Is Hillary Rodham Clinton the President? Disambiguating Names across Documents

Yael RAVIN
T. J. Watson Research Center, IBM
P.O. Box 704,
Yorktown Heights, NY 10598
ravin@us.ibm.com

Zunaid KAZI
T. J. Watson Research Center, IBM
P.O. Box 704,
Yorktown Heights, NY 10598
Zhkazi@us.ibm.com

Abstract

A number of research and software development groups have developed name identification technology, but few have addressed the issue of cross-document coreference, or identifying the same named entities across documents. In a collection of documents, where there are multiple discourse contexts, there exists a many-to-many correspondence between names and entities, making it a challenge to automatically map them correctly. Recently, Bagga and Baldwin proposed a method for determining whether two names refer to the same entity by measuring the similarity between the document contexts in which they appear. Inspired by their approach, we have revisited our current cross-document coreference heuristics that make relatively simple decisions based on matching strings and entity types. We have devised an improved and promising algorithm, which we discuss in this paper.

Introduction

The need to identify and extract important concepts in online text documents is by now commonly acknowledged by researchers and practitioners in the fields of information retrieval, knowledge management and digital libraries. It is a necessary first step towards achieving a reduction in the ever-increasing volumes of online text. In this paper we focus on the identification of one kind of concept - names and the entities they refer to.

There are several challenging aspects to the identification of names: identifying the text strings (words or phrases) that express names; relating names to the entities discussed in the document; and relating named entities across documents. In relating names to entities, the main difficulty is the many-to-many mapping between them. A single entity can be referred to by several name variants: *Ford Motor Company*, *Ford Motor Co.*, or simply *Ford*. A single variant often names several entities: *Ford* refers to the car company, but also to a place (Ford, Michigan) as well as to several people: President Gerald Ford, Senator Wendell Ford, and others. Context is crucial in identifying the intended mapping. A document usually defines a single context, in which it is quite unlikely to find several entities corresponding to the same variant. For example, if the document talks about the car company, it is unlikely to also discuss Gerald Ford. Thus, within documents, the problem is usually reduced to a many-to-one mapping between several variants and a single entity. In the few cases where multiple entities in the document may potentially share a name variant, the problem is addressed by careful editors, who refrain from using ambiguous variants. If Henry Ford, for example, is mentioned in the context of the car company, he will most likely be referred to by the unambiguous *Mr. Ford*.

Much recent work has been devoted to the identification of names within documents and to linking names to entities within the document. Several research groups [DAR95, DAR98], as well as a few commercial software packages [NetOwl97], have developed name identification technology[1]. In contrast, few have investigated named entities across documents. In a collection of documents, there are multiple contexts; variants may or may not refer to the same entity; and ambiguity is a much greater problem.

[1] among them our own research group, whose technology is now embedded in IBM's Intelligent Miner for Text [IBM99].

Cross-document coreference was briefly considered as a task for the Sixth Message Understanding Conference but then discarded as being too difficult [DAR95].

Recently, Bagga and Baldwin [BB98] proposed a method for determining whether two names (mostly of people) or events refer to the same entity by measuring the similarity between the document contexts in which they appear. Inspired by their approach, we have revisited our current cross-document coreference heuristics and have devised an improved algorithm that seems promising. In contrast to the approach in [BB98], our algorithm capitalizes on the careful intra-document name recognition we have developed. To minimize the processing cost involved in comparing contexts we define compatible names -- groups of names that are good candidates for coreference -- and compare their internal structures first, to decide whether they corefer. Only then, if needed, we apply our own version of context comparisons, reusing a tool -- the Context Thesaurus -- which we have developed independently, as part of an application to assist users in querying a collection of documents.

Cross-document coreference depends heavily on the results of intra-document coreference, a process which we describe in Section 1. In Section 2 we discuss our current cross-document coreference. One of our challenges is to recognize that some “names” we identify are not valid, in that they do not have a single referent. Rather, they form combinations of component names. In Section 3 we describe our algorithm for splitting these combinations. Another cross-document challenge is to merge different names. Our intra-document analysis stipulates more names than there are entities mentioned in the collection. In Sections 4-5 we discuss how we merge these distinct but coreferent names across documents. Section 4 defines compatible names and how their internal structure determines coreference. Section 5 describes the Context Thesaurus and its use to compare contexts in which names occur. Section 6 describes preliminary results and future work.

1 Intra-Document Name Identification

Our group has developed a set of tools, called Talent, to analyze and process information in text. One of the Talent tools is Nominator, the name identification module [RW96]. We illustrate the process of intra-document name identification -- more precisely, name discovery -- with an excerpt from [NIST93].

```
...The professional conduct of lawyers in other jurisdictions is guided by American Bar Association rules ... The ABA has steadfastly reserved ... But Robert Jordan, a partner at Steptoe & Johnson who took the lead in ... "The practice of law in Washington is very different from what it is in Dubuque," he said. ... Mr. Jordan of Steptoe & Johnson ...
```

Before the text is processed by Nominator, it is analyzed into tokens - words, tags, and punctuation elements. Nominator forms a candidate name list by scanning the tokenized document and collecting sequences of capitalized tokens as well as some special lower-case ones. The list of candidate names extracted from the sample document contains:

```
American Bar Association
Robert Jordan
Steptoe & Johnson
ABA
Washington
Dubuque
Mr. Jordan of Steptoe & Johnson
```

Each candidate name is examined for the presence of conjunctions, prepositions or possessives (*'s*). These may indicate points at which the candidate name should be split into component names. A set of heuristics is applied to each candidate name, to split it into as many smaller independent names as are found in it. *Mr. Jordan of Steptoe & Johnson* is split into *Mr. Jordan* and *Steptoe & Johnson*. Without recourse to semantics or world knowledge, we do not always have sufficient evidence. In such cases we prefer to err on the conservative side and not split, so as to not lose any information. This explains the presence of "names" such as *American Television & Communications and Houston Industries Inc*. or *Dallas's MCorp and First RepublicBank and Houston's First City Bancorp. of Texas* in our intra-document results. We discuss later the splitting of these conjoined “names” at the collection level.

As the last step in name identification within the document, Nominator links all variants referring to the same entity. For example *ABA* is linked to *American Bar Association* as a possible abbreviation. Each linked group is categorized by an entity type and assigned a canonical string as identifier. The result for the sample text is shown below. Each canonical string is followed by its entity type (PL for PLACE; PR for PERSON) and the variant names linked to it.

```
American Bar Association [ORG]: ABA
Steptoe & Johnson [ORG]
Washington [PL]
Dubuque [PL]
Robert Jordan [PR]: Mr. Jordan
```

In a typical document, a single entity may be referred to by many name variants, which differ in their degree of potential ambiguity. To disambiguate highly ambiguous variants, we link them to unambiguous ones occurring within the document. Nominator cycles through the list of names, identifying 'anchors', or variant names that unambiguously refer to certain entity types. When an anchor is identified, the list of name candidates is scanned for ambiguous variants that could refer to the same entity. They are grouped together with the anchor in an equivalence group.

A few simple indicators determine the entity type of a name, such as *Mr.* for a person or *Inc*. for an organization. More commonly, however, several pieces of positive and negative evidence are accumulated in order to make this judgment. We have defined a set of obligatory and optional components for each entity type. For a human name, these components include a professional title (e.g., *Attorney General*), a personal title (e.g., *Dr*.), a first name, and others. The various components are inspected. Some combinations may result in a high negative score -- highly confident that this cannot be a person name. For example, if the name lacks a personal title and a first name, and its last name is marked as an organization word (e.g., *Department*), it will receive a high negative score. This is the case with *Justice Department* or *Frank Sinatra Building*. The same combination but with a last name that is not a listed organization word results in a low positive score, as for *Justice Johnson* or *Frank Sinatra*.

Names with low or zero scores are first tested as possible variants of names with high positive scores. However, if they are incompatible with any, they are assigned a weak entity type. Thus in the absence of any other evidence in the document, *Beverly Hills* or *Susan Hills* will be classified as PR? (PR? is preferred to PL? as it tends to be the correct choice most of the time.)

2 Current System for Cross-Document Coreference

The choice of a canonical string as the identifier for equivalence groups within each document is very important for later merging across documents. The document-based canonical string should be explicit enough to distinguish between different named entities, yet normalized enough to aggregate all mentions of the same entity across documents. Canonical strings of human names are comprised of the following parts, if found: first name, middle name, last name, and suffix (e.g., *Jr*.). Professional or personal titles and nicknames are not included as these are less permanent features of people's names and may vary across documents. Identical canonical strings with the same entity type (e.g., PR) are merged across documents. For example, in the [NIST93] collection, Alan Greenspan has the following variants across documents -- *Federal Reserve Chairman Alan Greenspan*, Mr. *Greenspan*, *Greenspan*, *Federal Reserve Board Chairman Alan Greenspan*, *Fed Chairman Alan Greenspan* -- but a single canonical string -- *Alan Greenspan*.

The current aggregation also merges near-identical canonical strings: it normalizes over hyphens, slashes and spaces to merge canonical names such as *Allied-Signal* and *Allied Signal*, *PC-TV* and *PC/TV*. It normalizes over "empty" words (*People's Liberation Army* and *People Liberation Army; Leadership Conference on Civil Rights* and *Leadership Conference of Civil Rights)*. Finally, it merges identical stemmed words of sufficient length (*Communications Decency Act* and *Communication Decency Act)*. Normalization is not allowed for people's names, to avoid combining names such as *Smithberg* and *Smithburg*.

Merging of identical names with different entity types is controlled by a table of aggregateable types. For example, PR? can merge with PL, as in *Beverly Hills* [PR?] and *Beverly Hills* [PL]. But ORG and PL cannot merge, so *Boston* [ORG] does not merge with *Boston* [PL]. As a further precaution, no aggregation occurs if the merge is ambiguous, that is, if a canonical name could potentially merge with more than one other canonical name. For example, *President Clinton* could be merged with *Bill Clinton*, *Chelsea Clinton*, or *Hillary Rodham Clinton*.

To prevent erroneous aggregation of different entities, we currently do not aggregate over different canonical strings. We keep the canonical place *New York* (city or state) distinct from the canonical *New York City* and *New York State*. Similarly, with human names: *Jerry O. Williams* in one document is separate from *Jerry Williams* in another; or, more significantly, *Jerry Lewis* from one document is distinct from *Jerry Lee Lewis* from another. We are conservative with company names too, preferring to keep the canonical name *Allegheny International* and its variants separate from the canonical name *Allegheny Ludlum* and its variant, *Allegheny Ludlum Corp*. Even with such conservative criteria, aggregation over documents is quite drastic. The name dictionary for 20MB of WSJ text contains 120,257 names before aggregation and 42,033 names after.

But conservative aggregation is not always right. We have identified several problems with our current algorithm that our new algorithm promises to handle.

1) Failure to merge -- often, particularly famous people or places, may be referred to by different canonical strings in different documents. Consider, for example, some of the canonical strings identified for President Clinton in our New York Times [NYT98] collection of 2330 documents:

```
Bill Clinton [PR]
Mr. Clinton [PR]
President Clinton [PR]
William Jefferson Clinton [PR]
Clinton [uncategorized]
```

Because of our decision not to merge under ambiguity (as mentioned above), our final list of names includes many names that should have been further aggregated.

2) Failure to split -- there is insufficient intra-document evidence for splitting "names" that are combinations of two or more component names, such as *ABC, Paramount and Disney*, or *B. Brown of Dallas County Judicial District Court*. Note that splitting is complex: sometimes even humans are undecided, for combinations such as *Boston Consulting Group in San Francisco*.

3) False merge -- due to an implementation decision, the current aggregation does not involve a second pass over the intra-document vocabulary. This means that canonical names are aggregated depending on the order in which documents are analyzed, with the result that canonical names with different entity types are merged when they are encountered if the merge seems unambiguous at the time, even though subsequent names encountered may invalidate it.

3 Splitting Names

We address the "splitting" problem first. The heuristics for splitting names within the document [WRC97] fail to address two kinds of combined names. First, there is a residue of names containing *and*, such as *Hoechst and Schering A.G.*, in which the *and* may or may not be part of the organization name. The cross-document algorithm to handle these is similar to the intra-document one: Iterate over the name string; break it into component strings at commas and *and*; verify that each component corresponds to an independently existing canonical string. If all do, split the name. The difference is that at the collection level, there are more canonical strings available for this verification. If the name is split, we repair the cross document statistics by folding the occurrence statistics of the combined form with those of each of the parts. On the collection level, we split strings like *AT&T Wireless and Primeco Personal Communications* and *Microsoft Network and AT&T Worldnet*, for which there was not enough evidence within the document.

More complex is the case of organization names of the form *X of Y* or *X in Y*, where *Y* is a place, such as *Fox News Channel in New York City* or *Prudential Securities in Shanghai*. The intra-

document heuristic that splits names if their components occur on their own within the document is not appropriate here: the short form may be licensed in the document only because the full form serves as its antecedent. We need evidence that the short form occurs by itself in other contexts. First, we sort these names and verify that there are no ambiguities. For example, it may appear that *Union Bank of Switzerland in San Francisco* is a candidate for splitting, since *Union Bank of Switzerland* occurs as a canonical name, but the existence of *Union Bank of Switzerland in New York* signals an ambiguity -- there are several distinct entities whose name starts with *Union Bank of Switzerland* and so no splitting applies. Similar ambiguity is found with *Federal District Court in New York*, *Federal District Court in Philadelphia*, etc. [2]

4 Merging Names

As discussed in [BB98], a promising approach to determining whether names corefer is the comparison of their contexts. However, since the cost of context comparison for all similar canonical strings would be prohibitively expensive, we have devised means of defining compatible names that are good candidates for coreference, based on knowledge obtained during intra-document processing. Our algorithm sorts names with common substrings from least to most ambiguous. For example, PR names are sorted by identical last names. The least ambiguous ones also contain a first name and middle name, followed by ones containing a first name and middle initial, followed by ones containing only a first name, a first initial and finally the ones with just a last name. PR names may also carry gender information, determined either on the basis of the first name (e.g. *Bill* but not *Jamie*) or a gender prefix (e.g. *Mr.*, but not *President*) of the canonical form or one of its variants. PL names are sorted by common initial strings. The least ambiguous have the pattern of <small place, big place>. By comparing the internal structure of these sorted groups, we are able to divide them into mutually exclusive sets (ES), whose incompatible features prevent any merging; and a residue of mergeable names (MN), which are compatible with some or all of the exclusive ones. For some of the mergeable names, we are able to stipulate coreference with the exclusive names without any further tests. For others, we need to compare contexts before reaching a conclusion.

To illustrate with an example, we collected the following sorted group for last name *Clinton*[3]:

```
William Jefferson Clinton[PR] (ES 1)
Hillary Rodham Clinton[PR]    (ES 2)
Larry Clinton [PR?]           (ES 3)
George Clinton [PR?]          (ES 4)
Chelsea Clinton [PR]          (ES 5)
```

The following MNs can be merged with these, based on compatibility, as indicated:

```
Bill Clinton [PR]         MN w/ 1, nickname
Bill Clinton [PR?]        MN w/ 1, nickname
Hillary Clinton [PR]      MN w/ 2, first name
President Clinton [PR](m)
        MN w/ 1,3,4, gender
President Clinton [PR](f)
        MN w/ 2,5,  gender
President Clinton [PR]    MN w/ all ESs
Mrs. Clinton [PR]         MN w/ 2,5,   gender
Mr. Clinton [PR]          MN w/ 1,3,4, gender
Clintons [uncategorized] MN w/ all ESs
```

There is too much ambiguity (or uncertainty) to stipulate coreference among the members of this sorted group. There is, however, one stipulated merge we apply to Bill Clinton [PR] and Bill Clinton [PR?]. We have found that when the canonical string is identical, a weak entity type can safely combine with a strong one. There are many cases of PR? to PR merging, some of PL? to ORG, (e.g., *Digital City*), and a fair number of PL? to PR, as in *Carla Hills*, *U.S.* and *Mrs. Carla Hills*. We discuss merging involving context comparison in the following section.

---

[2] Note that this definition of ambiguity is dependent on names found in the collection. For example, in the [NYT98] collection, the only *Prudential Securities in/of ...* found was *Prudential Securities in Shanghai*.

[3] Intra-document analysis identified *President Clinton* once as referring to a male, since *President Clinton* and *Mr. Clinton* were merged within the document(s); another time as referring to a female, since only *President Clinton* and *Mrs. Clinton* appeared in the document(s) in question and were merged; and a third *President Clinton*, based on documents where there was insufficient evidence for gender.

5 Comparing Contexts

The tool used for comparing contexts, the Context Thesaurus (CT), is a Talent tool that takes arbitrary text as input and returns a ranked list of terms that are related to the input text with respect to a given collection of documents. More specifically, the CT is used in an application we call Prompted Query Refinement [CB97], where it provides a ranked list of canonical strings found in the collection that are related to users' queries, out of which users may select additional terms to add to their queries.

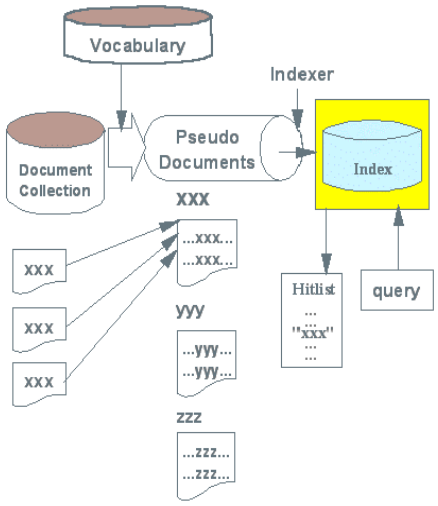

Figure 1 (Context Thesaurus)

The CT works with a collection concordance, listing the collection contexts in which a particular canonical string occurs. The size of the context is parameterized, but the default is usually three sentences -- the sentence where the string occurs, the preceding and following sentence within the same paragraph. We also collect occurrence statistics for each canonical string. The use of the concordance to generate relations among terms was inspired by the *phrase finder* procedure described in [JC94].

The CT is an ordinary information retrieval document index -- we use IBM's Net-Question query system [IBM99] -- which indexes special documents, referred to as "pseudo documents" (Figure 1). A pseudo document contains collection contexts in which a particular canonical string occurs. The title of the pseudo document is the canonical string itself. When a query is issued against the index, the query content is matched against the content of the pseudo documents. The result is a ranked list of pseudo documents most similar to the query. Recall that the titles of the pseudo documents are terms, or canonical strings. What is in fact returned to the user or the application looks like a ranked list of related terms. If the query itself is a single term, or a canonical string, the result is roughly a list of canonical strings in whose context the query canonical string occurs most often.

As an example, the query *American foreign policy in Europe* issued against a CT for the [NYT98] collection returns the following hit list:

```
American foreign policy, business interest,
American people, long decision-making
process, Clinton foreign policy, American
policy, Women in Foreign Service, Georgetown
University School of Foreign Service,
alliance official, senior American, foreign
corporation, conventional weapon, Central
American subsidiary, Armed Services
Committee, American court, foreign policy
adviser, foreign policy table, Serbs in
Bosnia
```

We can use a CT to simulate the effect of context comparisons, as suggested by [BB98]. To determine whether *President Clinton* in one document is the same person as *Bill Clinton* in another, we query the CT with each item. The Net-Question index returns a ranked hit list of documents (in our case canonical strings) in which each item occurs. The rank of a canonical string in the resulting hit list is an interpretation of the strength of association between the queried item and the hit-list canonical string. The underlying assumption for merging the two canonical forms is the fact that if they corefer, the contexts in which they each occur should contain similar canonical strings. Hence, if the two hit lists have a sufficient number of canonical strings in common (determined empirically to exceed 50%), we assert that the original items corefer.

We have identified four cases for merging that can benefit from context comparisons after all simpler methods have been exhausted.

1) One-to-one merging occurs when there are only two names -- one mergeable and one

exclusive. This is the case, for example, with two ORG canonical strings, where one is a substring of the other, as in *Amazon.com* and *Amazon.com Books*. Note that we cannot simply stipulate identity here because different organizations may share a common prefix, as *ANA Enterprises* and *ANA Hotel Singapore*, or *American Health Products* and *American Health Association*. We invoke queries to the CT using these canonical forms and aggregate if there is more than 50% overlap in the hit lists.

The query for *Amazon.com* returns:

```
Amazon.com, Amazon.com Books, Manesh Shah, growing competition, amazon.com, small number, Jeff Bezos, Kathleen Smith, online commerce, associates program, Yahoo 's Web, Robert Natale, classified advertising, Jason Green, audiotapes, Internet company, larger inventory, Day One, Society of Mind, Renaissance Capital
```

The query for *Amazon.com Books* returns:

```
small number, Amazon.com Books, audiotapes, Amazon.com, banned book, Manesh Shah, growing, competition, Jeff Bezos, bookstore, superstores, Kathleen Smith, online commerce, Yahoo 's Web, Robert Natale, classified advertising, Jason Green, larger inventory, Internet company, Renaissance Capital, Day One
```

Since there is an 80% match, there is more than ample evidence for merging the two names.

2) One-to-many merging occurs when there is one mergeable name but several distinct exclusive ones that are compatible with it. For example, *Cohen* [PR] can match either *Marc Cohen* [PR] or *William Cohen* [PR].

3) A many-to-one merging occurs quite frequently in the corpora we have experimented with. Several names of type PR, PR? or even uncategorized names share the same last name and have compatible first or middle names across documents. For example:

```
1: Madeleine Korbel Albright [PR] -  ES
2: Madeleine K. Albright [PR] - MN
3: Madeleine Albright [PR] - MN
```

Querying the CT results in a 60% match between 1 and 2, a 90% match between 2 and 3, and an 80% match between 3 and 1. Again, there is sufficient evidence for merging the three names.

4) The most complex case involves a many-to-many match, as illustrated by the Clinton example mentioned before.

Here are the results of the CT context matches[4]:

```
ES1: William Jefferson Clinton [PR]
ES2: Hillary Rodham Clinton [PR], Hillary Clinton [PR], Mrs. Clinton[PR]
ES3: Larry Clinton [PR?]
ES4: George Clinton[PR?]
ES5: Chelsea Clinton [PR]
ES6: Bill Clinton [PR?], Bill Clinton [PR], President Clinton [PR], Mr. Clinton [PR], Clintons [uncategorized]
```

Notice that *Bill Clinton* failed to merge with *William Jefferson Clinton*. This example suggests that failing to merge compatible names using the CT, we can use other information. For example, we can check if the mergeable canonical string is a variant name of the other, or if there is an overlap in the variant names of the two canonical strings. Our variant names contain titles and professional descriptions, such as *then-Vice President* or *Professor of Physics*, and checking for overlap in these descriptions will increase our accuracy, as reported in similar work by Radev and Mckeown [RM97].

6 Results and Future Work

We report here on preliminary results only, while we work on the implementation of various aspects of our new algorithm to be able to conduct a larger scale evaluation. We plan to evaluate our results with [BB98]'s proposed measure. So far, we have experimented with various examples from two collections - a small set of current New York Times articles [NYT98] and a larger collection of Wall Street Journal articles from 1990 [NIST93]. Here are some statistics comparing the two:

	NYT	**WSJ**
Collection size	16MB	70MB
Unique CFs	35,974	89,024
Unmerged PR/PR? with	10.413	41,241

[4] Note: for simplification, we combined the three *President Clinton* described above into a single canonical name, unmarked for gender.

identical strings		
Merged PR/PR? with identical strings	10,186	38,120

The distribution of distinct entities (or exclusive names) and mergeable names varies significantly from one sorted group to another. On one hand, there are the "famous" entities, such as President Bush (see below). These tend to have at least one exclusive name with a high number of occurrences. There are quite a few mergeable names -- a famous entity is assumed to be part of the reader's general knowledge and is therefore not always fully and formally introduced -- and a careful context comparison is usually required. On the other end of the scale, there are the non-famous entities. There may be a great number of exclusive names, especially for common last names but the frequency of occurrences is relatively low. There are 68 members in the sorted group for "Anderson" and 7 is the highest number of occurrences. Expensive processing may not be justified for low-frequency exclusive names. It seems that we can establish a tradeoff between processing cost versus overall accuracy gain and decide ahead of time how much disambiguation processing is required for a given application.

Canonical names with Bush as last name:

```
Neil Bush [PR] (m)        freq.: 309  (ES  1)
Senior Bush [PR]          freq.:  14  (ES  2)
Prescott Bush [PR] (m)    freq.:  13  (ES  3)
Lips Bush [PR] (m)        freq.:  10  (ES  4)
Top Bush [PR] (m)         freq.:   9  (ES  5)
Frederick Bush [PR] (m)   freq.:   7  (ES  6)
Jeb Bush [PR](m)          freq.:   5  (ES  7)
James Bush [PR] (m)       freq.:   4  (ES  8)
Keith Bush [PR?] (m)      freq.:   2  (ES  9)
George W. Bush [PR?] (m)  freq.:   2  (ES 10)
Charles Bush [PR?] (m)    freq.:   1  (ES 11)
Marvin Bush [PR?] (m)     freq.:   1  (ES 12)
Nicholas Bush [PR?] (m)   freq.:   1  (ES 13)
Marty Bush [PR?]          freq.:   1  (ES 14)

George Bush [PR] (m)   freq.: 861(MN w/  10)
President Bush [PR](m) freq: 1608(MN w/1-14)
then-Vice President Bush [PR](m) freq.:  12
       (MN w/ 1-14)
Mr. Bush [PR]          freq.: 5  (MN w/1-14)
Vice President Bush [PR] (m) freq.: 2
       (MN w/ 1-14)

Barbara Bush [PR?] (f)    freq.:  29  (ES 15)
Mary K. Bush [PR] (f)     freq.:  18  (ES 16)
Nancy Bush [PR?] (f)      freq.:   1  (ES 17)
Sharon Bush [PR?] (f)     freq.:   1  (ES 18)

Mrs. Bush [PR] freq.: 2    (MN w/ 14, 15-18)
Bush [uncategorized] freq.: 700 (MN w/ 1-18)
Bush [PR]              freq.:   5 (MN w/ 1-18)
Congress and President Bush [PR] freq.: 5
       (MN w/ 1-18)
U.S. President Bush [PR] freq.:2 (MN w/1-18)
Dear President Bush [PR] freq.:1 (MN w/1-18)
```

Acknowledgements

Various members of our group contributed to the Talent tools, in particular Roy Byrd, who developed the current cross-document aggregation. Our thanks to Eric Brown for suggesting to use CT for context comparisons.

References

BB98. A. Bagga and B. Baldwin. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of COLING-ACL 1998*, pages 79-85.

CB97. J. Cooper and R. J. Byrd. Lexical navigation: Visually prompted query expansion and refinement. In *DIGLIB 97*, Philadelphia, PA, 1997.

DAR95. Tipster Text Program. *Sixth Message Understanding Conference (MUC-6).*

DAR98. Tipster Text Program. *Seventh Message Understanding Conference (MUC-7).*

IBM99. *http://www.software.ibm.com/data/iminer/.*

JC94 Y. Jing and W. B. Croft. An association thesaurus for information retrieval. In RIAO 94, pages 146-160, 1994.

NetOwl97. NetOwl Extractor Technical Overview (White Paper). *http://www.isoquest.com/,* March 1997.

NIST93. Tipster Information-Retrieval Text Research Collection', CD-ROM, NIST, Gaithersburg, Maryland.

NYT98. Articles from http://www.nytimes.com/.

RM97. D. R. Radev and K. R. McKeown. Building a Generation Knowledge Source using Internet-Accessible Newswire. In *5th Conference on Applied Natural Language Processing*, pages 221-228, 1997.

RW96. Y. Ravin and N. Wacholder. Extracting Names from Natural-Language Text. IBM Research Report 20338. 1996.

WRC97. N. Wacholder, Y. Ravin and M. Choi. Disambiguation of proper names in text. In *5th Conference on Applied Natural Language Processing*, pages 202-208, 1997.