Comparing Domain-Specific and Non-Domain-Specific Anaphora Resolution Techniques

Daniel Abadi
Churchill College
MPhil Dissertation
July 2003

1. Abstract

A quantification is provided for the improvements made in traditional salience-based pronominal anaphora resolution precision when input text that has been parsed using a large scale grammar to locate syntactic function of noun phrases, is used instead of input text where more shallow syntactic analysis techniques were used for identifying grammatical function. In addition, domain-specific techniques where domain knowledge can be automatically acquired from a database backend schema representation in an e-mail response system are examined as a means for further improvements in pronoun resolution.

2. Introduction

Anaphora, or the phenomenon of referring to some text evoking a real world entity, occurs at the discourse level of language processing. In many cases, if the same real world entity is referred to multiple times in the discourse, the structure of the referring expressions might differ from one another. For example, in the following text:

(a) I'd like to buy a Dell Inspiron 2600 laptop. But I only want the laptop if it comes with 256MB of RAM.

A Dell Inspiron 2600 laptop, *the laptop*, and *it*, all refer to the same real world entity. However each of these three referring expressions have different structures; the first is an indefinite noun phrase, the second is a definite noun phrase, and the third is a pronoun. But despite the different structures, at the semantic level, all three noun phrases should be analyzed as equivalent.

Resolution of these anaphora can play an important part in many natural language processing techniques. For example, if a search algorithm is able to recognize that there are three instances (rather than one) in the above text of *Dell Inspiron 2600 laptop,* then it can increment the term frequency of this phrase in the corresponding document, and the document would be more likely to be returned on searches for this phrase. Likewise, if an article is written about Bill Clinton but only uses his full name once at the beginning of the article, and subsequently refers to him as "Mr. Clinton", "he", "him", or "the former president", the search algorithm could use these references to the same entity to correctly boost the term frequency of *Bill Clinton* compared with the term frequency of other people who may have been mentioned in the article.

Pronoun resolution is particularly important in machine translation from languages such as English that have gender unspecific pronouns (like "*it*") to languages such as French or Spanish where pronouns have grammatical gender. The pronoun must first be resolved to the real world entity that it refers to, in order to infer the gender of the corresponding translated pronoun.

A third example of anaphora resolution techniques being used in natural language processing is in email response systems (this is the domain examined in this project) or dialogue systems. In order for these systems to communicate well with the user, a full semantic analysis must be performed on the user's input text. In example (a) above, the system must resolve the pronoun *it* to the *Dell Inspiron 2600 laptop* in order check (likely with a backend database) whether this machine comes with the requisite 256 MB of RAM before initiating a purchase order. In a flight purchase system, for example:

(b) User: I'd like to fly to Maastricht today
System: RyanAir Flight FR2292 leaves at 12:30 from London Stansted.
User: When does it arrive?

The system must resolve *it* to *RyanAir Flight FR2292* in order to respond correctly to this question.

Non-domain-specific approaches to anaphora resolution systems have been primarily syntactically based (Hobbs, 1978; Brennan and Pollard, 1987; Lappin and Leass, 1994; Kennedy and Boguraev, 1996; Siddharthan, 2003) where syntactic constraints such as person, number, and gender agreement on coreference are enforced, and then preferences such as proximity and grammatical category (subjects are preferred over existential emphasis which is preferred over direct objects, etc.) are used to choose between possible antecedents for referring expressions. Despite the reliance on syntactic details, the method for acquiring this information varies between algorithms. Hobbs (1978) relies on a full parse tree in order to resolve anaphora, Lappin and Leass (1994) rely on parsed text from a fairly shallow, broad-coverage parser, while Kennedy and Boguraev (1996) rely only on a part-of-speech tagger enriched with estimates of grammatical function, and Siddharthan (2003) is the most shallow of these examples and can guarantee an pronoun analysis for every text by using pattern matching on chunked text to infer grammatical function.

The advantage of using a robust technique in resolving anaphora is that an analysis can be generated without the computational cost of first deeply parsing the text, a process that would be impractical or impossible for some natural language processing techniques. In addition, anaphora could be resolved even for those sentences that no parses could be generated for. For instance, in the search example listed above, it would be impractical to parse all of the text documents just for the purpose of making the term frequency values more accurate. Further, errors are acceptable as long as they don't propagate too far. For this task, a shallow anaphora resolution technique that doesn't require the overhead of parsing the text and can provide analyses for a higher percentage of sentences is far more desirable.

On the other hand, if a parser is used, the syntactic information is more reliable than the information inferred from the shallower techniques, so it is assumed that the resolution precision should be higher. For the email-response and dialogue system examples listed above, a deep parse must be made anyway on the input text in order to generate a semantic representation of this text, so deeper anaphora resolution techniques should be used.

The following sections quantify the improvements in pronominal anaphora resolution precision that can be gained between a shallow, robust resolution technique (Siddharthan, 2003) and a deeper technique that relies on a full parse using the large scale LinGO English Resource Grammar (ERG) that provides detailed syntactic information about the input text.

In addition, a domain-specific algorithm was coded that uses knowledge about the types of entities upon which the text (in this case e-mails to an online technology retail company inquiring about orders made) might focus in order to resolve pronouns. The domain knowledge required for this algorithm is acquired automatically from a database backend so that this algorithm can be ported to other e-mail response systems over a different domain. This domain specific algorithm is evaluated both independently and together with the technique using domain-independent detailed syntactic information in order to analyze the compatibility of domain specific and non-domain-specific techniques.

Since the majority of work on anaphora resolution algorithms has focused on pronoun resolution (Jurafsky and Martin, 2000), evaluation of the domain-specific and non-domain-specific algorithms will only be on pronoun precision. The following section will examine previous pronominal anaphora resolution techniques that use only syntactic information and are thus domain independent. Section 4 will then describe the parser output representation language used for the non-domain-specific algorithm implanted for this project. Section 5 describes the e-mail corpus against which the pronoun resolution algorithms are evaluated. Section 6 will then describe the baseline system (the robust pronoun resolution algorithm not requiring a parser; Siddharthan, 2003) and Section 7 describes the extensions made to this baseline that use the detailed syntactic information produced by the parser, and that also use domain-knowledge. Section 8 describes how these algorithms were evaluated on the corpus and Section 9 provides the results. Section 10 analyzes these results and Section 11 concludes.

3. Non-Domain Specific Pronominal Anaphora Resolution Techniques

Pronoun resolution can be performed reasonably well using just syntactic knowledge about the pronouns and antecedents alone; using no domain knowledge or semantic analysis. Pronouns and potential antecedents are checked for agreement constraints, and then any remaining potential antecedents are ranked according a set of syntactic preferences. These constraints and preferences are listed below, followed by a description of three algorithms that implement some of these constraints and preferences.

One constraint that must hold when finding an antecedent for a pronoun is that the pronoun and antecedent must match in person and number. If the pronoun is third person and singular, then it can only refer to antecedents that are also third person and singular. Take for example:

(c) I^{1}'d like to buy a laptop^{2}. How much is it^{2}?
(d) I^{1}'d like to buy a laptop^{2}. Do you$^{??}$ have some in stock?
(e) I^{1}'d like to buy a laptop^{2}. How much are they$^{??}$ sold for?

Example (c) shows that the pronoun *it* has two possible antecedents: *I* and *laptop*. But since *it* is third person singular and *I* is first person, then the only possible antecedent for *it* is *laptop* (which is also third person singular). Example (d) shows that since the pronoun *you* is second person and the only possible antecedents are first or third person, this pronoun can not be resolved to an antecedent found in the text. Example (e) shows a problem with making these constraints too hard; since *they* is a plural pronoun and *a laptop* is singular, no match will be found.

Another constraint that must hold when finding an antecedent for a pronoun is that the pronoun and antecedent must match in gender and animacy. If a potential antecedent is of male gender, it is inappropriate to refer to this antecedent as *she*. Further, it is inappropriate to refer to an inanimate object as either *he* or *she* (rather: *it*). Take for example:

(f) I^{1} bought my wife^{2} a laptop^{3} yesterday. She^{2} doesn't want it^{3}.

The pronoun *I* can be eliminated for the potential antecedent list of both *she* and *it* using the person constraint listed above. But *wife*, *laptop*, *she*, and *it*, are all third person singular. Since *wife* is animate and female, this cannot be a possible antecedent for the pronoun *it*, and likewise for *laptop* and *she*. Thus in each case the correct antecedent can be found for each pronoun in this example.

However, in many cases the person, number, gender, and animacy constraints are not sufficient to filter a potential antecedent list down to 1 (or 0) possibilities. So a mechanism for choosing between potential antecedents must be implemented. There are four preferences that are typically applied to antecedent lists that don't require any domain-specific or semantic knowledge (Jurafsky and Martin, 2000). The first preference is that antecedents that are close to the pronoun being resolved are preferred to those antecedents that are further away. For example:

(g) My husband^{1} wants a desktop computer for his birthday. But my son^{2} thinks I should get a laptop. He^{2} says that laptops are easier to carry to and from work.

After filtering the potential antecedent list for person, number, gender, and animacy constraints for the pronoun *He* in the third sentence, the only possible antecedents for this pronoun are *husband* and *son*. But *son* is (correctly) preferred because it was used more recently than *husband*.

A second preference for potential antecedents is their grammatical function. Antecedents that were used inside noun phrases that served as the subject of a sentence are preferred over antecedents that were used as direct objects which are preferred over antecedents that were used as indirect objects which are preferred over antecedents used in other types of grammatical roles. For example:

(h) The mouse[1] that came with my computer[2] doesn't work. It[1] needs to be replaced.

In this example, the antecedent *mouse* is preferred to *computer* for the pronoun *It* because *mouse* appears in the subject position of the sentence.

A third preference used to choose between potential antecedents is repeated mention of the same antecedent. If the same antecedent is used to refer to an entity multiple times, it's more likely that a pronoun might be used to shorten the referring expression. For example:

(i) The computer[1] I bought came with a free keyboard[2] and mouse[3]. But I think the mouse[3] is not working properly, even though everything else works great. Can you replace it[3]?

Since *mouse* is mentioned twice (and no other potential antecedent is mentioned more than once), it is likely that the pronoun in the third sentence refers to *mouse*.

A final preference that can be applied to choose between potential antecedents is that a pronoun can often be used in parallel grammatical roles with its antecedent. For example:

(j) The laptop[1] I bought last week came with a 512MB of RAM[2]. Does the computer[3] currently advertised on your website[4] come with it[2] as well?

Even though *RAM* is never the subject of any sentence, it's not mentioned repeatedly, and is not the most recent potential antecedent to the pronoun *it*, it is preferred over the other potential antecedents because it is used in parallel with the pronoun.

None of these listed constraints and preferences require any domain knowledge, and thus can apply to pronoun resolution in any domain. Three simplified versions of non-domain-specific algorithms that use some or all of these constraints and preferences will be briefly discussed: Hobbs, 1978; Brennan et al., 1987; and Lappin and Leass, 1994.

The Hobbs, 1978, algorithm works directly from a parse tree of the input text, searching this tree looking for potential noun phrase antecedents. The recency and grammatical role preferences are implemented by the order in which the parse trees are searched. First the current tree is searched, and then the trees for each previous sentence in order of recency starting with the most recent are searched. Each tree is searched from left to right, which means that the subject NP will be encountered before the direct object NP, which will be encountered before noun phrases in other syntactic roles in typical S → NP VP parse trees. Number and gender constraints are then applied to NP antecedent proposals in the order in order in which they are searched.

The Brennan et al., 1987 algorithm used centering theory to choose between antecedents. Utterances are ordered (U_n, U_{n+1}, U_{n+2} …) and after each utterance, the center of that utterance is determined using an ordered list of entities mentioned in the previous utterance (the order is by grammatical role), the center of the previous utterance, and the proposed list of entities mentioned in the current utterance after the pronouns have been resolved. Pronouns are resolved so that the proposed list of entities mentioned in the current utterance results in a center for the current utterance that results in no (or if this is not possible then a smooth) shift from the center of the previous utterance. Thus pronouns are resolved in this algorithm using only the grammatical role of the entities inside utterances and the order of these utterances. So like the Hobbs 1978 algorithm, it accounts for the recency and grammatical role preferences. It also accounts for the repeated mention preference because pronouns are often resolved to the entity being centered upon if that entity was centered upon in the previous utterance.

A third algorithm that used syntactic information alone to resolve pronouns is Lappin and Leass, 1994. While Hobbs 1978 and Brennan et al. 1987 indirectly implemented the potential antecedent preferences, Lappin and Leass implemented a point system that directly indicated which antecedent a pronoun should be resolved to given the recency, grammatical function, repeated mention, and parallel use preferences. The recency preference was implemented by incrementing the point score for each entity by 100 points for each sentence that an entity was mentioned in; but the point score for this entity was divided in half with each new sentence between the referral to the entity and the pronoun. The grammatical function preference was implemented by incrementing the point score for that entity depending on the grammatical role for its referring expression. Subjects received 80 points, existential emphasis received 70 points, direct objects received 50 points and indirect objects received 40 points. Lappin and Leass used equivalence classes in order to implement the repeated mention preference. The number of points given to an entity is set to be equal to the highest value of any member of that entity's equivalence class (e.g. if an entity is referred to in both the subject and object of the sentence, that entity gets the subject point factor for that sentence). Further if the same entity is referred to in multiple sentences, this entity's point factor gets increased for each sentence (scaled down for recency). Finally, Lappin and Leass implement the parallel grammatical role preference by incrementing the point factor for a pronoun-entity pair by 35 points if their grammatical roles are identical.

4. Minimal Recursion Semantics

The syntactic information needed the for non-domain-specific pronoun resolution was derived for this project from output of text parsed using the LinGO English Resource

Grammar (a large freely available computational grammar of English). The parsed output is expressed in MRS (Minimal Recursion Semantics) feature structures: a semantic representation of the text that is linked to its syntactic information. The assumption behind MRS output structures is that the fundamental unit of interest in the semantic representation of a sentence can be expressed in elementary predications (EPs) where an EP consists of a single relation (generally a lexeme) and its associated arguments. These relations cannot be embedded inside one another directly, instead each EP has a handle and these handles can be used as arguments of quantifiers and other relations in which a relation would be embedded. The MRS output for an example sentence parsed using the LinGO ERG is given below (see also Appendix A):

```
[ LTOP: h1
 INDEX: e2 [ EVENT
             E.TENSE:  PRESENT*
             E.ASPECT:  NO_ASPECT*
             E.MOOD:  MODAL_SUBJ*
             DIVISIBLE:  BOOL ]
 RELS: <
        [ int_m_rel
          LBL: h1
          CFROM: STRING
          CTO: STRING
          MARG: h3 ]
        [ prpstn_m_rel
          LBL: h3
          CFROM: STRING
          CTO: STRING
          MARG: h4 ]
        [ _could_rel
          LBL: h5
          CFROM: 0
          CTO: 1
          ARG0: e2
          ARG1: h6 ]
        [ pron_rel
          LBL: h7
          CFROM: 1
          CTO: 2
          ARG0: x8 [ FULL_REF-IND
                     DIVISIBLE:  -
                     PNG.PN:  2PER
                     PNG.GEN:  REAL_GENDER
                     PRONTYPE:  STD_PRON ] ]
        [ pronoun_q_rel
          LBL: h9
          CFROM: STRING
          CTO: STRING
          ARG0: x8
          RSTR: h10
          BODY: h11 ]
```

```
          [ _speed_v_rel
            LBL: h12
            CFROM: 3
            CTO: 4
            ARG0: e13 [ EVENT
                        E.TENSE:  NO_TENSE
                        E.ASPECT:  NO_ASPECT*
                        E.MOOD:  INDICATIVE*
                        DIVISIBLE:  BOOL ]
            ARG1: x8 ]
          [ _up_rel_ind
            LBL: h5
            CFROM: 4
            CTO: 5
            ARG0: e14 [ EVENT
                        E.TENSE:  TENSE
                        E.ASPECT:  ASPECT
                        E.MOOD:  MOOD
                        DIVISIBLE:  BOOL ]
            ARG1: e2
            ARG2: x15 [ REF-IND
                        DIVISIBLE:  +
                        PNG.GEN:  NEUT*
                        PNG.PN:  3SG* ] ]
          [ bare_div_q_rel
            LBL: h16
            CFROM: STRING
            CTO: STRING
            ARG0: x15
            RSTR: h17
            BODY: h18 ]
          [ _delivery_rel
            LBL: h19
            CFROM: 5
            CTO: 6
            ARG0: x15
            ARG1: v20 [ NON_EXPL
                        DIVISIBLE:  BOOL ] ] >
 HCONS: <   h4 QEQ h5
            h6 QEQ h12
            h10 QEQ h7
            h17 QEQ h19 > ]
```

Figure 1: MRS for *Could you please speed up delivery?*

Figure 1 shows the MRS for the input sentence: *Could you please speed up delivery*. The MRS output for a sentence has four features: LTOP, INDEX, RELS, and HCONS. LTOP contains the handle name of the outermost EP in the sentence. In the example in figure 1, the outermost EP is the int_m_rel, indicating that the sentence is an interrogative, so the value of LTOP is the handle h1. The INDEX feature stores the event variable describing the “main” (or semantically prominent) event of the sentence. Following the INDEX feature is the RELS list, or a list of all of the EP relations created during the semantic analysis. Each of these relations contain a handle (under the LBL feature), a link to the text token in the input string (the CFROM and CTO features) and arguments to that relation (the ARG0, ARG1, … features). After this list of EPs, the HCONS list specifies how relation trees can be constructed from the handles of component relations.

For the purposes of pronoun resolution, the important part of MRS representations are the relations in the relation list that correspond to tokens in the input text (especially the *pron_rel* relation which correspond to pronouns), the object ("x") variables that correspond to entities that are potential antecedents, and the arguments to verb relations that describe the grammatical role of these entities. The algorithms for doing this, along with further example MRS representations will be presented in section 7.

5. The Corpus

The domain for this project is e-commerce, where queries have been written in e-mail form inquiring about or attempting to cancel orders made at the online electronics retail website. 1595 such e-mails had been acquired in a Wizard-of-Oz experiment (for the original intention of implanting an automatic e-mail response system), and these e-mails consist of the text corpus used for this project. These e-mails were then parsed using the LinGO ERG grammar, and for each e-mail, as soon as a sentence was reached that would not parse, this sentence and the remaining sentences in the e-mail were discarded. The reason for this is that the pronoun resolution algorithms implemented (aside from the baseline system) require, as input, parsed sentences represented in MRS. But if a sentence does not parse, no MRS can be produced for that sentence. Even if there are parsable sentences after this sentence, the antecedents that pronouns in future sentences might refer to could be located in the unparsed sentence, so future sentences are also discarded. E-mails for whom the first sentence does not parse are completely discarded.

The remaining data is split into two categories. E-mails that inquire about order status, and e-mails that request that an order be canceled. The former corpus (consisting of 886 e-mails) were used as training data and the latter (consisting of 709 emails) as test data. Since most e-mails were short and did not contain many pronouns, there were only 178 third person pronouns in the training data and 168 third person pronouns in the test data (only third person pronouns are resolved on this data set because the first person pronouns can be trivially resolved to the e-mail sender and the second person pronouns can be resolved to the company to whom the e-mails were sent).

6. Baseline System

The baseline system used to resolve pronouns in this corpus was the system implemented in Siddharthan, 2003. The techniques used in this system are shallower than any of the systems that have been thus far discussed, only requiring that the input text be run through a part-of-speech tagger and noun chucker. Grammatical roles of noun phrases are then inferred using an ordered sequence of simple pattern matching rules. This sequence of rules is given below (from Siddharthan, 2003):

1. Prep NP^{obliq}
2. NP^{subj} [", [^Verb]+," | "Prep NP"]* Verb
3. Verb NP^{dobj}
4. Verb [NP]+ NP^{iobj}

These patterns are sufficient for identifying the subject noun phrase (rule 2) with 88% precision, the direct object noun phrase (rule 3) with 89% precision and indirect objects (rule 4) with 26% precision. The precision for indirect objects is low because oblique noun phrases found using rule 1 are pooled with the noun phrases found using rule 4 in order to increase recall (to 89%) of indirect object noun phrases. The distinction between indirect objects and oblique noun phrases is irrelevant for a Lappin and Leass type pronoun resolution algorithm because both receive the same salience factor for grammatical role. Siddharthan shows that the precision, recall, and F-measure of inferring the grammatical function of noun phrases using this pattern matching method is comparable to the determination of grammatical function using parsers. The advantage of not using a parser is that this method can resolve pronouns for all sentences; not just for those sentences that a parser is able to analyze. However, for the purposes of this experiment, since the input text in an e-mail response domain will have to be parsed anyway, this baseline algorithm will be run on the same dataset as for the extension algorithms, which, as described above, consist of all e-mails in the corpus up until the first sentence that does not parse.

After inferring grammatical function of the noun phrases, Siddharthan (2003) implemented a Lappin and Leass type algorithm using the same salience factors as Lappin and Leass, 1994. As in Lappin and Leass 1994, a list of possible antecedents is created and filtered for person, number, gender, and animacy. The part-of-speech tagger provides the person and number information about pronouns and potential antecedents, but provides no information about gender or animacy. For this reason, Siddharthan infers these features using a variety of techniques. The gender and animacy of proper nouns can sometimes be inferred by searching for keywords in the noun phrase. For example, if the noun phrase is *Mrs. Bush*, then the gender of this entity, along with the gender of every entity in its equivalence class, can be inferred to be of female gender and animate. For common noun phrases, WordNet is used to check whether the head noun is a hypernym of *human*, *animal*, or *organization*. In such a case, the noun phrase can be inferred to be an animate object (otherwise it is left unspecified). WordNet can sometimes provide gender information as well. The baseline algorithm also checks for keywords in appositives and existential constructs that might indicate whether the corresponding noun phrase is animate, and verbs for whom the subject must be animate (such as *said*, *reported*, or *stated*) also can help with the gender and animacy deduction.

The baseline algorithm thus is similar to Lappin and Leass, 1994, except that it does not require a parser, and uses WordNet and other techniques to help infer agreement values for the antecedent filter.

7. Extension Systems

Two separate extensions to the shallow baseline algorithm were implemented and evaluated separately, and then evaluated when run in combination. The first extension used the MRS output from the parsed text (using the LinGO ERG) to acquire detailed syntactic information. A Lappin and Leass (1994) type algorithm was then implemented using this syntactic information. The second extension used domain knowledge where noun phrase potential antecedents are obtained automatically from a database backend and are used to increase pronoun resolution precision. Each of these extensions will be discussed in turn.

As described above, the Lappin and Leass (1994) algorithm resolved pronouns in two steps. The first step was to create a discourse model containing a list of potential noun phrase antecedents and an associated salience value for each phrase calculated from a list of preferences derived from their grammatical function. The second step then filtered this list of potential antecedents with respect to each pronoun being resolved using some syntactic constraints (such as person, number, and gender agreement) along with altering the global salience score with additional scores specific to that particular pronoun (if the pronoun was used in parallel with the antecedent or was cataphoric). A similar algorithm was implemented as an extension to the baseline algorithm, where the grammatical function information and person, number, and gender information were inferred from the semantic analysis (in MRS) of the input text generated by the parser. An explanation of how this syntactic information is inferred is given in the example below:

```
;;; MRS for: I still haven't received my order.
 [ LTOP: h1
  INDEX: e2 [ EVENT
              DIVISIBLE:  BOOL
              E.TENSE:  PRESENT*
              E.ASPECT:  PERF*
              E.MOOD:  INDICATIVE* ]
  RELS: <
          [ prpstn_m_rel
            LBL: h1
            CFROM: STRING
            CTO: STRING
            MARG: h3 ]
          [ pron_rel
            LBL: h4
            CFROM: 0
            CTO: 1
            ARG0: x5 [ FULL_REF-IND
                         DIVISIBLE:  -
                         PNG.PN:  1SG
                         PNG.GEN:  REAL_GENDER
                         PRONTYPE:  STD_PRON ] ]
          [ pronoun_q_rel
            LBL: h6
            CFROM: STRING
            CTO: STRING
            ARG0: x5
            RSTR: h7
            BODY: h8 ]
          [ _still_rel
            LBL: h9
            CFROM: 1
            CTO: 2
            ARG1: e2 ]
          [ neg_rel
            LBL: h9
            CFROM: STRING
            CTO: STRING
            ARG1: h10 ]
          [ _receive_rel
            LBL: h11
            CFROM: 3
            CTO: 4
            ARG0: e2
            ARG1: x5
            ARG2: x12 [ REF-IND
                         DIVISIBLE:  STRICT_BOOL
                         PNG.GEN:  NEUT*
                         PNG.PN:  3SG* ] ]
```

```
          [ def_explicit_q_rel
            LBL: h13
            CFROM: 4
            CTO: 5
            ARG0: x12
            RSTR: h14
            BODY: h15 ]
          [ pro_poss_rel
            LBL: h16
            CFROM: STRING
            CTO: STRING
            ARG0: e18 [ EVENT
                          E.TENSE:  NO_TENSE
                          E.ASPECT:  ASPECT
                          E.MOOD:  MOOD
                          DIVISIBLE:  BOOL ]
            ARG1: x17 [ FULL_REF-IND
                          PNG.PN:  1SG*
                          PNG.GEN:  REAL_GENDER
                          PRONTYPE:  STD_PRON
                          DIVISIBLE:  -* ]
            ARG2: x12 ]
          [ pronoun_q_rel
            LBL: h19
            CFROM: STRING
            CTO: STRING
            ARG0: x17
            RSTR: h20
            BODY: h21 ]
          [ pron_rel
            LBL: h22
            CFROM: STRING
            CTO: STRING
            ARG0: x17 ]
          [ _order_n_rel
            LBL: h16
            CFROM: 5
            CTO: 6
            ARG0: x12
            ARG1: v23 [ NON_EXPL
                          DIVISIBLE:  BOOL ] ] >
  HCONS: <  h3 QEQ h9
            h7 QEQ h4
            h10 QEQ h11
            h14 QEQ h16
            h20 QEQ h22 > ]
```

Figure 2: MRS for: ***I still haven't received my order*.**

Figure 2 shows the MRS typed feature structure output for the input text: "I still haven't received my order. The first stage of the algorithm identifies all possible noun phrase antecedents. These antecedents will typically have an object variable associated with the corresponding relation. This example has two pronouns (*I* and *my*) and one noun phrase (*order*) so there are three antecedents that need examination. Object variables will be of type FULL_REF-IND or REF-IND, so in the figure above, variables x5, x12, and x17 are all object variables (the convention is for these variable names to begin with the letter 'x'). When a variable is first introduced, its typed feature structure is displayed, where information about the object can be obtained. For the purposes of pronoun resolution, the key information that needs to be extracted from the variable feature structure are the values for the PNG.PN and PNG.GEN features. PNG.PN yields information about person and number, while PNG.GEN yields information about the grammatical gender of this object. For example, for the variable x12 (which will later be inferred to correspond to the *order* noun phrase) has the PNG.PN feature type of 3SG* and PNG.GEN has a type of NEUT*[1]. Thus x12 corresponds to a neutral object that is third person and singular. These values will be stored and used later for the constraint-based part of the pronoun resolution algorithm.

The next step of the algorithm is to associate these objects with a relation and then to associate this relation with its corresponding token in the input text. Both of these steps are non-trivial. Most relations in the MRS output will have a list of arguments to the relation (in the form of ARG0, ARG1, … features) where ARG0 feature lists the variable directly associated with that relation. Some relations with have object variables as an ARG0 (eg. nouns, determiners, and quantifies), some will have event variables (eg. verbs, adjectives, and prepositions), and some will not be able to specify an association with a defined variable. What makes this task non-trivial is that the same object variable might serve as the ARG0 for multiple relations. For instance, for definite noun phrases, the object variable associated with the head noun will also be associated with the determiner. Likewise, for indefinite noun phrases, the object variable will also both be associated with the head noun and the quantifier. To choose which relation to associate the object with, the relation name is passed to a procedure that decides whether the relation is likely the main noun corresponding with the object. This procedure uses cues such as the part of speech tag in the relation name (some nouns will have a '_n_' embedded in the relation name such as _order_n_rel above, and quantifiers will have a _q_ embedded in the relation name) to drop quantifiers and use nouns. If the procedure is unable to deduce from just the relation name whether the relation is likely the head noun of the corresponding object, the default return value is that it is a possible head noun relation; and if (in the rare case) more than one relation is tagged as a possible head noun relation then the most recent tagged relation used is assumed to be associated with that object.

Once the object variable has been associated with a relation in the MRS output structure, this relation must still be associated with the corresponding token in the input text. For some pronoun resolution applications, this would not be necessary. For instance, if the purpose was a full semantic analysis of the text in the context of an email response system, the object variables for pronouns would be set equal to the object variables of their antecedents, their types unified, and the antecedent relation name would be substituted for the *pron_rel* relation. In contrast, for the search application explained in the introduction, setting the semantic equivalency of the antecedent and pronoun would not be enough; the token in the corresponding input text must be found so that its term frequency can be incremented. However, the traditional mechanism for evaluating the success of the pronoun resolution is to mark the noun phrase antecedent in the input text using some index value, and then to coindex any pronouns that refer to the same referent as this antecedent with the same index value. Thus, despite the fact that the data used in these experiments are obtained from an e-mail response domain, in order to evaluate the success of the pronoun resolution algorithms (independently from measuring the success

[1] The asterisks in these type names allow for further sub-types of these PN and GEN features, but for the purposes of this algorithm the asterisks are ignored.

of the semantic representation of the input e-mails or the quality of the response emails), the input text token corresponding with the chosen antecedent and pronoun relations must be located and appropriately indexed.

To accomplish this task, the CFROM and CTO features of each relation are extracted, which correspond to the input text token where the relation begins and ends respectively (counting from 0). So in the example in figure 2, the first *pron_rel* relation has a CFROM of 0 and a CTO of 1, meaning that the first token of the input text (which is *I*) corresponds with this relation. However, only five out of the eleven relations in figure 2 have defined values for the CFROM and CTO features (the rest have the unspecified type *STRING*), and in general there are usually more relations than there are input tokens. This is often because multiple relations are needed to semantically analyze one token (for example, in figure 2 above, the relations *def_explicit_q_rel*, *pro_poss_rel*, *pronoun_q_rel*, and *pron_rel* all are associated with the token: *my*). To approximate a solution to this problem, the relation-to-token algorithm assumes that if the CFROM and CTO features are unspecified for a particular relation, they are equivalent to the most recent relation for whom these features are defined. This assumption is quite frequently not true because in many cases a relation that is associated with a token appears in the relation list before the element in the relation list that contains the defined CFROM and CTO features for that token. However, for the purposes of finding tokens associated with relations that contain *objects* as their ARG0, this assumption almost always holds (at least for the parsed sentences of the training e-mail response data) and the correct token is identified.

At this point in the algorithm, those input text tokens that correspond with object variables in the semantic analysis (and these tokens are assumedly nouns located inside noun phrases) have been identified and can be indexed appropriately. The next step of the algorithm is to calculate the salience of these objects inside the discourse model (similarly to Lappin and Leass, 1994) by identifying their grammatical function. This is done by locating the relation that contains the main sentence event as its ARG0. This event is listed as the variable associated with the INDEX feature of the MRS output (listed at the beginning; before the relation list). In the example in figure 2, the main event variable is e2, and the relation that contains e2 as its ARG0 is *_receive_rel*. In general, the relations that contain events as their ARG0 can correspond to a variety of parts of speech (such as verbs, adjectives, or prepositions). However, in most cases, only relations corresponding to verbs or prepositions will have other arguments in addition to ARG0. A procedure checks relation names with list of names corresponding to prepositions in order to filter out these prepositions, so that if a relation contains the main event as it's ARG0, contains other arguments in addition to ARG0, and is not listed as a prepositional relation, it can be assumed that this is the index verb of the sentence. The relation *_receive_rel* in figure 2 fulfills all of these requirements and is deduced to be the index verb of the sentence *I still haven't received my order*.

Once the index verb of the sentence has been located, finding the subject and objects of the sentence is straightforward. In the MRS representation of the semantic analysis of the sentence, the ARG1 of verb relations correspond to the subject of those verbs, and subsequent arguments correspond to objects (ARG2 is the direct object and ARG3 is the indirect object if they exist). If these arguments have as their values object ("x") variables that have been analyzed and extracted, these object variables are given appropriate salience factors in the discourse model. These salience factors are similar to the Lappin and Leass factors, but their magnitudes are generally smaller. Subjects are given a factor of 40, direct objects (and also existential emphasis by the nature of their being interpreted as the direct object of the *_cop_id_rel* to be relation) are given a factor of 20, indirect objects are given a factor of 8 and all other objects are given a factor of 5. So for the example in figure 2, since x5 and x12 are the ARG1 and ARG2 of the index verb *_receive_rel*, they receive saliency factors of 40 and 20 respectively. After calculating the salience factor derived from the grammatical function of every object in each sentence, the salience factor of objects that are listed as the ARG2 of prepositional relations are decreased by dividing its salience factor by 2.

Conjunctive sentences complicate this process because they don't have a true index verb, and thus don't have a subject, object, etc. for the entire sentence. Appendix A shows the MRS output for a sample conjunctive sentence: "*The first one arrived yesterday, but I'm still waiting for the second one*". These types of sentences are treated as two separate sentences where the left-side of the conjunct has a main event and the right-side has a main event. These events are identified using the following algorithm. The INDEX feature for conjunctive sentences will be of type CONJ_EVENT. This index variable is then searched for as the C-ARG of the relation corresponding to the conjunction. In the example in Appendix A, this relation is *_but_rel*. The handle variable value for the L-HNDL and R-HNDL features are extracted. If these handle variables appear as the LBL value for a verb relation, then this verb is assumed to be the index verb of that side of the conjunction and the corresponding ARG0 of that relation is assumed to be the index event. If these handle variables appear as the LBL value for a proposition (*prpstn_m_rel*), then the MARG feature of this proposition is extracted (which will be a handle variable), and this variable is looked up in the HCONS list. The variable will appear on the left hand side of a QEQ (equality modulo quantifier) constraint. The right hand side of this QEQ will also be a handle variable, and if a verb relation contains this handle variable as its LBL, then this verb is assumed to be the INDEX verb and its ARG0 the index event. So, for the example in Appendix A, the L-HNDL for the conjunction relation *_but_rel* is h3. The relation that has h3 as its LBL is a *prpstn_m_rel* with an MARG of h4. The HCONS list shows h4 QEQ h11 and h11 is the LBL for the _arrive_rel relation (along with the _unspec_loc_rel relation, but _arrive_rel is the verb). Thus _arrive_rel is the index verb, and the salience factors for its object arguments (in this case it only has one argument, ARG1 - its subject) are updated. In this case, the salience factor of x7 (which had previously been inferred to correspond to the token: *one*) is increased by a value of 40. Likewise, the R-HNDL for _but_rel is h19 which is also a label for a proposition, and a similar chain of deduction leads the relation _wait_v_rel to be inferred as the index verb for the right side, and the salience factor of its subject, x23, is incremented.

Thus the salience factors for the possible antecedent objects are calculated for the discourse model. The differences between the discourse model representation used for this algorithm and the discourse model used in Lappin and Leass' algorithm should be discussed. The primary difference is that the model used for this algorithm does not use equivalence classes to store previously resolved pronouns in the same place in the model as their antecedents. Lappin and Leass use equivalence classes as a means of giving preference to those entities that have been mentioned repeatedly in the discourse. The salience factor of that entity is set to be equal to the highest value of any member of that entity's equivalence class (e.g. if an entity is referred to in both the subject and object of the sentence, that entity gets the subject saliency factor for that sentence). Further if the same entity is referred to in multiple sentences, this entity's salience factor gets increased for each sentence (scaled down for recency). However, for this e-mail query domain, the discourse rarely reaches more than two or three sentences so it is unusual for the same entity to be referred to more than twice. So instead of using equivalence classes as a means of giving preference to those entities that have been mentioned repeatedly, the algorithm uses a cruder method: it assigns pronouns a higher saliency factor (incrementing its value by 15) in the discourse model since a pronoun will be at least the second reference to its referent (except in the rare case that the pronoun is cataphoric). Thus if there is a third pronominal reference to the same referent, the algorithm will be more likely to resolve this pronoun to have the same antecedent as the preceding pronoun.

Another difference between the discourse model used for this algorithm and the Lappin and Leass discourse model is that Lappin and Leass give preference to head nouns as antecedents by directly incrementing head nouns by a salience factor of 80. The MRS based algorithm performs the same task, but less directly. The object arguments of verbs will always correspond to the head object of the noun phrase argument (assuming a correct parse) in the MRS representation. So if an object is not a head noun, it can not receive additional salience weights for its grammatical function, so its maximum salience factor will be 5.

Once the discourse model has been updated for an input text sentence, pronouns can be resolved. Since all first person pronouns can be trivially resolved to the e-mail sender in this domain, only third person pronouns are resolved. For the example in figure 2, none of the three objects were third person pronouns (there were two first person pronouns and the third object was a normal noun). So after analyzing this sentence, this discourse model contains three variables: x5 (which has been marked as corresponding to the token: *I*) with a salience factor of 40, x12 (which has been marked as corresponding to the token: *order*) with a salience factor of 20, and x17 (which has been marked as corresponding to the token: *my*) with a salience factor of 5. The next input sentence is then read in and analyzed. This sentence can be found in figure 3, below.

```
;;; MRS for: It is a brand new cordless phone.
[ LTOP: h1
  INDEX: e2 [ EVENT
              DIVISIBLE:  BOOL
              E.TENSE:  PRESENT*
              E.ASPECT:  NONPRG+NONPRF
              E.MOOD:  INDICATIVE* ]
  RELS: <
          [ prpstn_m_rel
            LBL: h1
            CFROM: STRING
            CTO: STRING
            MARG: h3 ]
          [ pron_rel
            LBL: h4
            CFROM: 0
            CTO: 1
            ARG0: x5 [ FULL_REF-IND
                         DIVISIBLE:  -
                         PNG.PN:  3SG
                         PNG.GEN:  NEUT*
                         PRONTYPE:  STD_PRON ] ]
          [ pronoun_q_rel
            LBL: h6
            CFROM: STRING
            CTO: STRING
            ARG0: x5
            RSTR: h7
            BODY: h8 ]
          [ _cop_id_rel
            LBL: h9
            CFROM: 1
            CTO: 2
            ARG0: e2
            ARG1: x5
            ARG2: x10 [ FULL_REF-IND
                         PNG.GEN:  REAL_GENDER
                         PNG.PN:  3SG*
                         DIVISIBLE:  -*
                         PRONTYPE:  REAL_PRON ] ]
```

```
          [ _a_q_rel
            LBL: h11
            CFROM: 2
            CTO: 3
            ARG0: x10
            RSTR: h13
            BODY: h12 ]
          [ _brand_new_rel
            LBL: h14
            CFROM: 3
            CTO: 5
            ARG0: e15 [ EVENT
                         E.TENSE:  TENSE
                         E.ASPECT:  ASPECT
                         E.MOOD:  MOOD
                         DIVISIBLE:  BOOL ]
            ARG1: x10 ]
          [ _cordless_rel
            LBL: h14
            CFROM: 5
            CTO: 6
            ARG0: e15
            ARG1: x10 ]
          [ _phone_rel
            LBL: h14
            CFROM: 6
            CTO: 7
            ARG0: x10 ] >
  HCONS: <  h3 QEQ h9
            h7 QEQ h4
            h13 QEQ h14 > ]
```

Figure 3: MRS for: *It is a brand new cordless phone*.

Upon analyzing this sentence, the discourse model is first updated. This sentence contains two objects: x5 (corresponding to the token: *It*) and x10 (corresponding to the token: *phone*). x5 is inferred to be the subject of this sentence and receives a saliency factor of 40. x10 is the direct object and receives a saliency factor of 20. After updating the discourse model, pronouns can be resolved. This sentence has one pronoun: the x5 object. Since it is third person, an attempt will be made to resolve it. Since the token corresponding to this object occurs at the beginning of this sentence (and cataphoric pronouns are rare), the algorithm only looks to the previous sentence for possible antecedents. The previous sentence contains three objects and thus three possibilities for antecedents. The first step of the pronoun resolution algorithm is to check whether the pronoun is pleonastic. The parser catches many cases of pleonastic pronouns and indicates that a pronoun is pleonastic by not assigning a relation of *pron_rel*, and instead ignoring this pronoun altogether. However, in some cases the parser is unable to decide whether a pronoun is pleonastic and will parse the pronoun to the *pron_rel* relation. So to supplement the parser's decision on pleonastic pronouns, a simple pleonastic filter is used to approximate whether a pronoun is pleonastic. This filter checks the next few tokens, and will resolve the pronouns in the following phrases to be pleonastic: *it's been*, *it has been*, *it possible*, and *it impossible*. If a pronoun is pleonastic, the algorithm marks it as unresolvable, and does not attempt to find an antecedent for it.

If the pronoun is not determined to be pleonastic, the resolution algorithm filters the list of possible antecedents for person, number, and gender agreement. The PNG.PN and PNG.GEN features, stored with the object variable are unified with the PNG.PN and PNG.GEN features of the pronoun being resolved. If unification is successful, then a new salience factor is calculated for each antecedent with the current pronoun. This new salience factor is equal to the old saliency factor, divided by (2 * the number of sentences between the current sentence and the sentence containing the potential antecedent), unless the potential antecedent and the pronoun objects appear as arguments for the same verb, in which case the new salience factor is 0. For the example in figure 3, the possible list of antecedents of the x5 pronoun is filtered down to only x12 (corresponding to the *_order_n_rel* relation) because the other two objects are of type 1SG and do not unify with the third person pronoun. The new salience factor for this antecedent-pronoun match is equal to the salience factor of 20 stored in the discourse model, divided by 2, since *_order_n_rel* occurs in the previous sentence. Thus the index for the pronoun *It* in figure 3 will be correctly coindexed with the index for the antecedent *order* in figure 2.

This pronoun resolution algorithm thus acquires detailed syntactic information from the semantic representation of the parsed input text. It accounts for the syntactic constraints of person, number, and gender agreement when that information is available and then uses the recency, repeated mention, and grammatical function preferences to choose between possible antecedents, similarly to Lappin and Leass, 1994. This algorithm does not use any domain knowledge, except in the crude way that the repeated mention preference was implemented because of the assumption of short input text.

A separate domain-specific algorithm for pronoun resolution was also implemented. The fundamental assumption behind the domain specific anaphora resolution algorithm is the following: A person writing an email to a company asking for information about a question assumes that the person responding to the e-mail is not significantly smarter than himself. So the only real difference between the person asking the question and the person responding to it is that the person responding to it has, at his disposal, data in some form that the writer does not have access to. So the email writer is intuitively aware that the person responding to the e-mail will use this data source to respond to this question, and thus will naturally place the focus of the question on this data source.

However, an important part of a pronoun resolution algorithm is to locate the focus of the sentence, since the pronouns usually refer to this focused element (Brennan et. al. 1987). Thus, because these emails tend to focus on some intuitive data source, the assumption is that the pronouns used in these emails can often be resolved to some element from this data source. In the case where this data is organized into a database, this entity from the data source can either be a word used in the schema (relation or column names) or tuple values contained within these tables.

Using this assumption, a domain-specific algorithm was created that can be easily ported to different domains and different database schema for the same domain, as long as the domain can be described using a backend database and the schema for this database is available. The first step of the algorithm involves acquiring the domain knowledge from the database schema and tuples. It is assumed that the section of the database that will be used for responding to the emails has already been identified (since this has to be done anyway in an e-mail response system). For this domain, the assumed relevant section of the (invented) corporate database schema can be found in figure 4 below.

(Column names marked with * are keys)

Relation: ORDER

order_id* package_id+ customer_id+

+package_id is a foreign key on the SHIPMENT relation
+customer_id is a foreign key on the CUSTOMER relation

Relation: PRODUCT

product_id* product_category product_name num_in_stock

Relation: ORDER_PRODUCT

order_id* product_id+ amount+

+order_id is a foreign key on the ORDER relation
+product_id is a foreign key on the PRODUCT relation

Relation: SHIPMENT

package_id ship_date exp_arrival_date

Figure 4: Relevant Database Schema for E-Mail Query Domain

This schema contains four relations, describing the contents, shipment, and customer information of online orders. Once the relevant section of the database has been identified, words used in the schema, column, and tuple values are extracted and converted to MRS relation names for noun phrases. For example, three out of the four relation names in this database cross-section can successfully be converted to noun relation names: ORDER can be converted to _order_n_rel, PRODUCT to _product_rel, and SHIPMENT to _shipment_rel.

For this schema, it is difficult to convert the column names directly into MRS relation names because most of the column names contain more than one word separated by underscores, or have the word name followed by “_id”. In general, an automatic conversion would require that the database schema for all databases would have column names that follow a naming convention; a phenomenon that does not naturally occur. If the desire is to automatically acquire domain knowledge with no human intervention, then the process of attempting to generate MRS relation names from column names should probably be skipped. However, with minimal human intervention, column names can be anglicized and even a thesaurus could be used to help find potential synonyms of these names. Finally, the unique contents of those columns that contain STRING valued single-word attributes (such as product_category in figure 4) are also converted into relations.

Using the assumed contents of the database along with the schema from figure 4, the following MRS relations are extracted from this domain:

_order_n_rel
_shipment_rel → **from database relation names**
_product_rel
--
_package_n_rel → **from database column names**
--
_laptop_rel
_computer_rel
_tv_rel
_camcorder_rel → **from database contents**
_phone_rel
_camera_rel
_calculator_rel

Figure 5: Relations extracted from e-mail query database schema and contents

The pronoun resolution algorithm then uses object "x" variables to create potential antecedent lists as in the non-domain-specific algorithm, and associates these variables with MRS relations as before. However, instead of using salience preferences to choose between potential antecedents (after this list has been filtered for agreement constraints), the algorithm checks the relations associated with each variable in the potential antecedent list, and if the relation had been derived from a relation name it receives a score of 45 points; if it had been derived from a column name it receives a score of 40 points; and if it had been derived from the contents of the database it receives a score of 35 points. Otherwise it receives a score of 0 points. If no element in the antecedent list receives any points, then the pronoun is marked as unresolvable given the input text. Otherwise, the antecedent with the highest score (where the rare case of a tie is resolved by taking the most recent antecedent) is coindexed with the pronoun. Clearly this algorithm relies heavily on the fact that the focus of an e-mail query in this domain will only be on the data source; pronouns will only be resolved to an antecedent found in the input text if one of the above listed 11 relations are found in the query and are associated with an extracted object variable.

8. Evaluation

In order to evaluate the pronoun resolution algorithms, pronouns were marked by hand and coindexed with their noun phrase antecedent (if one existed; otherwise it was given an index of -2) in the training and test corpus. Rather than mark every noun phrase by hand, the baseline algorithm was run or the corpuses to get a preliminary marking of noun phrases. Despite the mistaken marking of many noun phrases (since the baseline algorithm did not use a parser) the noun phrase marks were only altered if they affected the resolution of a pronoun. For example: one e-mail from the training data began with the sentence:

(k) I^{0} have ordered a digital camera last month^{1}.

The baseline algorithm marked the noun phrases as above, incorrectly marking the temporal adjunct *last month* as the direct object, and not marking the noun phrase *digital camera* with an index. But since this e-mail contained no third person pronouns, the incorrect noun phrase marking was irrelevant and was not changed for the marked data. If a sentence contained a third person pronoun, this pronoun was given a noun phrase index, and then was coindexed with its noun phrase antecedent after a '#' character. For example:

(ℓ) I^{0}ordered a digital camera^{1} on the web^{2} two weeks^{3} ago.
I^{4} have been waiting for it$^{5\#1}$ to arrive since then.

The pronoun *it* is given its own index (5) in case it will need to serve as the antecedent for another pronoun, while at the same is coindexed with the noun phrase *digital camera*.

In the cases where there exists more than one possibility for an antecedent, both possibilities are marked, for example:

(m) The laptop^{1} I^{2} ordered is model^{3} # 784503^{4}.
The order^{5} # is 43186^{6}.
Where is it$^{7\#1\#5}$?

The pronoun *it* can potentially refer to the *laptop* or the *order*, so both possibilities are marked.

A major problem with evaluating the correct coindexing of pronouns with their antecedents are that different algorithms will not index the same number of noun phrases and often different words will be marked. For this reason, it is not possible to directly check the numerical values of the indices with the hand marked indices to check for a correct reference. Nor can one resolve the index values to their corresponding text tokens in order to evaluate if a pronoun was correctly marked. Such an algorithm would mark the following example as correct:

(n) My old laptop^{0} is broken.

I^{1}'d like to buy a new laptop^{2}.
If I^{3} ordered it$^{4\#0}$ now, how long is shipping?

In this example, the antecedent for the pronoun *it* is the noun phrase *laptop*, but the only correct instance of this antecedent would be the noun phrase in the second sentence (not the *laptop* in the first sentence). But an evaluation algorithm that associated index numbers with a token in the input text and then used this token as the reference for the pronoun (i.e. If I^{3} ordered it$^{4\#\mathrm{laptop}}$ now, how long is shipping?) example (n) would be incorrectly scored as a correct reference. So instead, the evaluation algorithm converts all indices to token numbers and uses these token numbers to evaluate pronoun references. So for example, it would convert example (n) to:

(n) My old laptop^{2} is broken.
I^{6}'d like to buy a new laptop^{13}.
If I^{16} ordered it$^{18\#2}$ now, how long is shipping?

The evaluation algorithm performs an identical conversion on the reference text; where the pronoun *it* is marked as 18#13 and will (correctly) mark this example as an incorrect reference.

The problem with evaluating pronoun reference by converting all indices to token numbers is that all algorithms (that are being evaluated using the same reference markings) have to tokenize the input text the same way. For this reason, an (almost) identical tokenizer was used to parse and output the e-mail text for the extension algorithms as were used by Siddharthan (2003) in the baseline algorithm. However, this did not solve the tokenization problem entirely as the LKB parser that produced the MRS representations for the extension algorithms tokenized input text differently. For this reason, a procedure was written that correlated the LKB parser tokens with the Siddharthan (2003) tokens and could then convert the CFROM and CTO values (which describe LKB tokens) to the corresponding Siddharthan (2003) tokens.

Once the output of the baseline and extension algorithms are able to be compared with the reference output, two evaluation metrics are made (taken from Siddharthan, 2003). The differences between these two metrics occur when a pronoun is resolved to having another pronoun as its antecedent. For example:

(o) I bought a laptop^{0} with a free external keyboard^{1}. But I^{2} think it$^{3\#1}$'s broken because I^{4} can't turn it$^{5\#3}$ on.

In this example, the pronoun *it*, marked with index 3, is incorrectly resolved to refer to the same referent as the antecedent *keyboard*. This pronoun will be marked as incorrect. However, the second instance of the pronoun *it*, marked with index 5, is correctly coindexed with the first instance of the pronoun (index 3). But since the first pronoun was incorrectly marked, the second pronoun can be perceived to be correctly or incorrectly marked depending on whether or not the chain of references is traced all the way backwards to the first instance of a non-pronominal entity. The Eval_Absolute metric performs this trace of the chain of references to determine if the absolute reference of the pronoun is correct. The Eval_Salience metric does not perform this trace, and will mark as correct pronouns that are correctly coindexed with their pronoun antecedents. So Eval_Absolute would mark example (o) as incorrect and Eval_Salience would mark this example as correct. Siddharthan, 2003, uses Eval_Salience in the training phrase to determine the value of the parameters to the algorithm so that those errors that propagate a long way do not receive preferential treatment for being fixed. However, in the end, Eval_Absolute is the metric of the true success of pronoun resolution.

9. Results

Figure 6 shows both the Eval_Salience and Eval_Absolute precision of the third person pronoun resolution algorithms on the training data for each of the four algorithms.

Figure 6: Results

	Eval_Salience	Eval_Absolute
Baseline Algorithm (Siddharthan 2003)	70.22%	64.61%
MRS-Derived Syntactical Knowledge Algorithm	78.21%	74.86%
Domain-Specific Algorithm	81.01%	81.01%
Combined Syntactic Knowledge and Domain-Specific Alg.	91.62%	91.62%

The results for the test data are not yet available (but will be for the final draft).

10. Analysis

The baseline algorithm does notably worse in this domain than on the corpus used in Siddharthan, 2003. The Eval_Salience on the Siddharthan, 2003 test data was found to be 85% and the Eval_Absolute was found to be 79%, 18% higher than the results reported here on the e-mail query corpus. The likely reason for this is that almost all of the third person pronouns in this corpus (along with their potential antecedents) are inanimate. So the baseline algorithm's inference mechanisms to deduce the gender and animacy of pronouns and their antecedents will not yield any new information. Because of this, the algorithm will not be able to successfully filter most antecedent lists using these agreement constraints (since almost everything in third person is inanimate in this domain). This explains why the results reported above are similar to the results reported in Siddharthan, 2003, when the gender and animacy inference mechanism is turned off (which yielded precision of 70% and 60% for Eval_Salience and Eval_Absolute respectively).

Both extension algorithms perform better than the baseline algorithm. The first extension algorithm, which used non-domain-specific syntactic knowledge, was expected to perform better than the baseline algorithm, since this algorithm had more detailed syntactic information that had been derived from the MRS feature structures. An example where this more accurate syntactic information helps identify the correct antecedent is given in the following example:

(p) I^{0} have a laptop^{1} on backorder^{2} that^{3} I^{4}'m waiting for .
Can you^{5} let me^{6} know what is going on with it$^{7\#2}$?

The baseline algorithm marks this email as shown, with the pronoun *it* being incorrectly resolved to the antecedent *backorder* rather than *laptop*. In this example, the baseline algorithm assigns the noun phrase *laptop on backorder* as the direct object of the sentence. However, it assumes the last noun in the noun phrase is the head noun, so it assigns the direct object salience factor to *backorder*. This antecedent received the highest salience factor of the potential antecedents (after filtering for just third person singular antecedents) and is chosen as the antecedent for the pronoun *it*. In contrast, the extension algorithm that has access to the MRS parse of this sentence does not make the same mistake, since the direct object (ARG2) of the index relation (*_have_rel*) is correctly set equal to the object corresponding to the head noun of the noun phrase *laptop on backorder* (which is the ARG0 of *_laptop_rel*).

Improved noun phrase indexing is another reason why the extension algorithm performs better than the baseline. The baseline algorithm frequently includes temporal adjuncts in the same phrase as the direct object of a sentence which results in the direct object not being marked. For example, the baseline algorithm with incorrectly mark the following e-mail:

(q) I^{0} am extremely upset and I^{1} can't believe your^{2} service^{3} .
I^{4} bought a computer last month^{5} .
You^{6} told me^{7} I^{8} should receive it$^{9\#3}$ last week^{10} .

Since the temporal adjunct *last month* was included in the same phrase as the direct object of bought (*a computer*), this object isn't marked and is not included in the potential list of antecedents for the pronoun *it*, and thus this pronoun cannot possibly be correctly resolved. In contrast, the MRS representation of the sentence *I bought a computer last month* contains two separate objects for *computer* and *last month*.

However, the extension algorithm does not always perform better than the baseline algorithm. Bad parses can lead to resolution errors. Take, for example, the following e-mail:

(r) I^{0} ordered my^{1} package^{2} #^{3} 78465^{4} last week^{5} .
When will I^{6} receive it$^{7\#3}$?

The incorrect resolution of the pronoun *it* to the # sign in the extension algorithm is due to the way that this sentence was parsed. Rather than reading *# 78465* as an appositive to the noun *package*, the parser read *package #* as a compound noun (with # as the head noun). In other words, the parser read the sentence as the *package number* being ordered rather than the package itself. The baseline algorithm does not mark the character '#' as a noun phrase, and correctly resolves the pronoun.

Overall, it is encouraging that it was found that increased syntactic knowledge does indeed lead to increased precision in pronoun resolution. However, the fact that the domain-specific algorithm performs better than either of these algorithms is quite surprising. The domain-specific algorithm is much simpler than the algorithm using syntactic knowledge; it will only resolve a pronoun to one of eleven possible antecedent relations (for this domain). Otherwise it gives up and marks the pronoun as unresolvable. The fact that this simple algorithm performs so well implies that the fundamental assumption behind this algorithm, that the focus (and thus the likely antecedent of a pronoun) is often on an element from the data source, is usually correct. It should be noted that an additional reason for the success of this algorithm is that 8% of the pronouns in the data set are marked as pleonastic or unresolvable in the reference text (and so have an index of -2), so by giving up easily, this algorithm will get most of these references correct.

The most encouraging result from this project is that when the non-domain-specific and domain specific algorithms are combined, the pronoun resolution improves to a precision above 90%. So using domain knowledge can aid a syntactical information based algorithm (or vice versa: using syntactical preferences can aid in domain-specific pronoun resolution). The algorithms were combined by simply taking a linear combination of the two scores (the salience and domain specific scores), giving them equal weight. So if a potential antecedent received a salience factor of 50 and a domain-specific factor of 40, then its resulting score in the combined algorithm will be 45. This is a crude way to combine the algorithms; perhaps it might be better to directly integrate the domain and syntactic knowledge by specifically weighing the relations extracted from the database backend depending on their grammatical function in the e-mail messages. But in the training corpus, the crude linear combination approach to combining the two algorithms was effective enough so that only 15 pronouns in the corpus were resolved incorrectly. Of these 15 pronouns, no pronouns were incorrectly marked due to the mechanism for combining the two extension algorithms. 6 of these 15 errors could be fixed with an improved filter for pleonastic pronouns; for example:

(s) I^{0}'d like to confirm that my^{1} laptop^{2} will arrive on 10-05 as it$^{3\#2}$ says in my^{4} order^{5} confirmation^{6}.

The remaining errors are generally due to confusing pronouns to resolve. For instance, the pronoun *them* is plural, but in the following example it should be resolved to the singular noun *customer service*:

(t) Please let me^{0} know how^{1} to reach customer^{2} service^{3} so that I^{4} can call them$^{5\#-2}$ to discuss questions^{6} I^{7} have about my^{8} order^{9}.

But *customer service* is filtered from the potential antecedent list for the pronoun *them* because of the number agreement constraint.

11. Conclusion

Three conclusions can be made from these results.

* First, detailed syntactic knowledge does indeed increase pronominal anaphora resolution precision for non-domain-specific algorithms. The results showed that the precision (measured using the Eval_Absolute metric) increased by a factor of 16% with increased syntactic knowledge.

* Second, a simple domain-specific algorithm where domain knowledge is acquired from a database backend to the e-mail response system performs better than the non-domain-specific algorithms relying on syntactical knowledge.

*Third, combining a domain-specific algorithm with a domain-independent algorithm further increases pronoun resolution accuracy. For the e-mail query corpus used for this project, over 90% of pronouns were resolved accurately using this combined technique.

12. Bibliography

Brennan, S.E., Friedman, M. W., and Polard, C. (1987). A centering approach to pronouns. In *ACL-97*, Stanford, CA, pp. 155-162. ACL.

Hobbs, J.R. (1978). Resolving Pronoun References. *Lingua*, 44, 311-338.

Jurafsky, D. and Martin, J. (2000). *Speech and language processing*. Prentice Hall, pp. 669-694.

Kennedy, C. and Boguraev, B. (1996). Anaphora for everyoneL Pronomial anaphora resolution without a parser. In *COLING-96*, Copenhagen, pp. 113-118.

Lappin, S. and Leass, H. (1994). An Algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4), 535-561.

Siddharthan, A. (2003). Resolving Pronouns Robustly: Plumbing the Depths of Shallowness. In Proceedings of the Workship on Computational Treatments of Anaphora, EACL-2003, pp 7-14.

Appendix A: MRS For: “The first one arrived yesterday but I’m still waiting for the second one.”

```
[ LTOP: h1
 INDEX: e2 [ CONJ_EVENT
             DIVISIBLE:  BOOL
             E.TENSE:  TENSE
             E.ASPECT:  ASPECT
             E.MOOD:  MOOD ]
 RELS: <
         [ prpstn_m_rel
           LBL: h3
           CFROM: STRING
           CTO: STRING
           MARG: h4 ]
         [ _def_q_rel
           LBL: h5
           CFROM: 0
           CTO: 1
           ARG0: x7 [ REF-IND
                      PNG.GEN:  REAL_GENDER
                      PNG.PN:  3SG*
                      DIVISIBLE:  - ]
           RSTR: h8
           BODY: h6 ]
         [ ord_rel
           LBL: h9
           CFROM: 1
           CTO: 2
           ARG0: e10 [ EVENT_OR_INDEX
                      DIVISIBLE:  BOOL ]
           ARG1: x7
           CARG: "1" ]
         [ _one_n_rel
           LBL: h9
           CFROM: 2
           CTO: 3
           ARG0: x7 ]
         [ _arrive_rel
           LBL: h11
           CFROM: 3
           CTO: 4
           ARG0: e12 [ EVENT
                      DIVISIBLE:  BOOL
                      E.TENSE:  PRES+PAST
                      E.ASPECT:  NOASP+PROGR
                      E.MOOD:  INDICATIVE* ]
           ARG1: x7 ]
         [ unspec_loc_rel
           LBL: h11
           CFROM: STRING
           CTO: STRING
           ARG0: e12
           ARG1: e12
           ARG2: x13 [ REF-IND
                      DIVISIBLE:  BOOL
                      PNG.GEN:  REAL_GENDER
                      PNG.PN:  3SG* ] ]
         [ time_rel
           LBL: h14
           CFROM: 4
           CTO: 5
           ARG0: x13 ]
         [ def_q_rel
           LBL: h15
           CFROM: STRING
           CTO: STRING
           ARG0: x13
           RSTR: h17
           BODY: h16 ]
         [ _yesterday_rel
           LBL: h14
           CFROM: STRING
           CTO: STRING
           ARG1: x13 ]
         [ _but_rel
           LBL: h1
           CFROM: 5
           CTO: 6
           C-ARG: e2
           L-HNDL: h3
           L-INDEX: v18 [ NON_EXPL
                      DIVISIBLE:  BOOL ]
           R-HNDL: h19
           R-INDEX: v20 [ NON_EXPL
                      DIVISIBLE:  BOOL ] ]
```

```
[ prpstn_m_rel
            LBL: h19
            CFROM: STRING
            CTO: STRING
            MARG: h21 ]
          [ pron_rel
            LBL: h22
            CFROM: 6
            CTO: 7
            ARG0: x23 [ FULL_REF-IND
                         DIVISIBLE:  -
                         PNG.PN:  1SG
                         PNG.GEN:  REAL_GENDER
                         PRONTYPE:  STD_PRON ] ]
          [ pronoun_q_rel
            LBL: h24
            CFROM: STRING
            CTO: STRING
            ARG0: x23
            RSTR: h25
            BODY: h26 ]
          [ _still_rel
            LBL: h27
            CFROM: 8
            CTO: 9
            ARG1: e28 [ EVENT
                         E.TENSE:  PRES+PAST
                         E.ASPECT:  NOASP+PROGR
                         E.MOOD:  INDICATIVE*
                         DIVISIBLE:  BOOL ] ]
          [ _wait_v_rel
            LBL: h27
            CFROM: 9
            CTO: 10
            ARG0: e28
            ARG1: x23
            ARG2: v29 [ NON_EXPL-IND
                         PNG.GEN:  REAL_GENDER
                         PNG.PN:  PERNUM
                         DIVISIBLE:  BOOL ] ]
          [ _for_rel
            LBL: h27
            CFROM: 10
            CTO: 11
            ARG0: e28
            ARG1: e28
            ARG2: x30 [ REF-IND
                         DIVISIBLE:  -
                         PNG.GEN:  REAL_GENDER
                         PNG.PN:  3SG* ] ]
          [ _def_q_rel
            LBL: h31
            CFROM: 11
            CTO: 12
            ARG0: x30
            RSTR: h33
            BODY: h32 ]
          [ ord_rel
            LBL: h34
            CFROM: 12
            CTO: 13
            ARG0: e35 [ EVENT_OR_INDEX
                         DIVISIBLE:  BOOL ]
            ARG1: x30
            CARG: "2" ]
          [ _one_n_rel
            LBL: h34
            CFROM: 13
            CTO: 14
            ARG0: x30 ] >
  HCONS: <  h4 QEQ h11
            h8 QEQ h9
            h17 QEQ h14
            h21 QEQ h27
            h25 QEQ h22
            h33 QEQ h34 > ]
```