DESCRIPTION OF THE UPENN CAMP SYSTEM AS USED FOR COREFERENCE

Breck Baldwin, Tom Morton, Amit Bagga, Jason Baldridge, Raman Chandraseker, Alexis Dimitriadis, Kieran Snyder, Magdalena Wolska,
Institute for Research in Cognitive Science
3401 Walnut St. 400C
Philadelphia, PA 19104. USA
Phone: (215) 898-0329
Fax: (215) 573-9247
Email: {breck,tsmorton}@linc.cis.upenn.edu

Introduction

In this paper we present some advances made to the CAMP system since it's inception for MUC-6. Although the infrastructure has been completely re-implemented, the architecture has remained fundamentally the same–consequently we will focus some advances we have made in our understanding of coreference and then discuss the performance of the system.

Scoring Coreference Output

Scoring the performance of a system is an extremely important aspect of coreference algorithm performance. The score for a particular run is the single strongest measure of how well the system is performing and it can strongly determine directions for further improvements. In this paper, we present several different scoring algorithms and detail their respective strengths and weaknesses for varying classes of processing. In particular, we describe and analyze the coreference scoring algorithm used to evaluate the coreference systems in the sixth Message Understanding Conference (MUC-6)[MUC-6, 95]. We also present two shortcomings of this algorithm. In addition, we present a new coreference scoring algorithm, our B-CUBED algorithm, which was designed to overcome the shortcomings of the MUC-6 algorithm.

Scoring in MUC-6/7: Vilain et al.

Prior to Vilain et al.'s coreference scoring algorithm [Vilain, 95] there had been a graph based scoring algorithm (Sundheim et al.) which produced unintuitive results for even very simple cases. [Vilain, 95] substituted a model-theoretic scoring algorithm which produced very intuitive results for the type of scoring desired in MUC-6. This algorithm computes computes the recall error by taking each equivalence class S (defined by the links in the answer key) and determining the number of coreference links m that would have to be added to the response to place all the entities in S into the same equivalence class in the response. Recall error then is the sum of m's divided by the number of links in the key. Precision error is computed by reversing the roles of the answer key and the response.

The full details of the algorithm are discussed next.

The Model Theoretic Approach To The Vilain et. al Algorithm[1]

[1]The exposition of this scorer has been taken nearly entirely from [Vilain, 95]

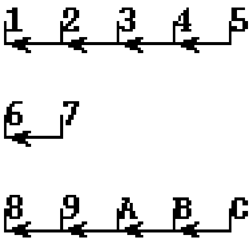

Figure 1: Truth

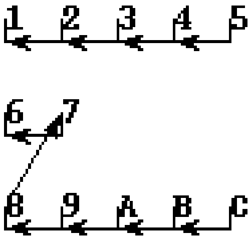

Figure 2: Response: Example 1

In the description of the model theoretic algorithm, the terms "key," and "response" are defined in the following way:

key refers to the manually annotated coreference chains (the truth).

response refers to the coreference chains output by a system.

An equivalence set is the transitive closure of a coreference chain. The algorithm computes recall in the following way.

First, let S be an equivalence set generated by the key, and let $R_1 \ldots R_m$ be equivalence classes generated by the response. Then we define the following functions over S:

- p(S) is a partition of S relative to the response. Each subset of S in the partition is formed by intersecting S and those response sets R_i that overlap S. Note that the equivalence classes defined by the response may include implicit singleton sets - these correspond to elements that are mentioned in the key but not in the response. For example, say the key generates the equivalence class S = {A B C D}, and the response is simply <A-B>. The relative partition p(S) is then {A B} {C} and {D}.

- c(S) is the minimal number of "correct" links necessary to generate the equivalence class S. It is clear that c(S) is one less than the cardinality of S, i.e.,

$$\mathrm{c(S)} = (|\mathrm{S}| - 1) \ .$$

- m(S) is the number of "missing" links in the response relative to the key set S. As noted above, this is the number of links necessary to fully reunite any components of the p(S) partition. We note that this is simply one fewer than the number of elements in the partition, that is,

$$\mathrm{m(S)} = (|\mathrm{p(S)}| - 1) \ .$$

Looking in isolation at a single equivalence class in the key, the recall *error* for that class is just the number of missing links divided by the number of correct links, i.e.,

$$\frac{\mathrm{m(S)}}{\mathrm{c(S)}} \ .$$

Recall in turn is

$$\frac{\mathrm{c(S)} - \mathrm{m(S)}}{\mathrm{c(S)}} \ ,$$

which equals

$$\frac{(|\mathrm{S}| - 1) - (|\mathrm{p(S)}| - 1)}{|\mathrm{S}| - 1} \ .$$

The whole expression can now be simplified to

$$\frac{|\mathrm{S}| - |\mathrm{p(S)}|}{|\mathrm{S}| - 1}. \tag{1}$$

Finally, extending this measure from a single key equivalence class to an entire set T simply requires summing over the key equivalence classes. That is,

$$\mathrm{R_T} = \frac{\sum(|\mathrm{S_i}| - |\mathrm{p(S_i)}|)}{\sum(|\mathrm{S_i}| - 1)} \ . \tag{2}$$

Precision is computed by switching the roles of the key and response in the above formulation.

Example

For example, let the key contain 3 equivalence classes as shown in Figure 1. Suppose Figure 2 shows a response. From Figure 3(I), the three equivalence classes in the truth, $\mathrm{S_1}$, $\mathrm{S_2}$, and $\mathrm{S_3}$, are {1, 2, 3, 4, 5}, {6, 7}, and {8, 9, A, B, C} respectively. And the partitions $\mathrm{p(S_1)}$, $\mathrm{p(S_2)}$, and $\mathrm{p(S_3)}$, with respect to the response, shown in Figure 3(II), are {1, 2, 3, 4, 5}, {6, 7}, and {8, 9, A, B, C} respectively. Using equation 2, the recall can now be calculated in the following way:

$$\mathrm{Recall} = \frac{(5-1) + (2-1) + (5-1)}{(5-1) + (2-1) + (5-1)} = 9/9 = 100\% \ .$$

Similarly, if the roles of the key and the response are reversed, then the equivalence classes in the truth, $\mathrm{S_1}$, and $\mathrm{S_2}$, are {1, 2, 3, 4, 5} and {6, 7, 8, 9, A, B, C}], and the partitions, $\mathrm{p(S_1)}$, and $\mathrm{p(S_2)}$, are {1, 2, 3, 4, 5} and [{6, 7} {8, 9, A, B, C}] respectively (Figure 3(III)). The precision can now be calculated as:

$$\mathrm{Precision} = \frac{(5-1) + (7-2)}{(5-1) + (7-1)} = 9/10 = 90\% \ .$$

Shortcomings of the Vilain et. al Algorithm

Despite the advances of the model-theoretic scorer, it yields unintuitive results for some tasks. There are two main reasons.

1. The algorithm does not give any credit for separating out singletons (entities that occur in chains consisting only of one element, the entity itself) from other chains which have been identified. This follows from the convention in coreference annotation of not identifying those entities that are markable as possibly coreferent with other entities in the text. Rather, entities are only marked as being coreferent

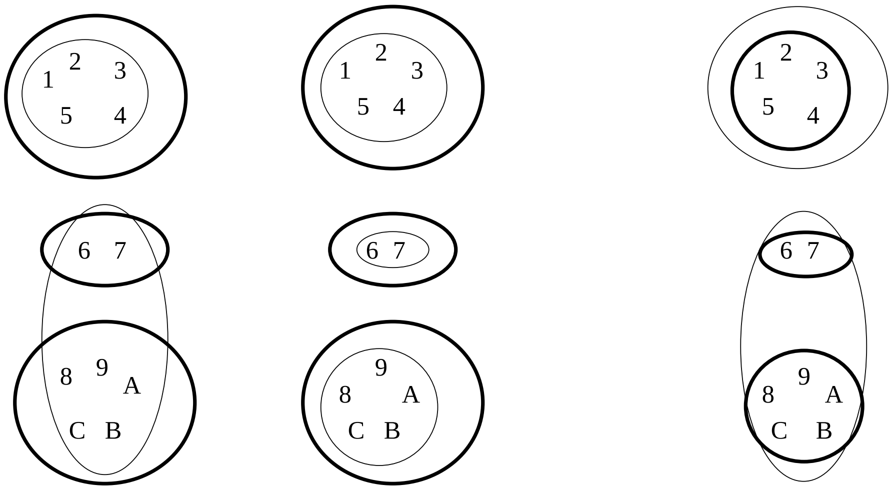

Figure I
Thick = key
Thin = response

Figure II
Thick = key
Thin = partition wrt/response

Figure III
Thick = partition wrt/key
Thin = response

Figure 3: Equivalence Classes and Their Partitions For Example 1

if they actually are coreferent with other entities in the text. This potential shortcoming could be easily enough overcome with different annotation conventions and with minor changes to the algorithm, but the decision to annotate singletons is a bit of a philosophical issue. On the one hand singletons do form equivalence classes, and those equivalence classes are significant in that they are NOT coreferent with another phrase in the text and they may play an important role in other equivalence classes out side the immediate text (as in cross document coreference). On the other hand, if coreference is viewed as being about the relations between entities, then perhaps is makes little sense to annotate and score singletons.

2. All errors are considered to be equal. The MUC scoring algorithm penalizes the precision numbers equally for all types of errors. It is our position that, for certain tasks, some coreference errors do more damage than others.

   Consider the following examples: suppose the truth contains two large coreference chains and one small one (Figure 1), and suppose Figures 2 and 4 show two different responses. We will explore two different precision errors. The first error will connect one of the large coreference chains with the small one (Figure 2). The second error occurs when the two large coreference chains are related by the errant coreferent link (Figure 4). It is our position that the second error is more damaging because, compared to the first error, the second error makes more entities coreferent that should not be. This distinction is not reflected in the [Vilain, 95] scorer which scores both responses as having a precision score of 90% (Figure 6).

Revisions to the Algorithm: Our B-CUBED Algorithm[2]

Our B-CUBED algorithm was designed to overcome the two shortcomings of the Vilain et. al algorithm. Instead of looking at the links produced by a system, our algorithm looks at the presence/absence of entities relative to each of the other entities in the equivalence classes produced. Therefore, we compute the precision and recall numbers for each entity in the document, which are then combined to produce final precision and recall numbers for the entire output. The formal model-theoretic version of our algorithm is discussed in the next section.

[2] The main idea of this algorithm was initially put forth by Alan W. Biermann of Duke University.

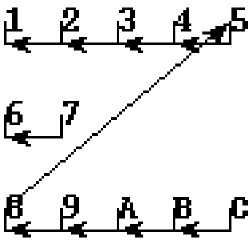

Figure 4: Response: Example 2

$$\text{Precision}_{\text{i}} = \frac{\textit{number of correct elements in the output chain containing entity}_i}{\textit{number of elements in the output chain containing entity}_i}$$

$$\text{Recall}_{\text{i}} = \frac{\textit{number of correct elements in the output chain containing entity}_i}{\textit{number of elements in the truth chain containing entity}_i}$$

Figure 5: Definitions for Precision and Recall for an Entity i

For an entity, i, we define the precision and recall with respect to that entity in Figure 5.

The final precision and recall numbers are computed by the following two formulae:

$$\text{Final Precision} = \sum_{\text{i=1}}^{\text{N}} \text{w}_{\text{i}} * \textit{Precision}_i$$

$$\text{Final Recall} = \sum_{\text{i=1}}^{\text{N}} \text{w}_{\text{i}} * \textit{Recall}_i$$

where N is the number of entities in the document, and w_i is the weight assigned to entity i in the document. It should be noted that the B-CUBED algorithm implicitly overcomes the first shortcoming of the Vilain et. al algorithm by calculating the precision and recall numbers for each entity in the document (irrespective of whether an entity is part of a coreference chain).

Different weighting schemes produce different versions of the algorithm. The choice of the weighting scheme is determined by the task for which the algorithm is going to be used.

When coreference (or cross-document coreference) is used for an information extraction task, where information about every entity in an equivalence class is important, the weighting scheme assigns equal weights for every entity i. For example, the weight assigned to each entity in Figure 1 is 1/12. As shown in Figure 6, the precision scores for responses in Figures 2 and 4 are 16/21 (76%) and 7/12 (58%) respectively, using equal weights for all entities. Recall for both responses is 100%. It should be noted that the algorithm penalizes the precision numbers more for the error made in Figure 4 than the one made in Figure 2. As evident from the two examples, this version of the B-CUBED algorithm (using equal weights for each entity) is a precision oriented algorithm i.e. it is sensitive to precision errors.

But, for an information retrieval (IR) task, or a web search task, where an user is presented with classes of documents that pertain to the same entity, the weighting scheme assigns equal weights to each equivalence class. The weight for each entity within an equivalence class is computed by dividing the weight of the equivalence class by the number of entities in that class. Recall is calculated by assigning equal weights to each equivalence class in the truth while precision is calculated by assigning equal weights to each equivalence class in the response. For example, in Figure 2, the weighting scheme assigns a weight of 1/10 to each entity in the first equivalence class, and a weight of 1/14 to each entity in the second equivalence class, when calculating precision. Using this weighting scheme, the precision scores for responses in Figures 2 and 4 are 39/49 (79.6%) and 3/4 (75%) respectively. Recall for both responses is 100%.

Output	MUC Algorithm	B-CUBED Algorithm (equal weights for every entity)
Example 1	P: $\frac{9}{10}$ (90%)	P: $\frac{1}{12} * [\frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{2}{7} + \frac{2}{7} + \frac{5}{7} + \frac{5}{7} + \frac{5}{7} + \frac{5}{7} + \frac{5}{7}] = \frac{16}{21}$ (76%)
	R: $\frac{9}{9}$ (100%)	R: $\frac{1}{12} * [\frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{2}{2} + \frac{2}{2} + \frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{5}{5}] = 100\%$
Example 2	P: $\frac{9}{10}$ (90%)	P: $\frac{1}{12} * [\frac{5}{10} + \frac{5}{10} + \frac{5}{10} + \frac{5}{10} + \frac{5}{10} + \frac{2}{2} + \frac{2}{2} + \frac{5}{10} + \frac{5}{10} + \frac{5}{10} + \frac{5}{10} + \frac{5}{10}] = \frac{7}{12}$ (58%)
	R: $\frac{9}{9}$ (100%)	R: $\frac{1}{12} * [\frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{2}{2} + \frac{2}{2} + \frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{5}{5}] = 100\%$

Figure 6: Scores of Both Algorithms on the Examples

Comparing these numbers to the ones obtained by using the version of the algorithm which assigns equal weights to each entity, one can see that the current version is much less sensitive to precision errors. Although the current version of the algorithm does penalize the precision numbers for the error in Figure 4 more than the error made in Figure 2, it is less severe than the earlier version.

The Model Theoretic Approach To The B-CUBED Algorithm

Let S be an equivalence set generated by the key, and let $R_1 \dots R_m$ be equivalence classes generated by the response. Then we define the following functions over S:

- p(S) is a partition of S with respect to the response, i.e. p(S) is a set of subsets of S formed by intersecting S with those response sets R_i that overlap S. Let $\mathrm{p(S)} = \{\mathrm{P_1, P_2, \dots, P_m}\}$ where each $\mathrm{P_j}$ is a subset of S.
- $\mathrm{m_j(S)}$ is the number of elements that are missing from each $\mathrm{P_j}$ relative to the key set S. Therefore,

$$\mathrm{m_j(S)} = (|\mathrm{S}| - |\mathrm{P_j}|)\ .$$

Since the B-CUBED algorithm looks at the presence/absence of entities relative to each of the other entities, the number of missing entities in an entire equivalence set is calculated by adding the number of missing entities with respect to each entity in that equivalence set. Therefore, the number of missing entities for the entire set S is

$$\sum_{j=1}^{m} \sum_{\text{for each e} \in \mathrm{P_j}} \mathrm{m_j(S)}\ .$$

The recall *error* is simply the number of missing entities divided by the number of entities in the equivalence set, i.e.,

$$\frac{\mathrm{m_j(S)}}{|\mathrm{S}|}\ .$$

Since the algorithm looks at each entity in an equivalence set, the recall *error* for that entire set is

$$\frac{1}{|\mathrm{S}|} \sum_{j=1}^{m} \sum_{\text{for each e} \in \mathrm{P_j}} \frac{\mathrm{m_j(S)}}{|\mathrm{S}|}\ .$$

Recall in turn is

$$1 - \frac{1}{|\mathrm{S}|} \sum_{j=1}^{m} \sum_{\text{for each e} \in \mathrm{P_j}} \frac{\mathrm{m_j(S)}}{|\mathrm{S}|}\ ,$$

which equals

$$1-\frac{\sum_{j=1}^{m}\ \sum_{\text{for each e} \in \mathrm{P_j}} \mathrm{m_j(S)}}{|\mathrm{S}|^2}\ .$$

The whole expression can now be simplified to

$$1-\frac{\sum_{j=1}^{m}\ \sum_{\text{for each e} \in \mathrm{P_j}} |\mathrm{S}|-|\mathrm{P_j}|}{|\mathrm{S}|^2}\ .$$

Moreover, the measure can be extended from a single key equivalence class to a set $\mathrm{T} = \{\mathrm{S_1}, \mathrm{S_2}, \ldots, \mathrm{S_n}\}$ of equivalence classes. Therefore, the recall $\mathrm{R_i}$ for an equivalence class $\mathrm{S_i}$ equals

$$\mathrm{R_i} = 1-\frac{\sum_{\mathrm{j=1}}^{\mathrm{m}}\ \sum_{\text{for each e} \in \mathrm{P_{ij}}} |\mathrm{S_i}|-|\mathrm{P_{ij}}|}{|\mathrm{S_i}|^2}\ ,$$

where $\mathrm{P_{ij}}$ is the jth element of the partition $\mathrm{p(S_i)}$, and, hence, is a subset of $\mathrm{S_i}$.

The recall numbers calculated for each class can now be combined in various ways to produce the final recall. Different versions of the algorithm are obtained by using different combination strategies. If equal weights are assigned to each class, the version of the algorithm produced is exactly the same as the version of the informal algorithm which assigns equal weights to each class, as described in the previous section. In other words, the final recall is an average of the recall numbers for each equivalence class, i.e.,

$$\mathrm{R_T} = \frac{1}{\mathrm{n}} \sum_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{R_i}\ .$$

To obtain the version of the informal algorithm which assigns equal weights to each entity, the final recall is computed by calculating the weighted average of the recall numbers for each equivalence class where the weights are decided by the number of entities in each class, i.e.,

$$\mathrm{R_T} = \sum_{\mathrm{i=1}}^{\mathrm{n}} \frac{|\mathrm{S_i}|}{\sum_{\mathrm{j=1}}^{\mathrm{n}} |\mathrm{S_j}|}\ \mathrm{R_i}\ .$$

Finally, as in the case of the Vilain et. al algorithm, the precision numbers are calculated by reversing the roles of the key and the response in the above formulation.

Task Relative Strengths and Weaknesses of the Two Algorithms

The Vilain et. al algorithm is useful for applications/tasks that use single coreference relations at a time rather than resulting equivalence classes. For our development in the coreference task, the two algorithms provide distinct perspectives on system performance. Vilain et. al provide a strong diagnostic for errors that reflect pairwise decisions done by the system. Our visual display techniques emphasize just this sort of processing.

Our total score under the Vilain algorithm, with a somewhat fuzzier extent requirement and stricter requirement for links is 81% precision and 45% recall.

The same files using the B3 algorithm resulted in 78% precision and 31% recall. The precision numbers are comparable which indicates that our goal of high precision is supported under both views of the data. The 14% drop in recall was however unexpected. The reason is fairly straight forward– our system is not doing a good job of relating large equivalence classes. This is the converse of penalizing the system for positing incorrect links that result in larger equivalent classes than smaller ones.

The drop in recall in the B3 scorer also suggests a distinct class of coreference resolution procedure that we could investigate–growing of large equivalence classes via an entity merging model which eschewed the standard left-to-right processing strategy of most coreference resolution systems. If such a procedure can reliably grow medium sized equivalence classes into large ones, then the recall figures will improve under the B3 scorer. The Vilain et. al scorer notes no difference between correctly relating two singleton equivalence classes and correctly relating two large equivalence classes.

Since large equivalence classes tend to include topically significant entities for documents, correctly identifying them is perhaps crucial to applications like summarization and information extraction.

Developing with the Vilain et al algorithm

The below analysis reflects how we assesed the individual contributions of the components during development. Since the B3 algorithm was not yet implemented, we did not use it for development.

Our explicit goal was to maximize recall at a precision level of 80%. We feel that this level of precision provides enough accuracy to drive a range of coreference dependent applications–most important for us was query sensitive text summarization. Our overall approach was to break down coreference resolution into concrete subprograms that resolved a limited class of coreference well. Each component could be scored separately by either running it in isolation, or by blocking coreference from subsequent processes. Below we discuss each component in the order of execution.

Genre Specific Coreference

A problematic aspect of any new genre of data is the existence of idiosyncratic classes of coreference and the MUC-7 data was particularly troubling since very oddly formatted text was fair game for coreference. For example, the strings 'HUGHES' and 'FCC' in '`<SLUG fv=tia-z> BC-HUGHES-FCC-BLOOM </SLUG>`' are coreferent with the same strings in '`<PREAMBLE>BC-HUGHES-FCC-BLOOM...`' which was outside the scope of our linguistic tools. Simple programs were written to recognize this sort of coreference. The performance by the Vilain scorer is 4.2% recall 67.5% precision.

This performance is well below what we observed in training data–the precision was 85-90% for similarly sized collections. Perhaps part of the problem was that we never quite grasped why some but not all these all CAPS strings were not coreferent.

La Hack 3

La Hack is a carry over from our original MUC-6 system, and it is responsible for identification of proper noun coreference. This component is indirectly helped by IBM's named entity tool 'Textract' which find extents of named entities in addition to assigning them properties like 'is_person', 'is_company'. It is the foundation upon which our coreference annotation is built–mistakes here can be devastating for the rest of the system. In MUC-6, La Hack performed at 29% Recall and 86% precision, but it faired somewhat worse in MUC-7 with, 24.0% precision 80.0% recall.

We observed that the New York Times data had far less regular honorific use and corporate designator use than the MUC-6 corpus based on Wall Street Journal. As a result, there were fewer reliable indicators of proper names.

Highly Syntactic Coreference

This component asserts coreference between phrases that are in appositive relations or that are in predicate nominal relations. We were quite surprised at how poorly this component performed since we expected performance to be above the 80% precision cutoff. Our actual performance is 3.3% precision 64.0% recall.

Quoted Speech

Quoted speech has idiosyncratic patterns of use that are better solved out side the scope of our standard coreference resolution module. We expected performance to be above 90% precision and were pleased with 2.6% recall and 86.8% precision. This module is a good example of how the coreference problem can be fruitfully broken up into sub-parts of individually high precision.

CogNIAC Proper Noun Resolution

CogNIAC is the most general purpose coreference resolution component of the system. It features a fairly sophisticated salience model and property confidence model to preorder order the set of candidate antecedents. The importance of the preorder is that it allows ties between equally salient antecedents–and in the case of ties the anaphor is not resolved.

When deficiencies were noted with the output of LaHack, the simplest solution was to add a proper noun resolution component to CogNIAC. In the end this addition added a bit of recall but with fairly low precision with 1.2% recall and 65.2% precision.

CogNIAC Common Noun Resolution

Common noun coreference is an important part of coreference, but it is very difficult to accurately resolve. Our MUC-6 system had fairly poor performance with 10% recall and a precision of 48%. We were surprised with an increase in performance over training data (78% precision) with 7.1% recall 90.7% precision.

Common noun anaphora is probably one of the most trying classes of coreference to annotate as a human. This is due to many difficult judgment calls required on the part of the human judges, and this was reflected in the consistency of annotation in the training data. We found it challenging to develop on the training data because the system would find what we considered to be reasonable instances of coreference that the annotator had not made coreferent. We believe that common noun anaphora is a large source of inter-annotator disagreement.

CogNIAC Pronouns

The pronominal system performed under our goal of 80% precision. In training, we found that we were constantly balancing the ability of pronouns to i) refer uniquely, and ii) have all entities have the correct property. We adopted a property confidence model that encouraged recall over precision. This meant that a proper noun like 'Mrs. Fields' would be both potentially an antecedent to feminine pronouns, and pronouns that referred to companies. A salience model was then applied to these overloaded entities and pronominal resolution served to be a word-sense disambiguation problem in addition to a coreference resolution problem. Our performance was 4.5% recall and 70.0% precision.

Conclusions

One of the stronger conclusions that we have come to regarding coreference is that there is an apparent linear trade-off between precision and recall given the performance of other systems with the coreference task. Our suspicion is that the same can be said with the B3 scorer but that will have to await experimentation. This is a positive result in its self because we now can choose from multiple types of coreference systems depending on our task. We consider high precision systems to be more useful for the types of systems that we build, but, it has not been clear that high precision systems were possible.

We also believe that the space of high precision 'contributors' to coreference is not exhausted. We doubt that there are any 10% recall/80% precision subcomponents that we have not already explored, but there are certainly 1-5% recall opportunities. How well they will sum to the recall of the entire system is unknown, but there is room for improvement.

REFERENCES

[Bagga, 98a] Bagga, Amit. How Much Processing Is Required for Cross-Document Coreference?, *this volume.*

[Vilain, 95] Vilain, Marc, et al. A Model-Theoretic Coreference Scoring Scheme, *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pp. 45-52, November 1995.

[MUC-6, 95] *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, November 1995, San Mateo: Morgan Kaufmann.