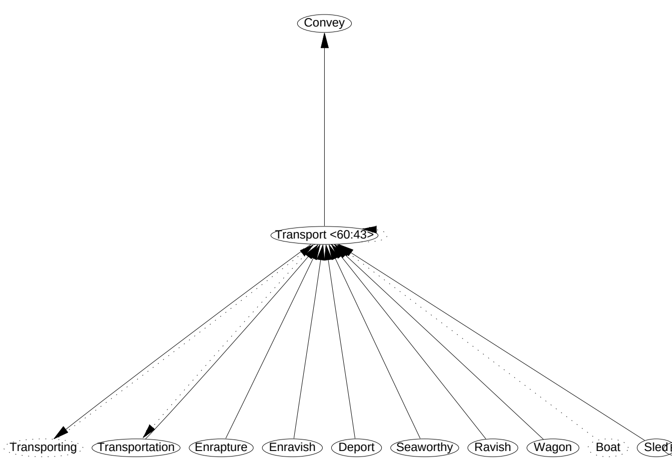

Figure 1: Terms Relating to Transport

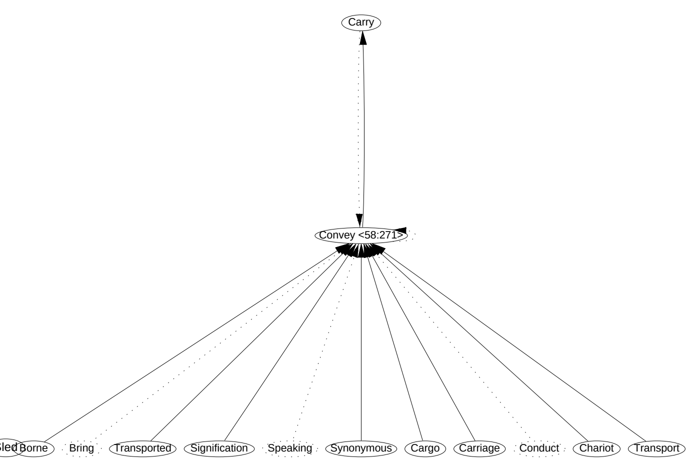

Figure 2: Convey Generalizes Transport

carry subsumes convey in the sense of transport, and that the term transported is also in its set of terms. We expect too that **Hold** expresses a more general notion relating to carry.

Starting from transport in the other direction, we select **Wagon** and consider Figure 4. Wagon is not a specialization of transport, although transport does subsume it: a wagon is one of a number of forms of transport. We see that terms such as **Car** and **Vehicle** also shown in Figure 4 represent the generalization relationship for wagon. Also, terms such as **Charioteer**, **Caravan** and **Wheelwright** relate to wagon without being specializations. bf Locomotive is however a specialization, and we next consider the graph in Figure 5.

The graph for locomotive illustrates a spectrum of relationships between terms, some of which are altogether unexpected, such as locomotive's relationship to the term **Appendix**. A glance at the definition of locomotive reveals that a reference to an illustration in the appendix of the dictionary appears inappropriately in the definition field of the term. The other associated terms all respect some subsuming or entailment relationship to locomotive.

Having traveled through a very small sam-

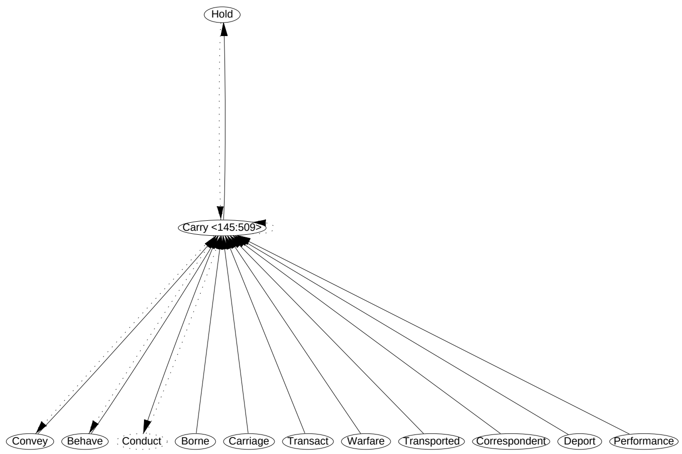

Figure 3: Carry Subsumes Convey

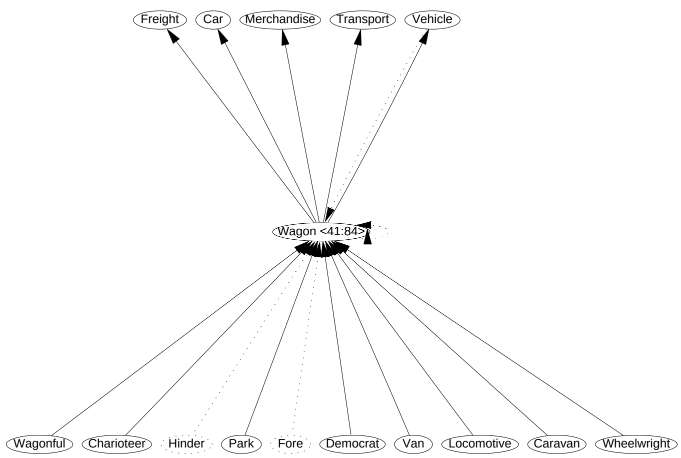

Figure 4: Wagon as a Means of Transport

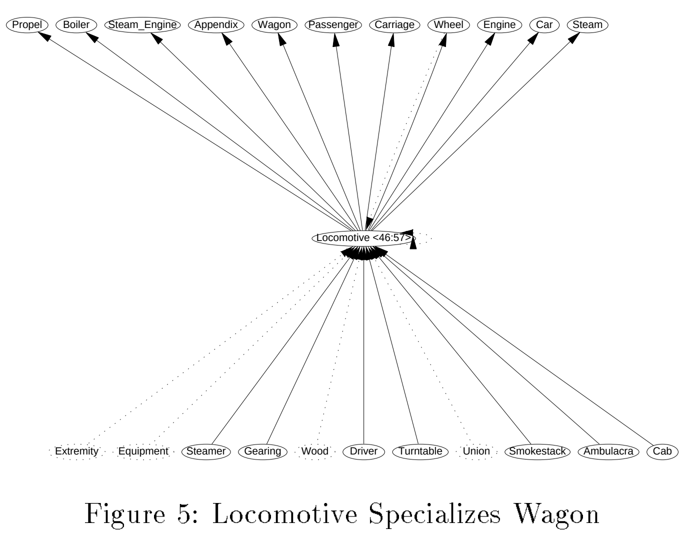

Figure 5: Locomotive Specializes Wagon

ple of the structure of the repository, it becomes clear that the ordering itself is not sufficient to automatically extract the significant terms relating to a given term. The algorithm to achieve this is the basis for the application we are building on top of the repository, and discussed in the following section. As it turns out the rankings provided by PageRank and ArcRank enable an efficient extraction procedure to maintain structure that is confirmed by relationships with other terms.

5 Applications

In this section we discuss applications of these new algorithms, and current directions of our research.

5.1 Relation Extraction

Having a repository with rank relationships between terms, it becomes possible to extract groups of related terms based on the strengths of their relationships. In particular, we are interested in extracting three relationships: *subsuming*, *specializing* and *kinship*. The kinship relationship is a similarity relationship broader than synonymy. We are able to achieve this extraction using a new iterative algorithm, based on the Pattern/Relation extraction algorithm [13], as follows in Table 4.

Table 4: Extract Relation

input graph with ArcRank computed, & seed arc set, *output* local hierarchy based on seed arc set

1. Compute set of nodes that contain arcs comparable to seed arc set
2. Threshold them according to ArcRank value
3. Extend seed arc set, when nodes contain further commonality
4. If node set increased in size repeat from 1.

The output of the algorithm computes a set of terms that are related by the strength of the associations in the arcs that they contain. These associations correspond to local hierarchies of subsuming and specializing relationships, and the set of terms are related by a kinship relationship. The algorithm is naturally self-limiting via the thresholds.

This approach allows us to distinguish senses of terms when they engender different structures according to the algorithm. Indeed, the senses of a word such as *hard*, are distinguished by the choice of association with *tough* and *severe*. Also, ranking the different senses of a term by the strength of its associations with other terms allows us to uncover the principal senses of a term.

We are currently investigating the utility of the ArcRank algorithm for traditional document classification applications, as well as to rank the association rules resulting from data mining queries. We are also using the results of the relation extraction algorithm to aid in the resolution of semantic heterogeneity in our ontology algebra research.

6 Conclusion

In this paper we have presented algorithms for ranking relationships represented in a graph structure. We have applied these algorithms to a graph extracted from an on-line dictionary to uncover the strongest relationships between dictionary terms, as given by term usage, rather than grammatical categorization. We consider this repository an adjunct, not a replacement, for handcrafted thesauri, to aid in the integration of disparate information sources, by reducing the effects of their lexical heterogeneity.

References

[1] PROMO.NET. Project Gutenberg. http://www.gutenberg.net/, 1999.

[2] R. Amsler. *The Structure of the Mirriam Webster Pocket Dictionary*. PhD thesis, University of Texas, Austin, 1980.

[3] D. P. Dailey. On the search for semantic primitives. *Computational Linguistics*, 12(4):306–307, 1986.

[4] G. A. Miller and *al.* Five papers on WordNet. Technical Report 43, Cognitive Science Laboratory, Princeton University, 1990.

[5] S. Richardson, W. Dolan, and L. Vanderwende. MindNet: acquiring and structuring semantic information from text. In *Proceedings of COLING '98*, 1998. ftp://ftp.research.microsoft.com/pub/tr/tr-98-23.doc.

[6] L. Page and S. Brin. The anatomy of a large-scale hypertextual web search engine. *Proceedings of the 7th Annual World Wide Web Conference*, 1998.

[7] S. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the Society for Information Science*, 41(6):391–407, 1990.

[8] J. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms*, 1998. http://simon.cs.cornell.edu/home/kleinber/auth.ps.

[9] William W. Cohen. Integration of heterogeneous databases without common domains using queries based on textual similarity. In *ACM SIGMOD '98*, pages 201–212. ACM, 1998.

[10] J. Jannink and G. Wiederhold. Ontology maintenance with an algebraic methodology: a case study. In *To appear: Proceedings, AAAI Workshop on Ontology Management*, 1999. http://www-db.stanford.edu/SKC/papers/summar.ps.

[11] R. Motwani and Raghavan P. *Randomized algorithms*. Cambridge University Press, New York NY, 1995.

[12] Jan Jannink. Webster's dictionary repository. http://skeptic.stanford.edu/data/, 1999.

[13] S. Brin. Extracting patterns and relations from the world wide web. http://www-db.stanford.edu/ sergey/booklist.html, 1998.