Anaphora Resolution in ExtrAns

Diego Mollá, Rolf Schwitter
Centre for Language Technology,
Macquarie University,
Sydney, Australia,
{diego,rolfs}@ics.mq.edu.au

Fabio Rinaldi, James Dowdall, Michael Hess
Institute of Computational Linguistics,
University of Zurich,
Zurich, Switzerland
{rinaldi,dowdall,hess}@cl.unizh.ch

Abstract

The true power of anaphora resolution algorithms can only be gauged when embedded into specific Natural Language Processing (NLP) applications. In this paper we describe the anaphora resolution module from ExtrAns, an answer extraction system. The anaphora resolution module is based on Lappin and Leass' original algorithm, which used McCord's Slot Grammar as the inherent parser. We report how to port Lappin and Leass' algorithm to Link Grammar, a freely available dependency-based parsing system that is used in a range of NLP applications. Finally, we report on how the equivalence classes that result from the anaphora resolution algorithm are incorporated into the logical forms used by ExtrAns.

1 Introduction

Research in anaphora resolution has been very intense during several periods in the past and present. Several anaphora resolution modules have been implemented in the past (Mitkov et al., 2002; Grosz et al., 1995; Kennedy and Boguraev, 1996), and a number of NLP applications use anaphora resolution components. In particular, several question-answering systems have implemented anaphora resolution in several ways (Jong-Hoon et al., 2001; Vicedo and Ferrández, 2000).

In this paper we consider the role of Anaphora Resolution in ExtrAns, a Question Answering system targeted specifically at technical documentation. After an initial application to the Unix man-pages (Mollá et al., 2000), ExtrAns was used in the Aircraft Maintenance Manual (AMM) of the Airbus A320 (Rinaldi et al., 2002), and currently we are targeting the Linux HowTos. ExtrAns translates documents and questions into a flat semantic representation using a comprehensive linguistic analysis. The system resolves pronominal references, disambiguates ambiguous structures, and includes modules capable of dealing with peculiarities of technical terminology (Rinaldi et al., 2003; Dowdall et al., 2002). ExtrAns derives the answers to questions by logical proofs from the document collection. A schematic representation of the architecture of the system can be seen in Figure 1.

Document sentences and questions are syntactically processed by Link Grammar, a parsing system that consists of a robust dependency-based parser and a wide-coverage grammar for English (Sleator and Temperley, 1993). In the current version of ExtrAns, anaphora resolution is restricted exclusively to pronominal cases since it is less clear how the explicit resolution of definite noun phrases and especially associative references might improve the answer extraction process without making use of complex external resources (e.g. domain/world knowledge, Ontologies).

The resolution algorithm we are going to present in this paper is an adaptation of a purely syntactic approach. The theory behind the anaphora resolution module of ExtrAns is based on (Lappin and Leass, 1994), but it has been fine-tuned in several ways for the answer extraction task. Its major advantage

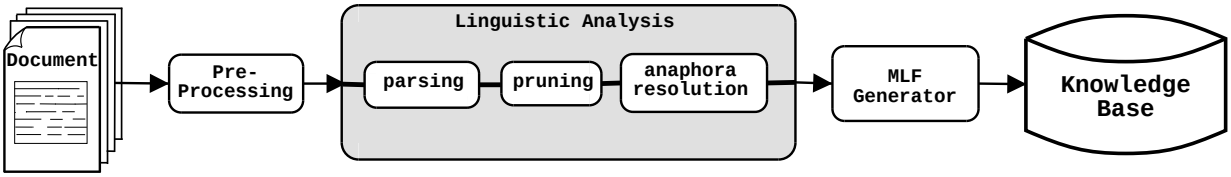


Figure 1: Schematic view of the Architecture of ExtrAns

against other algorithms is that it can be adapted to use the resources that are available in a state-of-the-art answer-extraction system such as ExtrAns. In particular it uses syntactic information, such as that produced by Link Grammar, rather than resorting to parser-free approaches like (Kennedy and Boguraev, 1996). We will see in the following sections that the type of syntactic information required for the anaphora resolution algorithm is different from the one provided by Link Grammar, but it is possible to largely re-create the necessary information.

Another advantage of the use of Lappin and Leass' algorithm is that it does not use semantic information, in contrast with to (van der Sandt and Geurts, 1991; Pinkal, 1991), nor real-world knowledge, in contrast with (Hobbs, 1978). Nor does it model intentional or global discourse structure, in contrast with (Grosz, 1981). Thus, the result is computed in relatively short time and uses less resources.

Lappin and Leass' anaphora resolution model has the following components (Lappin and Leass, 1994, 536):

- An intrasentential syntactic filter for ruling out anaphoric dependence of a pronoun on an NP on syntactic grounds.
- A morphological filter for ruling out anaphoric dependence of a pronoun on an NP due to non-agreement of person, number, or gender features.
- A procedure for identifying pleonastic (semantically empty) pronouns.
- An anaphor binding algorithm for identifying the possible antecedent binder of a reflexive or reciprocal pronoun within the same sentence.
- A procedure for assigning values to several salience parameters for an NP.
- A procedure for identifying anaphorically linked NPs as an equivalence class.
- A decision procedure for selecting the preferred element of a list of antecedent candidates for a pronoun.

Each one of these steps has been adapted to ExtrAns, as we will discuss in the following sections.

2 Anaphora Resolution in ExtrAns

2.1 Emulation of the Slot Grammar

Lappin and Leass' algorithm relies heavily on the output of the parser, a Prolog clausal implementation of Slot Grammar (McCord et al., 1992). The resulting syntactic analysis includes the head-argument and head-adjunct relations of the phrase structure that the Slot Grammar assigns to the sentence or phrase. These relations (also called **slots**) include "subject", "agent", "object", "indirect object", and "prepositional object".

Since Slot Grammar is dependency-based, it is possible to approximate its behaviour by means of the dependency structures returned by Link Grammar. However, Link Grammar does not show the direction of dependencies explicitly. This information is easily recovered by examining the link types and occasionally some specific local arrangements of the link structures, and it has been added in a post-processing module of ExtrAns' parser (Mollá et al., 2000).

Given a dependency structure such as the one provided by ExtrAns' parser module, we can explore the labels of the links to compute the relations listed above: [1]

[1] It is possible that the algorithm tries to find the slot of a particular word several times (if the word is a candidate for coreference for several pronouns, for example). To speed up processing, the algorithm first checks if the information has already been computed. If not, the slot is computed and the result is stored in the noun-related data structure and returned.

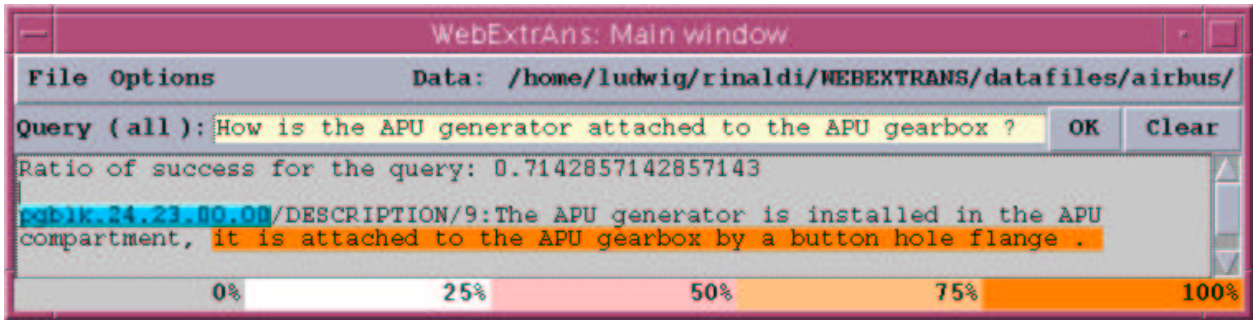


Figure 2: An example of Anaphora Resolution in ExtrAns

Subject (`subj`). Check if the link is `S`[2] [subject] or `SI` [inverted subject]. Alternatively, check if the word is actually the subject of a relative clause (links `B` [external subject/object] and `RS` [subject relative pronoun] in a specific pattern).

Agent (`agent`). Check if the word is the main noun of a PP headed by *by*, and the sentence is passive.

Direct (`obj`) and indirect object (`iobj`). This is decided by the link `O` [object]. The object closest to the verb is the direct object, and the rest are indirect objects.

Prepositional object (`pobj`). The object of a PP can be found by checking if the link is `J` [object of a PP] or `U` [idiomatic noun].

2.2 The syntactic filter on pronoun-NP coreference

The filter uses a set of syntactic rules to prune and remove anaphoric dependencies which are syntactically impossible. Lapping and Leass use the following terminology:

Immediate containment. A phrase P is immediately contained in a phrase Q iff P is either an argument or an adjunct of Q. In terms of ExtrAns, P is directly dependent on Q. Note that the algorithm must also check if P or Q is a member of a coordination.

Containment. A phrase P is contained in a phrase Q iff:

1. P is immediately contained in Q, or
2. P is immediately contained in some phrase R, and R is contained in Q.

Argument domain. P is in the argument domain of a phrase N iff P and N are both arguments of the same head. Since Link Grammar does not differentiate between arguments and adjuncts, the rule is implemented so that P and N are both immediately contained in the same head.

Adjunct domain. P is in the adjunct domain of N iff N is an argument of a head H, P is the object of a preposition $PREP$, and $PREP$ is an adjunct of H. In terms of Link Grammar, this means that N and $PREP$ are both immediately contained in H.

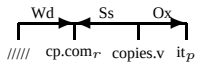
NP domain. P is in the NP domain of N iff N is the determiner of a noun Q and:

1. P is an argument of Q, or
2. P is the object of a preposition $PREP$ and $PREP$ is an adjunct of Q.

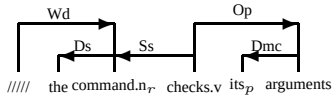
In terms of Link Grammar, the definitions of "argument" and "adjunct" merely mean immediate containment.

Now, we can say that a pronoun P cannot corefer with a non-reflexive, non-reciprocal noun phrase R if any of the following conditions hold:

1. P is in the argument domain of R:

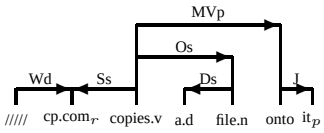


This rule does not apply when P is a determiner:

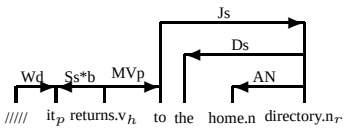


[2] See http://www.link.cs.cmu.edu/link/ for a summary of the link types used by Link Grammar.

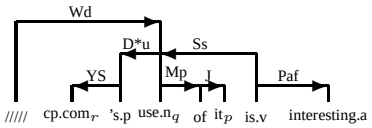
2. P is in the adjunct domain of R:



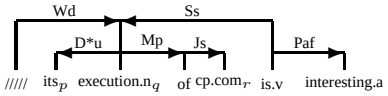
3. P is an argument of a head H, R is not a pronoun, and R is contained in H:



4. P is in the NP domain of R:



5. P is the determiner of a noun Q, and R is contained in Q, or R is Q:



   Note that the example above is ruled out by rule 3 because of our lax definition of argument.

Apart from the original rules, we have added the following rules. A pronoun P cannot corefer with a noun R if:

1. P is R. This rule is pretty obvious.

2. P is contained in R. This rule handles some of the cases that were not ruled out by the rules above because of our particular definition of argument and adjunct dependency.

2.3 Tests for agreement

In the original algorithm, the agreement features of an NP are number, person, and gender. In the technical domain of ExtrAns, the pronoun resolution has been implemented only for neuter pronouns, and therefore gender and person are irrelevant. Thus, we use the number only.

To compute the number agreement of a word, we use the label of the link of the word to its head. The algorithm checks whether the link has a suffix `s` or `m` (indicating singular), or whether the suffix is `p` (indicating plural). [3]

2.4 Identifying pleonastic pronouns

Pleonastic pronouns are pronouns that do not carry any meaning, like in the phrase *it is likely that*. To identify them, we use Link Grammar's link labels. The following link labels indicate the use of a pleonastic pronoun:

SF is a special connector used to connect "filler" subjects like *it* and *there* to finite verbs: *THERE IS a problem, IT IS likely that ....*

SFI connects "filler" subjects like *it* and *there* to verbs in cases with subject-verb inversion: *IS THERE a problem?, IS IT likely that ...?*

One stage in the process of anaphora resolution consists in identifying all the nouns in the sentence, including pronouns. The algorithm intentionally ignores both `SF` and `SFI`, as a result pleonastic pronouns are not selected.

2.5 The treatment of lexical anaphors

According to Lappin and Leass, a lexical anaphor is either a reciprocal or a reflexive pronoun. The syntactic rules of coreference of lexical anaphors do not check if the coreferent nouns are incompatible. Instead, the rules check if the noun can corefer with the lexical anaphor. Two assumptions are that the noun and the anaphor appear in the same sentence, and the noun must appear before the anaphor.

To compute the possibility of coreference, it is necessary to rank the slot information of the noun and the anaphor, as follows:

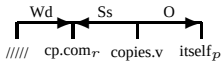
$$\text{subj} > \text{agent} > \text{obj} > (\text{iobj—pobj})$$

In other words, a subject occupies a higher argument slot than an object, and the slot position of an indirect object and a prepositional object are equivalent.

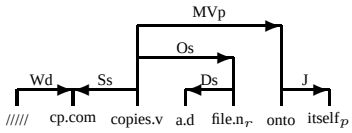
A noun R is a possible antecedent binder for a lexical anaphor P iff R and P do not have incompatible agreement features, and one of the following conditions holds:

[3] In some cases, the link label does not provide the information. When this happens, the number of the word is unspecified (using an unbound Prolog variable).

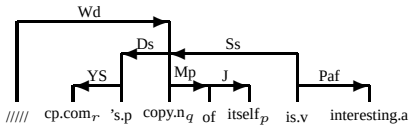
1. P is in the argument domain of R, and R fills a higher argument slot than P:



2. P is in the adjunct domain of R:



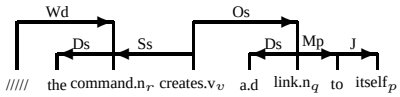
3. P is in the NP domain of R:



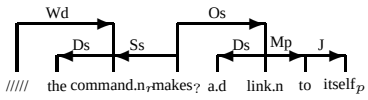
4. R is an argument of a verb V, and there is an NP Q in the argument domain or in the adjunct domain of R such that:

   (a) P is an argument of Q, or

   (b) P is an argument of a preposition $PREP$ and $PREP$ is an adjunct of Q:



   In this rule, we need to determine that V is a verb. We do that by checking the tag assigned by the parser ('.v' in the examples above). If the parser gives a wrong tag or no tag at all, the verb is not recognised. For that reason, the anaphora in the following sentence is not recognised:



5. P is a determiner of a noun Q, and:

   (a) Q is in the argument domain of R and R fills a higher argument slot than Q, or

   (b) Q is in the adjunct domain of R

2.6 The salience parameters

Salience is a measure that indicates how likely a particular noun is to corefer with a specific pronoun or lexical anaphor. The salience of a noun is a combination of several factors. Lappin and Leass' salience factors are based on (Alshawi, 1987), though the actual factors and values are more specific to the task of pronominal anaphora resolution.

Salience factors can be classified into two types: independent factors and dependent factors. The salience of a noun will be the sum of the weights of all the salience factors that apply to it.

2.7 Independent salience factors

Independent salience factors are those that do not depend on the syntactic relation between the pronoun and the noun. The following salience factors used by Lappin and Leass have been implemented: [4]

Subject. The noun is the head of a subject.

Agent. The noun is an agent in a passive sentence.

Existential emphasis. The noun is in an existential construction, such as in "*there are only* a few restrictions *on LQL query construction for WordSmith*".[5]

Accusative. The noun is the object.

Indirect object. The noun fills indirect object slot.

Oblique complement. The noun fills the prepositional object slot.

Head noun. The noun is not contained in another noun, using the Slot Grammar notion of "containment within a phrase" (it is either an argument or an adjunct of a noun).

Non-adverbial. The noun is not contained in an adverbial PP demarcated by a separator. [6]

2.8 Dependent salience factors

Dependent salience factors may or may not apply to a specific noun, or they may apply with a different value, depending on the syntactic relation with the pronoun. The factors implemented in ExtrAns are:

Cataphora penalty. Cataphora (when the pronoun appears in the sentence *before* the noun) is to be discouraged, by adding a rather large negative value to the accumulated score.

[4] Table 1 lists the weight values selected for the independent salience factors.

[5] This is checked by exploring the label of the link that connects the word to its head: it should match `O?t`, where `?` matches any link label suffix.

[6] ExtrAns uses the link labels `J` and `U` to determine if a noun is in a PP, and `Xc` and `Xd` to determine the existence of a separator.

Factor type	Weight
Subject	80
Agent	80
Existential emphasis	70
Accusative	50
Indirect object	40
Oblique complement	40
Head noun	80
Non-adverbial	50

Table 1: Weight values for independent salience factors

Parallel roles reward. If both the noun and the pronoun fill the same slot, the probability of being coreferent is higher.

Recency reward. Intrasentential coreference is to be encouraged.

The actual values of the dependent salience factors are listed in Table 2. The values are the same as

Factor type	Weight
Cataphora	-275
Parallel roles	35
Recency	100

Table 2: Weight values for dependent salience factors

in the original algorithm (Lappin and Leass, 1994) except for the case of cataphora, where we decided to increase the penalty by 100 units (up from -175).

2.9 Equivalence classes

Coreference between the pronoun and the noun is signaled by classifying both words as belonging to the same equivalence class. An equivalence class represents the set of the words that point to the same instance in the world of the application domain. [7]

[7] Membership of the same equivalence class is expressed in the original algorithm by means of the predicate `coref(u,y)`, which is inserted in the logical form of the sentence. For ExtrAns, however, we decided not to use additional predicates to express coreference chains. Instead, variable substitution is made. Thus, if `u` corefers with `y`, then all the predicates in the sentence that have `y` as an argument will replace `y` with `u`.

Lappin & Leass' algorithm assigns a salience to the equivalence classes. How they do it, though, is not clear. In ExtrAns, the equivalence class salience is that of the class representative (for the time being, the most recent element of the equivalence class). An interesting possibility worth considering is the computation of the equivalence class salience based on when the last new word was introduced to the equivalence class. In other words, we can update the equivalence class salience on the basis of the current focus.

2.10 Decision procedure

With the modifications detailed in the previous sections, the implementation of Lappin and Leass' anaphora resolution algorithm is straightforward:

1. Create the list of antecedents:
   (a) Create an initial list of IDs for all the NPs in the sentence.
   (b) Compute the independent salience factors of every ID.
   (c) Group the antecedents in equivalence classes according to their coreference (for the obvious cases, like names or command arguments).
2. For every pronoun in the sentence:
   (a) Compute the list of possible antecedent candidates (the most recent of each equivalence class) $\mathcal{A}$.
   (b) Compute the list of incompatible references $\mathcal{B}$, according to syntactic and agreement grounds.
   (c) Compute the list of possible references of reflexive pronouns $\mathcal{C}$, according to syntactic and agreement grounds.
   (d) The final list of candidates is $(\mathcal{A} - \mathcal{B}) \cap \mathcal{C}$.
   (e) Compute the dependent salience factors of the final candidates.
   (f) Select the candidate with higher salience (the sum of independent and dependent salience factors). If there are several with the same salience, choose the candidate closest to the anaphor.
   (g) Add the pronoun to the equivalence class of the selected candidate.

3 Evaluation

In order to evaluate the effectiveness of the reference resolution algorithm we selected arbitrarily 50 sentences of the Airplane Maintenance Manual corpus which contained at least one pronoun. The total number of pronouns to be resolved was 60. We then manually examined the logical forms generated by ExtrAns to verify if the pronouns had been resolved correctly: we found that it was only 26 (43%) of the pronouns. However, the reference resolution algorithm is only responsible for some of the failures, other processing factors produced the rest. For instance, the most frequent cause of an error is a wrong parse (in a few cases the parse failed altogether). We therefore re-examined the cases were the pronoun was not resolved correctly and could filter out 18 sentences were the failure was clearly caused by reasons external to the reference resolution algorithm.

This left us with 32 sentences and 36 pronouns, of which 26 were resolved correctly. After manual examination 3 cases were then excluded because they were judged genuinely ambiguous, like the following: "Do not take the repair kit to the repair area until it is ready to use".

So we have 26 pronouns correctly resolved out of 33, which results in an accuracy of 79%. This is lower than the result reported by (Lappin and Leass, 1994) of 86%. But, of course, our data is too small to be representative.

We then examined in detail what happened for each pronoun type:

It: Out of 28 anaphoric *it*, 21 were correctly solved. Thus, the result is of 75% correct resolutions.

Its: Of 3 cases, only 1 was correctly solved.

Itself: Out of 3 cases, 2 were correctly solved.

Them: There were 2 cases, all of them solved correctly.

Of interest is the fact that the pronoun *its* had a rather low success ratio. Lappin and Leass did not report on the success ratio of individual pronouns, and our data is too small to draw any conclusion. Further tests are necessary, and if considered necessary, the algorithm should be modified to enhance the results.

4 Discussion

The original algorithm can be applied to intersentential anaphora in the same way as for intrasentential. We only need to consider a few additional points. First of all, syntactic restrictions do not cross sentence boundaries. Thus, in theory, any noun in the previous sentence can corefer with a non-reflexive pronoun in the current sentence. Also, reflexives cannot corefer with nouns in previous sentences. Finally, the more sentence boundaries between the pronoun and the noun, the less likely the noun corefers with the pronoun. This is implemented by degrading (halving) the salience of the noun for every sentence boundary that is crossed.

The implementation of the anaphora resolution algorithm in ExtrAns allows for the possibility of intersentential anaphora only with the previous sentence. The reason for this restriction is twofold. First, it is very rare for a pronoun to corefer with a noun more that one sentence away. We did not find any case in the test corpus. Second, to compute the salience of a noun, it is necessary to know the syntactic structure of the sentence where the word appears, e.g. for the parallel roles reward. This means that we need to keep that information available together with other information regarding the noun. Since this information is very unlikely to be used at all, it is not practical to keep the information in the system.

ExtrAns' semantic interpreter uses the information from the anaphora resolution algorithm to merge the variables of the logical form predicates that correspond to words belonging to the same equivalence class.

For example, without information about equivalence classes, the semantic interpreter would produce the following logical form for the sentence *"The APU Generator is installed in the APU compartment, it is attached to the APU gearbox by a button hole flange"*:[8]

```
object(APU_generator,o1,[x2]),
evt(install,e4,[a4,x2]),
object(anonym_object,o5,[a4]),
in(e4,x8),object(APU_compartment,o2,[x8]),
object(it,o3,[x1]),evt(attach,e3,[x12,x1]),
object(button_hole_flange,o4,[x12]),
to(e3,x7), object(APU_gearbox,o6,[x7]).
```

[8] See (Mollá et al., 2000) for details about the logical forms

Since the anaphora resolution algorithm groups *directory* and *it* into the same equivalence class, the semantic interpreter replaces x1 with x2 and produces the following logical form:

```
object(APU_generator,o1,[x2]),
evt(install,e4,[a4,x2]),
object(anonym_object,o5,[a4]),
in(e4,x8),object(APU_compartment,o2,[x8]),
object(it,o3,[x2]),evt(attach,e3,[x12,x2]),
object(button_hole_flange,o4,[x12]),
to(e3,x7),object(APU_gearbox,o6,[x7]).
```

This way, a question like *"How is the APU generator attached to the APU gearbox?"* prompts ExtrAns to return the answer shown in Figure 2.

5 Conclusion

In this paper we have presented the approach adopted for Anaphora Resolution in a Question Answering System specifically developed to target technical documentation. The particular nature of the domain constraints the types of anaphoras that need to be targeted.

We think that technical documentation provides an important and interesting application for real-world Question Answering systems and certainly Anaphora Resolution has an important role to play.

References

Hiyan Alshawi. 1987. *Memory and Context for Language Interpretation*. Cambridge University Press, Cambridge.

James Dowdall, Michael Hess, Neeme Kahusk, Kaarel Kaljurand, Mare Koit, Fabio Rinaldi, and Kadri Vider. 2002. Technical terminology as a critical resource. In *Proc. LREC 2002*, pages 1897–1903, Las Palmas, Spain.

B. J. Grosz, K. J. Aravind, and S. Weinstein. 1995. Centering: A Framework for Modeling the Local Coherence of Discourse. *Computational Linguistics*, 21(2):203–255.

Barbara J. Grosz. 1981. Focusing and description in natural language dialogues. In Aravind K. Joshi, Bonnie L. Webber, and Ivan A. Sag, editors, *Elements of Discourse Understanding*, chapter 3, pages 84–105. Cambridge University Press, Cambridge.

Jerry R. Hobbs. 1978. Resolving pronoun references. *Lingua*, 44:311–338. Also in Grosz, Barbara J. & Karen Sparck Jones & Bonnie Lynn Webber (1986), *Readings in Natural Language Processing*, Kaufmann, Los Altos, CA.

O. Jong-Hoon, L. Kyung-Soon, C. Du-Seong, W. S. Chung, and C. Key-Sun. 2001. TREC-10 Experiments at KAIST: Batch Filtering and Question Answering. In *Proc. of TREC-10*, Gaithersburg, Maryland.

Christopher Kennedy and Branimir Boguraev. 1996. Anaphora for everyone: pronominal anaphora resolution without a parser. In *Proceedings of the 16th International Conference on Computational Linguistics*, Copenhagen, Denmark.

Shalom Lappin and Herbert J. Leass. 1994. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561.

Michael McCord, Arendse Bernth, Shalom Lappin, and Wlodek Zadrozny. 1992. Natural language processing within a slot grammar framework. *International Journal on Artificial Intelligence Tools*, 1(2):229–277.

R. Mitkov, R. Evans, and C. Orasan. 2002. A new, fully automatic version of Mitkov's knowledge poor pronoun resolution method. In *Proc. of CICLing 2002*, pages 168 – 186, Mexico City, Februrary.

Diego Mollá, Rolf Schwitter, Michael Hess, and Rachel Fournier. 2000. Extrans, an answer extraction system. *Traitment Automatique des Langues*, 41(2):495–522.

Manfred Pinkal. 1991. On the syntactic-semantic analysis of bound anaphora. In *Proc. of the Fifth Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Berlin.

Fabio Rinaldi, James Dowdall, Michael Hess, Diego Mollá, and Rolf Schwitter. 2002. Towards answer extraction: An application to technical domains. In Frank van Harmelen, editor, *Proc. ECAI2002 July 21-26 2002, Lyon, France*, pages 460–464, Amsterdam. IOS Press. Available at `http://www.cl.unizh.ch/CLpublications.html`.

Fabio Rinaldi, James Dowdall, Michael Hess, Kaarel Kaljurand, and Magnus Karlsson. 2003. The Role of Technical Terminology in Question Answering. In *Proceedings of TIA-2003, Terminologie et Intelligence Artificielle*, pages 156–165, Strasbourg, April. Available at `http://www.cl.unizh.ch/CLpublications.html`.

Daniel D. Sleator and Davy Temperley. 1993. Parsing English with a link grammar. In *Proc. Third International Workshop on Parsing Technologies*, pages 277–292.

Rob A. van der Sandt and Bart Geurts. 1991. Presupposition, anaphora, and lexical content. In Otthein Herzog and Claus-Rainer Rollinger, editors, *Text Understanding in LILOG: Integrating Computational Linguistics and Artificial Intelligence - final report on the IBM Germany LILOG project*, volume 546 of *Lecture Notes in Computer Science*, pages 259–296. Springer-Verlag, Berlin.

J. L. Vicedo and A. Ferrández. 2000. A Semantic Approach to Question Answering Systems. In *Proc. of TREC-9*, Gaithersburg, Maryland.