A Scalable, Empirical Approach to Anaphoric Reference

Caroline Tice

August 30, 1994

Contents

Abstract

Most of the suggested solutions for the anaphora problem have been concerned with getting the correct answer in all situations. This means they must involve large amounts of syntactic, semantic and world knowledge, and therefore can only be applied to a limited scope of documents. These solutions do not scale well. We have designed and implemented a scalable heuristic approach to the anaphora problem. Our main concern was not to get the correct answer in all situations, but to create an easily scalable solution that will find the correct answer most of the time. We designed our heuristics in two stages. First we created the simplest possible solution we could. We then used this simple solution as a baseline against which to measure our more advanced heuristics. Our heuristic solutions work for both definite noun phrase anaphora and pronoun anaphora. We tested our implementations on two moderate-sized pieces of text, containing a total of 670 definite noun phrases and 95 pronouns. Our baseline program chieved 50.9% accuracy for the definite noun phrases, and 30.5% accuracy for the pronouns. Our more advanced heuristics showed a dramatic improvement over the baseline, with 71.0% accuracy for the definite noun phrases, and 73.7% accuracy for the pronouns.

1 Introduction

The anaphora problem remains a difficult one, both to define and to solve. The problem itself stems from the fact that natural languages are very rich and therefore there are many alternative ways to specify any given object. The person attempting to understand natural language text is then obliged to sort out which objects are being referenced and which words are referencing them. The anaphora problem is determining when two or more phrases, often containing different words, are referring to the same object. Such an occurrence is called *co-reference*.

To date, work on the anaphora problem has been concerned with getting the correct answer in all situations ([6],[7],[13],[15]). Not surprisingly this requires a large amount of knowledge of all types (syntactic, semantic and world). In fact it has been shown that the problem requires an arbitrarily large amount of knowledge [1] [2]. Consequently the solutions implemented so far have only been for very limited domains, and do not scale well.

Recently there has been a lot of interest in using statistical analysis and heuristics on large corpora of documents in an attempt to find scalable solutions to certain traditional natural language problems such as statistical parsing, disambiguation and topic selection ([3], [5], [9], [17]). We have taken a similar approach to the anaphora problem.

In designing our heuristics, our main goal was to create a solution that scales easily to large and diverse corpora. As a result, our heuristic uses only a minimal amount of syntactic and semantic knowledge. Obviously such a solution will not always be correct. However we are not concerned with perfection so much as practicality. We want to get the correct answer most of the time, which we do. These heuristics could be useful in many ways, including increasing the accuracy of statistics used for automatic topic assignment, information retrieval, etc.

As previously mentioned, both above and elsewhere ([13], [15]), words do not themselves refer to other words. Rather words refer to objects in some shared conceptual world. However, for the sake of brevity in this paper, when two phrases refer to the same notional object, we will say that the phrase that occurs later in the text *refers* to the earlier phrase, and that the first phrase is the *referent* or *antecedent* of the second.

The two most common ways for two phrases to co-refer are with definite noun phrases and with pronouns. For the purposes of this paper, the term *noun phrase* will mean a simple noun phrase, i.e., a noun phrase consisting of an optional determiner, optional modifiers, and one or more nouns. The term *definite noun phrase* will mean a noun phrase in which the determiner is one of *the, this, that, these* or *those.*

Our heuristics are modelled to some extent upon Hobbs' simple heuristic for pronoun reference [10]. The basic idea is, given a definite noun phrase or a pronoun, to search backwards through the text examining each noun phrase in turn until one finds a noun phrase with a similar meaning[1]. This noun phrase is then proposed as the antecedent for the initial anaphora.

We designed and implemented these heuristics in two stages (a baseline version and a smarter version), and have run it on two texts ([8], [14]). These texts consist of 349 sentences (approximately 8000 words) containing a total of 670 definite noun phrases and 95 pronouns. On these texts, our baseline

[1] Here "similar meaning" is determined by comparing the head nouns of the two noun phrases using WORDNET. See section 2 for a more complete description.

heuristic determines the coreference correctly for 50.9% of the definite noun phrases and for 30.5% of the pronouns. Our second, more intelligent heuristic determines the coreference correctly for 71.0% of the definite noun phrases and for 73.7% of the pronouns.

The rest of this paper is organized as follows. In section 2 we present the design, implementation and results of our definite noun phrase heuristics. In section 3 we present the design, implementation and results of our pronoun heuristics, and section 4 presents our conclusions and a discussion of future work.

2 Definite Noun Phrase Anaphora

As was mentioned above, the basic idea of the heuristic we propose is to search backwards through the text until a plausible noun phrase is found and to propose that as the antecedent. We tackle this problem in two stages (following Hobbs [10]). In the first stage we implemented the most simple-minded version of this heuristic that we could, and took a look at how well it performed. We then used this as a baseline against which we compared our later, more intelligent implementation.

2.1 Baseline Implementation

The baseline (and the advanced) implementation takes as input a text file that has been tagged for parts of speech, and in which the noun phrases have been bracketed. To do this simple tagging and bracketing, we use the PARTS tagger [3]. Our program then examines each tagged noun phrase in turn. All noun phrases within the given search window are stored in a list. Whenever a definite noun phrase is encountered, the list of noun phrases is searched in order (most recent to least recent), looking for a noun phrase that is close in meaning to the definite noun phrase. The first such noun phrase found (if any) is proposed as the antecedent for the definite noun phrase. If no noun phrase is found within the window, it is assumed that the definite noun phrase does not have an explicit antecedent in the text.

Two important parts of the implementation which we have mentioned but not described are the search window, and determining if two noun phrases are "close in meaning". One might assume that allowing the heuristic to search

Window Size	Errors in Text 1	Errors in Text 2	Total Errors
1 Sentence	138	101	239
2 Sentences	130	84	214
3 Sentences	127	84	211
4 Sentences	127	80	207
5 Sentences	123	75	198
6 Sentences	128	74	202
7 Sentences	127	74	201
8 Sentences	127	71	198
9 Sentences	123*	71*	194*
10 Sentences	127	73	200
15 Sentences	124	72	196
20 Sentences	124	76	200
30 Sentences	125	77	202
40 Sentences	128	80	208

Table 1: Results using various window sizes. '*' indicates the actual size used in our experiments.

for antecedents for an unbounded distance would improve performance. However allowing the heuristic to search indefinitely far back leads to "false positive" errors (proposed antecedents for definite noun phrases that do not in fact co-refer with anything in the text).

We selected our initial window size by running hand simulations of the baseline algorithm on the two texts, using various window sizes. The results of these hand simulations suggested that the optimal window size was 8. This is the size we then used in our baseline implementation. After implementing our advanced algorithm (described later), we ran it on a variety of window sizes to verify the optimal size. We concluded that the best size for a search window was 9 sentences (not including the sentence in which the definite noun phrase itself occurs). See table 1 for details of this analysis. If the window is smaller than 9, relatively more valid antecedents are missed, and if it is larger than 9, relatively more false antecedents are proposed. We use a window of 9 with our advanced algorithm.

To determine the closeness in meaning of two noun phrases, we used the

WORDNET synset hierarchy [11].[2] For each noun phrase, all of the WORDNET synsets for the head noun[3] of the phrase are found. These two sets of synsets are then compared. If any synset for one of the head nouns is within a distance of two (within the WORDNET hierarchy)[4] from any synset for the other head noun, the two head nouns are considered to be "close enough" in meaning.

2.2 Baseline Results

We implemented our program in C on a Dec 3000 alpha workstation. We ran it on two moderate-sized pieces of text, "Zebra: A Striped Network File System" [8] and two chapters out of "Early Civilization in China" [14]. Each run takes approximately ten minutes, depending on the size of the text.

The two texts together consist of 346 sentences, containing a total of 670 definite noun phrases, and many more indefinite noun phrases. Of the 670 definite noun phrases, 325 actually co-refer with other phrases in the text, while 345 do not. In looking at the performance of our heuristic, there are two broad categories of errors that it could make. One pertains to definite noun phrases that have antecedents, and the other pertains to definite noun phrases that do *not* have antecedents. Therefore when checking the accuracy of the algorithm, a definite noun phrase is considered to have been treated correctly either a) when it has an antecedent and the correct antecedent has been proposed, or b) when it does not have an antecedent in the text, and none have been proposed.

Under these circumstances a program that does absolutely nothing will get 345 noun phrases correct (they have no antecedent) and 325 wrong. In other words it would be 51.5% accurate. Considered in this light, the performance of the baseline algorithm (see table 2) looks pretty poor. On the other hand, if we compare it against a program that randomly selects any

---

[2] WORDNET is an IS-A hierarchy of word meanings. Each node in the hierarchy is a *synset*. A synset (short for *syn*onym *set*) is a list of words which all share a meaning, namely the meaning of the synset. Because a synset is a meaning, any given word may be in multiple synsets, one for each meaning of the word.

[3] We make the simplifying assumption that the last noun in the noun phrase is the head noun.

[4] I.e., the two synsets are the same; or, they are siblings; or, one is the parent of the other; or, one is the grandparent of the other.

	Text 1	Text 2	Total
Total no. of DNP's	372	298	670
DNP's treated correctly	147 (39.5%)	194 (65.1%)	341 (50.9%)
Errors made	225 (60.5%)	104 (34.9%)	329 (49.1%)

Table 2: Performance of the baseline algorithm for definite noun phrase anaphora. DNP = Definite Noun Phrase.

	Text 1	Text 2	Total
Total Errors:	225	104	329
False Positives:	122	63	185
False Negatives:	50	17	67
Incorrect Antecedent Selected:	53	24	77

Table 3: Breakdown of the errors for the baseline algorithm.

option, either selecting one of the possible antecedents in the search window or selecting no antecedent, then the baseline looks much better. If we state that on average a sentence contains four noun phrases (a rather gross underestimate for the particular texts we used), and there are nine sentences in the search window, then there are 36 noun phrases from which to choose, not considering any that precede the definite noun phrase in the current sentence. Adding to this the option of not selecting a antecedent, then for any definite noun phrase the program has 37 options from which to choose. Choosing randomly from the 37 choices, the program has roughly a 3% chance of making the correct choice. Compared with this possibility, the baseline program performs very well.

As can be seen in table 2, the performance of our heuristic varied greatly on the two texts. This was due to some extent to an extreme difference in writing styles, and also to the large occurrence of proper noun phrases in the first text (the baseline has no special rules for matching proper noun phrases). The main types of errors that the algorithm made can be seen in table 3. The errors involving noun phrases that have antecedents are broken into two categories: those where the algorithm found the wrong antecedent and those where it failed to find any antecedent.

The three major categories of error that can occur, as shown in the table, are false positives, false negatives, and incorrect antecedents. A false positive is when the program suggests an antecedent for a definite noun phrase that does not actually have one. A false negative is when the program fails to suggest any antecedent for a definite noun phrase that *does* have one. An incorrect antecedent, as the name suggests, is when the program finds an incorrect antecedent, but there is a correct one. As can be seen from the table, by far the most common type of error is the false positive.

Table 4 shows a more detailed analysis of the errors and their various causes. The causes of error are broken into three main subcategories: those errors due to various design and implementation decisions, and which therefore we did not attempt to correct; those errors which the advanced version of our heuristic attempts to correct; and other miscellaneous errors. Errors marked as "unfortunate circumstances" mean that, although the definite noun phrase in question does not truly co-refer with anything in the text, there happens to be a noun phrase in the search window that, by pure coincidence, really *means* the same thing as the definite noun phrase (but refers to something else), and so we get a false positive. An example of this is *the remaining servers* being proposed as the antecedent for *the servers*, when in fact the two phrases refer to two different sets of servers. The category "requires understanding text" covers a broad spectrum, from simple things, such as understanding that *the eastern area* co-refers with *the east*, all the way up to understanding that "the simple grave-goods" is actually co-referring with "pots and stone tools" in the sentence "The burials ...with the body laid on the back and accompanied by pots and stone tools." The category "unfortunate circumstances" mentioned above could also be considered a subcategory of "requires understanding", since those errors do require text understanding in order to correct. However these two categories are distinct in that "unfortunate circumstances" will only cause false positive errors, while "requires understanding" will only cause false negative and incorrect antecedent errors. "Partitives" and "Dunce Phrases"[5] are errors caused by the program finding plausible antecedents for partitive and dunce phrases when, as explained in the next section, such phrases do not in fact have any antecedents. "Ambiguous" errors occur when two head nouns happen to have secondary meanings (not the ones being used in the text) which match in WORDNET, so the heuristic erroneously proposes them as co-referents. "Proper noun phrase" errors are errors that arise when trying to find antecedents for definite noun phrases

[5]See section 2.4 for an explanation of dunce phrases.

Cause	of the Error	# of Errors	Type of Errors
Basic	Design or Implementation Decisions		
	Too far apart in WORDNET hierarchy	14	false negative
	Farther than 9 sentences away	30	false negative
	WORDNET is missing the word	3	false negative
Addressed	in Advanced Algorithm		
	Found incorrect match before correct one	36	incorrect
	Partitives	81	false positive
	Proper noun phrases	28	all
	Dunce phrases	10	false positive
	Ambiguous	30	false positive
Other			
	Requires understanding text	40	incorrect, false negative
	Unfortunate circumstances	28	false positive
	Miscellaneous Other	29	all

Table 4: Various causes of errors in the baseline algorithm.

that contain proper nouns. These errors are due to certain idiosyncracies of proper noun phrases, and are explained more fully in section 2.4.

2.3 Non-Referring Definite Noun Phrases

It turns out that a significant number of definite noun phrases (about 50%) do not have any antecedent in the text. There are several different categories of definite noun phrases that do not co-refer with previous phrases in the text. Table 5 shows some of these categories, and how the non-co-referring definite noun phrases in our texts divide among them. (There may be other categories as well, but these are all the ones we found.) A brief description of each category is given below.

Object is understood from the context means that the cognitive object which the definite noun phrase specifies has not been mentioned explicitly in the preceding text. However the object is notionally present from the context. If one is talking about a baseball game, one can mention *the pitcher*; if one is speaking about a restaurant, one can mention *the waiter*. These objects

Categories of Phrases	# of Occurrences
Object is understood from the context (frame)	67 (20.6%)
Dunce phrases	67 (20.6%)
Partitive phrases	52 (16.0%)
Features/characteristics of objects	41 (12.6%)
Phrases modified by a relative clause	29 (8.9%)
Co-refers with objectified sentence	20 (6.1%)
Superlative/compartive phrases	18 (5.5%)
Object is unique to general default frames	17 (5.2%)
Forward (immediate) reference	8 (2.4%)
Generic objects	3 (0.9%)
Meta-references	3 (0.9%)

Table 5: The various types of definite noun phrases that do not co-refer with any previous phrase in the text, and their distribution in our texts.

are part of a *frame* [4], [12] that has already been introduced. An example from our texts is:

"The burials are in...pits, with *the body* laid on *the back*, and..."

Here neither a body nor a back were mentioned previously, but the reference is immediately clear from the context.

Dunce phrases are definite noun phrases with very little content, and which do not have antecedents. They are explained in detail in section 2.4.

Partitive phrases are phrases that refer to parts or possessions of another object. In some ways they are very similar to the frame objects mentioned above. It is normal and expected that the main object will have such a part or possession, so mention of the main object brings in enough context that the partitive can be specified with a definite noun phrase. Examples of partitives include "...below *the present top* of the limestone tump" and "...can be reconstructed using *the contents* of the remaining servers...".

Features/characteristics of objects are very similar to partitive phrases. The difference is that rather than being a possession or subpart of the main object, features and characteristics are integral to the main object. E.g. "*The geological age* of the cave..." or "*The advantage* of this approach...".

Phrases modified by a relative clause are allowed to be definite noun phrases because the relative clause can make it clear which object is being specified, even if the object has not been previously mentioned. It is worth mentioning, however, that a definite noun phrase can be modified by a relative clause and still have an antecedent in the preceding text. Some examples we found which did not have antecedents in the text include: "*The portion*...that is written to each server..." and "*...the clay and stony rubble* which...filled up their lairs."

Co-refer with objectified sentence means that the definite noun phrase does have an antecedent of sorts, but the antecedent is not any particular preceding phrase. Rather it is a concept distilled from one or more preceding phrases. The classic example of this is "A boy spilled paint all over a goat. When his mother came out she said, 'Did you do *that*?"'[16]. Examples from our texts are "The picture which emerges from *these findings*...", "...it has been suggested that...but *this theory*...", and "Both of *these considerations* make..."

Superlative and comparative phrases are fairly self-explanatory. Superlative phrases can be definite without having an antecedent, because by their very nature they tend to be unique. E.g. "*The earliest record* of human activity,,," or "*...the most famous* site in the country...".

Object is unique to general default frames means that it is some unique object in most people's standard every-day frame of reference. Examples of this might include *the sun* or *the moon*. In our texts we found "...which record the physical development of *the human race*...". This category is obviously similar to the first one. What makes it different is that the *frame* does not need to be explicitly introduced before the references are made. The frame is always there, so to speak.

Forward reference definite noun phrases actually do have co-referents in the text. However the co-referents occur *after* the definite noun phrase (often immediately after), so our heuristic cannot find it. An example of this is "*...the village* of Chou K'ou Tien...". It might be possible to design heuristics for dealing with these cases (e.g. one might include all of the current sentence in the search, rather than just that portion that precedes the definite noun phrase). We did not do this, as there were only 8 such definite noun phrases (1

Generic objects are when a definite noun phrase is used not to refer to any particular object but to a generic ideal representing a whole class of objects: "In the northwest the bones of cattle and goats have been found, but *the horse* was apparently not yet known either here or in the Central Plain."

Meta-references are when a definite noun phrase is used to refer, not to any object mentioned in the text, but to the text itself. Examples include "*This paper* presents..." and "...the rest of *this paper*...". Such phrases often do not have antecedents in the text. Even when there are multiple meta-references in the text, they are often so far apart that they fall well outside the search window.

2.4 "Smart" Implementation

As table 4 shows there are five main causes of error that we decided to address in our second implementation: partitives; dunces; plausible but incorrect antecedents being found before the correct antecedents; proper noun phrases; and, ambiguity. In attempting to resolve each of these types of error, we always kept our goals of scalability, minimal knowledge, and *mostly* correct in mind. As a result of this, our solutions are frequently not perfect, and sometimes they even introduce a few errors. However in all cases, the number of errors the solutions might introduce is significantly less than the number of errors they eliminate.

As previously mentioned, by far the largest number of errors we got were false positives. Upon close inspection, we found that many of the definite noun phrases for which these false antecedents were proposed (49.2%) fell into two broad categories: partitives[6] (e.g. "*the wheel* of a car") and *dunce phrases*. By a *dunce phrase* we mean a noun phrase which, while not quite empty of meaning, does not convey very much. Some examples of dunce phrases are "the amount of" "the appearance", "the idea that", etc. Because we are attempting to work in a knowledge-poor environment, dunces and partitives are not necessarily easy to recognize. Our approach to solving the dunce phrase problem is to make a list of dunce head nouns. The list contains 22 words.[7] Whenever the head of any definite noun phrase is a member of this list, the phrase is considered a dunce, and no antecedent is searched for.

Careful examination of the partitive phrases revealed that in nearly every case (91.7%), the partitive definite noun phrase was immediately followed by

[6]Because they are so similar, we put features and characteristics into this category as well.

[7]*amount, appearance, assumption, conclusion, contents, direction, fraction, idea, majority, manner, number, order, performance, place, portion, practice, process, rate, time, type* and *use*

a prepositional phrase beginning with the word "of". We further found that very few definite noun phrases that were not partitives had this particular structure (33 out of 670). The remaining partitive phrases were of the form *the noun1's noun2*. Therefore, we use these two patterns to attempt to recognize partitive definite noun phrases. When any definite noun phrase is identified as being a partitive, no antecedent is searched for.

The next cause of error we addressed is that sometimes the correct antecedent for the definite noun phrase would not be the *first* plausible match found, but would be slightly farther back in the search window. In order to fix errors arising from the correct antecedent being farther back in the search window than a plausible but incorrect antecedent, we decided to alter our search strategy slightly, from first match to best match. In our second implementation, whenever a definite noun phrase is encountered, *all* of the noun phrases in the search window are considered. Any noun phrase in the window that is close enough in the WORDNET hierarchy to be considered a candidate match is given a score. The noun phrase with the best score is selected as the correct antecedent. A number of factors are taken into consideration when scoring the various noun phrases, including how many words in the candidate phrase are identical to words in the definite noun phrase, how many words are different or missing, how far apart the different words are in the WORDNET hierarchy, and how far back the noun phrase is in the search window. See the appendix for a complete explanation of the scoring.

The fourth source of errors came from trying to find antecedents for definite noun phrases that contain proper nouns. Before going any farther we should state that by *proper noun phrase* we mean any simple noun phrase (as previously defined) that contains a proper noun anywhere in it. The head word does not necessarily have to be a proper noun. Matching proper noun phrases presented a few unique problems. For example it is possible for two proper noun phrases to have an identical head word, but refer to two completely different entities (e.g. *the Amazon river* versus *the Nile river*). Also, it is frequently the case that some words in a proper noun phrase may be omitted later, but the two phrases still refer to the same thing. Some examples of this are "the Nile river" later being referred to as "the Nile", and "the Shang Bronze Age" later being referred to as "the Bronze Age". These special properties of proper noun phrases are accounted for in our heuristic by the following rules:

- A proper noun phrase can only co-refer with another proper noun phrase.

- All of the proper nouns in one phrase must be in the other. E.g. *the Shang Bronze Age* will match *the Bronze Age*, but neither will match *the Iron Age*.

- If at least one head word is a proper noun, then the head words do *not* need to match in WORDNET (otherwise they *do*). This rule is to catch the case where one head word is a proper noun and the other is not, as in *the Nile river* matching *the Nile* (*Nile* and *river* may very well be farther apart than two in WORDNET). If *both* head words are proper nouns, then the matching has already been taken care of by the previous rule; if *neither* head word is a proper noun, then normal WORDNET matching is done.

The final source of error we addressed resulted from a combination of ambiguity of the head nouns and an infelicitous matching strategy . It often occurs that each head noun has a second meaning (not the one being used in the text), and the second meanings happen to match. To reduce the number of errors that occurred because of ambiguous words in the text, we re-examined our strategy for deciding that two words were "close enough" in meaning. In particular, while it makes sense for words that are in the same synset or in a parent or grandparent relationship to be considered as possible co-referents, this is not true of words that are in sibling synsets. For example, "knife" and "axe" both share the parent "cutting tool", but one would not really want to say that the words refer to the same thing.[8] Eliminating the sibling matches reduces our ambiguity errors by 40%. The remaining 60% of the ambiguity errors might be eliminated by running a disambiguation algorithm such as [5] or [17] on the text before doing the reference resolution. However since resolving the remaining ambiguity errors would only gain us at most a 3% improvement in accuracy, we decided not to do so at this time.

[8]There were certain cases in WORDNET where we really *did* want elements in sibling synsets to co-refer, but we decided that this was an idiosyncracy of WORDNET rather than something that necessarily should be the case. For example "places" and "sites" are in sibling synsets, but we wanted them to match. The same is true of "jars" and "pots".

2.5 "Smart" Implementation Results

Table 6 shows the performance of the advanced heuristic. The baseline totals are also shown, for comparison. Tables 7 and 9 show an analysis of the errors made by the advanced heuristic, analogous to tables 3 and 4. Once again the errors in table 9 have been broken into subcategories: those which are inherent to our design and implementation choices; those that have been introduced by our solutions; and, those of the remaining errors (labeled "miscellaneous other" and "ambiguous") that it might be possible for us to correct, if we were to run disambiguation and to write various small hacks[9]. The errors due to partitives and proper noun phrases shown in table 9 are those original partitive and proper noun phrase errors that our heuristics do not correct. The partitives that our heuristic fails to recognize are: "...and had a brain about two-thirds *the size* usual in modern man" and "The houses...appear to have been built on a round plan...with a round depression at *the center* which probably marks...". There does not seem to be any easy way to recognize these partitives without introducing knowledge and understanding to our system. The proper noun phrase errors are a result of our third proper noun matching rule (the head nouns of two proper noun phrases do not have to match in WORDNET). However this rule still corrects more errors than it causes, so we intend to keep it. Obviously it would not be good if our "solutions" introduced more errors than they solved. Therefore table 8 shows a comparison of the errors solved versus those introduced by our various solutions. The one anomalous solution is the one for partitives of the form *the noun1's noun2*. In the texts we use, there are no definite noun phrases of this form for which false antecedents are proposed, while unfortunately there are two definite noun phrases of this form that *do* have antecedents. However we suspect that this is a coincidental circumstance of these texts, and that if we look at more texts we will find that this heuristic pays off.

[9]Such hacks would include special code for handling conjunctions, and for phrases such as *the former, the latter, these last, the rest* and *the one.* Doing all of this (and the remaining disambiguation) *perfectly* would gain us at most a 5.8% increase accuracy, however.

	Text 1	Text 2	Total	Baseline
Total no. of DNP's	372	298	670	670
DNP's treated correctly	249 (66.9%)	227 (76.2%)	476 (71.0%)	341 (50.9%)
Errors	123 (33.1%)	71 (23.8%)	194 (29.0%)	329 (49.1%)

Table 6: Performance of the advanced algorithm for definite noun phrase anaphora.

	Text 1	Text 2	Total
Total Errors:	123	71	194
False Positives:	37	26	63
False Negatives:	63	28	91
Incorrect Antecedent Selected:	23	17	40

Table 7: Breakdown of the errors for the advanced algorithm.

Solution	Errors Solved	Errors Caused	Net Gain
partitives marked by "of"	78	33	+45
partitives marked by "'s"	0	2	-2
dunce head nouns	10	0	+10
proper noun rules	21	7	+14
remove sibling matches	12	2	+10
Total	121	42	+77

Table 8: Analysis of errors corrected versus those caused by the various simple solutions.

Cause	of the Error	# of Errors	Category of Errors
Basic	Design or Implementation Decisions		
	Too far apart in WORDNET hierarchy:	17	false negative
	Farther than 9 sentences away:	12	false negative
	WORDNET is missing the word:	6	false negative
Errors	that Might be Corrected		
	Ambiguous:	18	false positive
	Miscellaneous other:	25	all
Errors	*Caused* by Solutions		
	Caused by Partitive Solution:	35	false negative
	Caused by Proper Noun Phrase Solution:	7	false negative
	Caused by Best Match Solution:	9	incorrect
Other			
	Partitives* :	2	false positive
	Proper Noun Phrases*:	7	false positive, incorrect
	Requires understanding text:	30	false negative, incorrect
	Unfortunate circumstances:	26	false positive

Table 9: Detailed analysis of errors made by advanced algorithm. '*' marks partitives and proper noun phrases for which our heuristics failed.

	Text 1	Text 2	Total
Pronouns in Text	46	49	95
Pronouns treated correctly	8 (17.4%)	21 (42.9%)	29 (30.5%)
Errors	38 (82.6%)	28 (57.1%)	66 (69.5%)

Table 10: Performance of the baseline algorithm for pronoun anaphora.

3 Pronoun Anaphora

3.1 Pronoun Baseline Implementation

Our pronoun baseline implementation was modelled very closely upon our baseline definite noun phrase algorithm, which in turn was very loosely modelled on the ideas presented in [10]. In the pronoun baseline, whenever the program encounters a pronoun, it searches backwards through the text looking for noun phrases (or other pronouns) of the same number, gender and person (first, second or third). The first phrase found that matches in number, gender and person is proposed as the antecedent.

This simple baseline algorithm for resolving pronoun co-reference did surprisingly poorly. While Hobbs claimed that his heuristic got the co-reference correct 88% of the time, our baseline implementation only found the correct antecedent about 30.5% of the time. There are many reasons for this discrepancy, which we discuss in later sections.

3.2 Pronoun Baseline Results

There are 95 pronouns in the two texts, 71 of which co-refer with other items in the text, and 24 of which do not. Again for comparison, a program that did absolutely nothing would get 24 pronouns correct and 71 wrong, thus having an accuracy of 25.3%. Considered in this light, the performance of the baseline algorithm is not dramatic. The performance of the baseline program is shown in table 10.

As can be seen, the performance of the heuristic again varied widely on the two texts. A breakdown of the errors is shown in table 11. As can be seen, most of the errors result from finding the wrong antecedent. The reason for this is given below, along with an explanation of why our results vary so drastically from Hobbs'. There are two other things worth mentioning before

	Text 1	Text 2	Total
Total Errors:	38	28	66
False Positives:	16	6	22
False Negatives:	2	2	4
Incorrect Antecedent Selected:	20	20	40

Table 11: Breakdown of the errors for the pronoun baseline algorithm.

discussing the more advanced pronoun algorithm. Our heuristics for determining the co-reference of pronouns and for determining the co-reference of definite noun phrases are all incorporated into one program. It determines both kinds of co-reference simultaneously. Because of this, we use the same window size for both the pronouns and the definite noun phrases. However all of the pronouns were found within a window of 2 sentences[10], so changing the window size from 8 to 9 between the baseline and the advanced algorithms had no effect on correctly determining the pronoun co-reference. The other thing worth mentioning is that *none* of the pronouns in our texts had antecedents that came after the pronoun in the text. All of the antecedents preceded the pronouns with which they co-referred.

3.3 "Smart" Pronoun Implementation

To begin with, Hobbs assumes that he had a correct parse *tree* for each sentence, while we are searching flat text (with bracketed noun phrases). Of even more importance, while Hobbs' heuristic moves up and back through the tree to find a starting point for the search, but the actual search for an antecedent progresses *left-to-right, top-to-bottom* through the tree. For all intents and purposes, *our* search was progressing right-to-left, bottom-to-top.

While we could not truly emulate Hobbs' search heuristic since we were not working with parse trees, we did modify our search order to more closely resemble his in our smarter implementation. Although we still search backwards through the window of sentences, within an individual sentence we start at the beginning of the sentence and search *forward* for a plausible antecedent. As before, the first plausible phrase found (correct number, gender

[10]For those pronouns which have antecedents in the text (71 total): 65 (91.5%) of the pronoun antecedents were within the same sentence; 4 (5.6%) were one sentence back; and, 2 (2.8%)were two sentences back.

and person) is proposed as the antecedent for the pronoun in question. This simple change in search strategy leads to a dramatic improvement in our accuracy, raising it from 31% to about 50% (see table 15).

That searching left-to-right, as opposed to right-to-left, made such a difference may be explained by the following. The majority of the pronouns (57%) in our texts co-refer with the subject, either of a sentence or of a relative clause. The next largest group of pronouns (27%) do not co-refer with anything. They are null subjects, null objects and extrapositionals. They appear in phrases such as "it was raining" or "it is possible that". If we remove these non-referring pronouns from consideration and only look at pronouns that actually do have antecedents, then 77% of all the pronouns co-refer with subjects. This observation is consistent with the idea that the subject of the sentences is most likely to be the main topic under discussion, and the main topic is the thing likely to be referenced most of the time.

Since such a large number of pronouns in the texts do not co-refer with anything, our next step was to attempt to recognize such pronouns and not search for antecedents for them. To be able to accurately recognize all such pronouns would require a large amount of actual text understanding, which would violate our goal of minimal knowledge. Therefore we settled, once again, for a simple, not-completely-correct heuristic, that works most of the time. A large number of these null phrases (93.3%) occurred in two main syntactic constructions: "this ...{*makes* — *made*} it" and "it...{ *is* — *was* — *will be* — *etc.* }...that". Therefore, whenever we come across the pronoun "it", we check the occurrence to see if it falls into either pattern. If so, the heuristic decides that it is part of a null construction and does not look for any antecedent. This brings our pronoun resolution heuristic up to 73.7% accuracy (see table 12).

This is still less than the 88% accuracy claimed by Hobbs. However there are several assumptions Hobbs made about pre-processing and available information, which are not in our heuristic, and which could not be added without a great deal of difficulty. (Some of them require far too much knowledge to be added at all). These assumptions include:

- "it" was not counted when occurring in a time or weather construction.
- Hobbs only considered the pronouns *'he'*, *'she'*, *'they'* and *'it'*, while our heuristic was run for *all* pronouns

- In dialogue, implicit "A said to B" has been recovered in the text.
- In dialogue, there are special rules that exclude the speaker and listener as possible antecedents of third person pronouns inside the quotes. (Since his algorithm only looks at third person pronouns, he doesn't have special rules about first and second person pronouns inside the quotes. However presumably if he were to expand his heuristic to cover all pronouns, he would have special rules about this too.)
- In seeking an antecedent for "they", there are special rules that favor the conjuntion of two plurals over either plural, and also which accept plural and collective singular noun phases and *collects selectionally compatible entities*. The example that he gives of this is

  > John sat on the sofa. Mary sat before the fireplace. *They* faced each other.

  Since determining which objects are selectionally compatible requires quite a bit of knowledge, our heuristic does not handle such cases.
- The algorithm is assumed to be part of a larger system which recovers syntactically recoverable material and which has access to coreference and non-coreference relations. So, to quote Hobbs, "the algorithm also avoids choosing 'the man' as antecedent of 'him' in

  > John said his mother would sue the man who hit him.

  for 'the man' is necessarily coreferential with the omitted subject of 'hit', which is necessarily non-coreferential with 'him'."

  Again our heuristic has no access to such detailed knowledge. However it is interesting to note that, in the given example, it would nevertheless select the correct antecedent for "him".

3.4 "Smart" Pronoun Implementation Results

Table 12 shows the performance of the advanced heuristic, and table 13 shows a simple breakdown of the errors. Again the baseline totals have been included in table 12 for easy comparison. As can be seen, the improvement is quite dramatic. To get a better idea of how much improvement is due to

	Text 1	Text 2	Total	Baseline
Total no. of pronouns	46	49	95	95
Pronouns treated correctly	35 (75.6%)	34 (69.4%)	70 (73.7%)	29 (30.5%)
Errors	11 (24.4%)	15 (30.6%)	25 (26.3%)	66 (69.5%)

Table 12: Performance of the advanced algorithm for pronoun anaphora

	Text 1	Text 2	Total
Total Errors:	11	15	26
False Positives:	1	1	2
False Negatives:	0	2	2
Incorrect Antecedent Selected:	10	12	22

Table 13: Breakdown of the errors for the advanced pronoun algorithm.

just altering the search strategy and how much is due to just recognizing null constructions, table 15 shows how much each solution improves the baseline.

Table 14 shows a more detailed analysis of the errors made by the advanced pronoun algorithm. As indicated in the table, it might be possible to correct some of the errors without having to incorporate text understanding into the system. The rest of the errors require knowledge and understanding to be able to correct. The category "co-refers with subject of relative clause" should be self-explanatory. The errors arose from a plausible antecedent occuring in the sentence before the correct one. The same is true of the category "co-refers with second subject in compound sentence". The errors in this second category might be corrected if we had more complete parsing. In that case we could identify the individual sentences that make up the compound sentence, and for the sake of resolving pronoun co-reference we could treat each constituent sentence as a full separate sentence. Better parsing could also resolve the errors in the category "subject is not first np in sentence". In those cases, the pronoun co-refers with the subject of the sentence, but there is a clause that precedes the main subject in the sentence, so an incorrect but plausible antecedent is found in the clause. Null constructions were explained in detail in previous sections. The two listed in this table do not fall into the two patterns we used to identify null constructions. They are "*It* is not clear why..." and "*There* are many reasons...". It might be possible to identify null subjects of the first type by looking for the pattern "it is {adjective}...",

Cause of Error	Total	Possibly Correctable
Co-refers with Subject of Relative Clause:	3	0
Co-refers with Second Subject in Compound Sentence:	2	2
Subject is not First NP in Sentence:	5	5
Null Construction:	2	2
Co-refers with Direct Object:	6	0
Co-refers with Object of Preposition:	2	0
Other:	5	3
Total:	25	12

Table 14: Detailed analysis of errors made by the advanced pronoun algorithm.

	Baseline	Baseline & Correct Search	Baseline & Corrected 'Nulls'	Advanced
Pronouns treated correctly	29 (30.5%)	52 (54.7%)	44 (46.3%)	70 (73.7%)
Errors	66 (69.5%)	43 (45.3%)	51 (53.7%)	25 (26.3%)

Table 15: Intermediate performance results of the advanced algorithm for pronoun anaphora, showing the results of changing the search heuristic.

where {adjective} is a member of a list of adjectives that might be used in these null constructions[11]. However there may be times when such a pattern occurs, yet the pronoun nevertheless has an antecedent in the text. Similarly, it is easy to recognize the pattern "There {is — are}...", but it is hard to tell whether *there* is null in such constructions. The categories "co-refers with direct object" and "co-refers with object of preposition" should be self-explanatory. These errors cannot be corrected without some understanding of the text. The final category, "other" includes some errors which it might be possible to correct by writing small hacks. These include recognizing that "here" (when it has an antecedent) should be co-referring with a place; writing special rules to limit possible antecedents for reflexive pronouns; and, allowing noun phrases connected by a conjunction to be treated as a single plural noun phrase. The rest of the errors in that category require text understanding in order to correct.

[11] Such a list might include *easy, hard, tough, great, wonderful, difficult, etc.*

4 Conclusions and Future Work

Our experiments show that it is possible to design and implement a simple, easily scalable heuristic, using very little syntactic or semantic knowledge, which determines antecedents for definite noun phrases and pronouns with moderate accuracy (approximately 71%). By writing code to deal with certain special cases, and by running disambiguation on the texts we could bring the accuracy up to about 75%.

This is still well below the accuracy that Hobbs claimed for his "simple" heuristic for resolving pronoun co-reference. However upon closer examination it becomes apparent that while the actual search heuristic that Hobbs proposes is fairly simple, the amount of knowledge and pre-processing that he assumes is available to the system makes it anything but simple to truly implement.

Finally we noted that it is fairly easy to design simple solutions to certain problems. However it is frequently the case that the simple solutions, while solving some errors often introduce others. Therefore one needs to pay close attention to the tradeoffs involved.

Although they are not likely to increase the accuracy of our heuristic very much there are still a few interesting things that might be done. There are several types of errors (usually in the miscellaneous category) for which it would be possible to write small hacks. These include:

- *Conjunctions*. The heuristic currently does nothing with conjunctive phrases. For the most part conjunctions are treated as two separate simple noun phrases. It would not be too difficult to modify the program to consider conjunctions both as two (or more) separate phrases and as one large plural phrase. This would help in certain cases.

- *Single-Plural Matches*. Another type of error has to do with certain types of singular definite noun phrases arguably co-referring with plural indefinite phrases (our heuristic currently insists that number be the same in order for two phrases to match). An example of this occurs in the following text fragment:

  > Compared with these the tools from locality 15 appear to be more advanced, for many of them are made on *small flakes*

> shaped with skilled retouching, i.e. minor flaking applied after the detachment from the parent mass of *the flake* forming the body of the tool.

It might be possible to create a few rules that would handle such cases.

- *Special Phrases.* There are several unique definite noun phrases for which specific hacks might be written. The ones that appeared in our texts are: *the latter, these last, the majority, the rest, the other, the later, the one* and *the two.*

- *Quotes and Dialogue.* Special rules for the resolution of pronouns within quoted dialogue, perhaps analogous to (but simpler than) the rules used by Hobbs, could be devised.

There are also some more general issues that might be worth examining further. It would be interesting to loosen the requirement that synsets be within a distance of two to be considered for candidate matches. Instead one might try allowing any two synsets in a direct ancestral chain to be allowed as matches.

So far we have only run our program on two texts. The two texts chosen, while written in very different styles and on very diverse subject matter, are nevertheless both factual, technical documents. It would be useful to run the program on more texts, especially of various genres, to see if the results vary much.

Above all, we need to embed our anaphora resolution heuristic in some more general programs that use statistical word counts for automatically determining various aspects of texts, in order to see how much impact anaphora resolution really has.

References

[1] Charniak, E. "Toward a model of children's story comprehension." *AI TR-266* Massachusetts Institute of Technology Artificial Intelligence Laboratory

[2] Charniak, E., "Jack and Janet in Search of a Theory of Knowledge" *Advance Papers from the Third International Joint Conference on Artificial Intelligence*, Stanford, CA, 1973

[3] Church, K. "A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text. *Proceedings of the Second Conference on Applied Natural Language Processing* Austin, Texas, 1989

[4] Fillmore, Charles "Frame Semantics" *Linguistics in The Morning Calm* Seoul: Hanshin. 1982

[5] Fisher, D. "Topic Characterization of Full Length Texts Using Direct and Indirect Term Evidence" Technical Report No. UCB/CSD 94-809 University of California, Berkeley, May 1994

[6] Gordon, P., Grosz, B., Gilliom, L. "Pronouns, Names, and the Centering of Attention in Discourse" *Cognitive Science*, vol 17, pp. 311-47. 1993

[7] Grosz, Barbara "The Representation and Use of Focus in a System for Understanding Dialogs" *Proceedings of the Fifth International Joint Conference on Artificial Intelligence* Cambridge, MA, 1977

[8] Hartman, John H., Ousterhout, John K. "Zebra: A Striped Network File System" *Proceedings of the USENIX File Systems Workshop*, May 1992

[9] Hearst, M. "Context and Structure in Automated Full-Text Information Access" Doctoral Disseratation, University of California, Berkeley. 1994

[10] Hobbs, J. "Resolving Pronoun Reference" *Lingua* 44, pp311-338. 1978

[11] Miller, G.A., Beckwith, R., Fellbaum, C. Fross, D. and Miller K. J. "Introduction to WordNet: An On-line Lexical Data Base" *Journal of Lexicography*, vol 3, no. 4, pp 235-244, 1990.

[12] Schank, Roger C., Abelson, Robert P. "Scripts, Plans, Goals and Understanding: An Inquire into Human Knowledge Understanding" Hillsdale: Lawrence Erlbaum. 1977

[13] Sidner, Candice "Focusing in the Comprehension of Definite Anaphora" *Computational Models of Discourse*, M. Brady and R. Berwick, eds. MIT Press, Cambridge, MA 1983

[14] Watson, W. Early Civilization in China. New York: McGraw-Hill. 1966

[15] Webber, Bonnie "So What Can We Talk About Now?" *Computational Models of Discourse*, M. Brady and R. Berwick, eds. MIT Press, Cambridge, MA 1983

[16] Wilensky, Robert, personal communication. 1994.

[17] Yarowsky, D. "Word sense disambiguation using statistical models of Roget's categories trained on large corpora" *Proceedings of the Fourteenth International Conference on Computational Linguistics*, pp. 454-460, Nantes, France 1992

A Appendix: Rating of Candidate Noun Phrases

Given a particular definite noun phrase (Phrase A) and a candidate matching noun phrase (Phrase B), Phrase B is given a rating based on a word-by-word comparison between the two phrases. There are six variables used in the calculation: *Window-Distance, Exact-Head-Words, Exact-Other-Words, Wordnet-Head-Distance, Missing-Words* and *Different-Words*. The rating assigned to Phrase B is the sum of these six variables. The candidate phrase that has the highest rating is the one that is chosen for a antecedent.

Window-Distance is the number of sentences between Phrase B and Phrase A. This number is then multiplied by a weight of -0.25, to obtain the value for this variable. (The less distance, the better.)

Exact-Head-Words is either 2 (if the two phrases have exactly the same head word) or -1 (if the two head words are different).

Exact-Other-Words is the number of non-head words that are identical in the two phrases.

Wordnet-Head-Distance is an approximation of how close in meaning the two head words are. It depends on the relationship between the head word synsets in the Wordnet hierarchy. If the two head words are in the same synset, the value is 4; if they are in a parent-child relation, the value is 3; and, a grandparent relation gets the value 2.

Missing-Words is the number of words that are in one phrase but are skipped over in the other phrase. In order for a word to count as "skipped over", both the word that precedes it and the word that follows it must be in the other phrase. For example, given the phrases *the blue ball* and *the ball*, the word *blue* is counted as a missing. After all the missing words have been counted, this sum is multiplied by the weight -0.5 to obtain the value for this variable.

Different-Words is a calculation based on the number of words that are actually different[12] in the two phrases, and the wordnet distance between such words. For example, given the phrases *the angry man* and *the old man*, the words *angry* and *old* count as different words (one difference). The weight given to this pair of words depends on how far apart their synsets are in Wordnet. If they are in the same synset, the weight is 0; if they are

[12]Because it is quite common for a definite noun phrase to co-refer with an indefinite noun phrase, the initial determiners of the two phrases (A and B) are *not* counted as different words, even if they are different.

in a parent-child relation the weight is -0.25; if they are in a grandparent relation the weight is -0.5; and if they are in a sibling relation, the weight is -0.75. Each pair of different words is given a weight, and the weights are all summed up to obtain the final value of this variable.