A Flexible Architecture for Reference Resolution

Donna K. Byron and *Joel R. Tetreault*
Department of Computer Science
University of Rochester
Rochester NY 14627, U.S.A.
dbyron/tetreaul@cs.rochester.edu

Abstract

This paper describes an architecture for performing anaphora resolution in a flexible way. Systems which conform to these guidelines are well-encapsulated and portable, and can be used to compare anaphora resolution techniques for new language understanding applications. Our implementation of the architecture in a pronoun resolution testing platform demonstrates the flexibility of the approach.

1 Introduction

When building natural language understanding systems, choosing the best technique for anaphora resolution is a challenging task. The system builder must decide whether to adopt an existing technique or design a new approach. A huge variety of techniques are described in the literature, many of them achieving high success rates on their own evaluation texts (cf. Hobbs 1986; Strube 1998; Mitkov 1998). Each technique makes different assumptions about the data available to reference resolution, for example, some assume perfect parses, others assume only POS-tagged input, some assume semantic information is available, etc. The chances are high that no published technique will exactly match the data available to a particular system's reference resolution component, so it may not be apparent which method will work best. Choosing a technique is especially problematic for designers of dialogue systems trying to predict how anaphora resolution techniques developed for written monologue will perform when adapted for spoken dialogue. In an ideal world, the system designer would implement and compare many techniques on the input data available in his system. As a good software engineer, he would also ensure that any pronoun resolution code he implements can be ported to future applications or different language domains without modification.

The architecture described in this paper was designed to provide just that functionality. Anaphora resolution code developed within the architecture is encapsulated to ensure portability across parsers, language genres and domains. Using these architectural guidelines, a testbed system for comparing pronoun resolution techniques has been developed at the University of Rochester. The testbed provides a highly configurable environment which uses the same pronoun resolution code regardless of the parser front-end and language type under analysis. It can be used, *inter alia*, to compare anaphora resolution techniques for a given application, to compare new techniques to published baselines, or to compare a particular technique's performance across language types.

2 The Architecture

2.1 Encapsulation of layers

Figure 1 depicts the organization of the architecture. Each of the three layers have different responsibilities:

The authors thank James Allen for help on this project, as well as the anonymous reviewers for helpful comments on the paper. This material is based on work supported by USAF/Rome Labs contract F30602-95-1-0025, ONR grant N00014-95-1-1088, and Columbia Univ. grant OPG:1307.

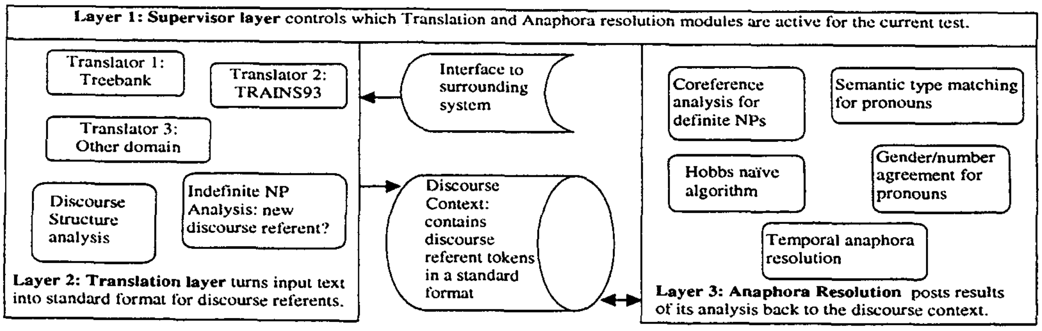
Layer 1: Supervisor layer controls which Translation and Anaphora resolution modules are active for the current test.
Translator 1: Treebank
Translator 2: TRAINS93
Translator 3: Other domain
Discourse Structure analysis
Indefinite NP Analysis: new discourse referent?
Layer 2: Translation layer turns input text into standard format for discourse referents.
Interface to surrounding system
Discourse Context: contains discourse referent tokens in a standard format
Coreference analysis for definite NPs
Semantic type matching for pronouns
Hobbs naïve algorithm
Gender/number agreement for pronouns
Temporal anaphora resolution
Layer 3: Anaphora Resolution posts results of its analysis back to the discourse context.

Figure 1: Reference Resolution Architecture

- **Layer 1: The supervisor** controls which modules in Layers 2 and 3 execute. In our implementation, the supervisor sets a runtime switch for each module in layer 2 and 3, and the first instruction of each of those modules checks its runtime flag to see if it is active for the current experiment.
- **Layer 2: Translation** reads the input text and creates the main data structure used for reference resolution, called the discourse context (DC). The DC consists of discourse entities (DEs) introduced in the text, some of which are anaphoric. This layer contains all syntactic and semantic analysis components and all interaction with the surrounding system, such as access to a gender database or a lexicon for semantic restrictions. All features that need to be available to reference resolution are posted to the DC. This layer is also responsible for deciding which input constituents create DEs.
- **Layer 3: Anaphora resolution** contains a variety of functions for resolving different types of anaphora. Responsibilities of this layer include determining what anaphoric phenomena are to be resolved in the current experiment, determining what anaphora resolution technique(s) will be used, and determining what updates to make to the DC. Even though the modules are independent of the input format, they are still somewhat dependent on the availability of DE features. If a feature needed by a particular resolution module was not created in a particular experiment, the module must either do without it or give up and exit. This layer's output is an updated DC with anaphoric elements resolved to their referents. If labeled training data is available, this layer is also responsible for calculating the accuracy of anaphora resolution.

2.2 Benefits of this design

This strict delineation of responsibilities between layers provides the following advantages:

- Once a translation layer is written for a specific type of input, all the implemented anaphora resolution techniques are immediately available and can be compared.
- Different models of DC construction can be compared using the same underlying reference resolution modules.
- It is simple to activate or deactivate each component of the system for a particular experiment.

3 Implementation

We used this architecture to implement a testing platform for pronoun resolution. Several experiments were run to demonstrate the flexibility of the architecture. The purpose of this paper is not to compare the pronoun resolution results for the techniques we implemented, so pronoun resolution accuracy of particular techniques will not be discussed here.[1] Instead, our implementation is described to provide some examples of how the architecture can be put to use.

3.1 Supervisor layer

The supervisor layer controls which modules within layers 2 and 3 execute for a particular experiment. We created two different supervisor

[1] See (Byron and Allen, 1999; Tetreault, 1999) for results of pronoun resolution experiments run within the testbed.

modules in the testbed. One of them simply reads a configuration file with runtime flags hard-coded by the user. This allows the user to explicitly control which parts of the system execute, and will be used when a final reference resolution techniques is chosen for integration into the TRIPS system parser (Ferguson and Allen, 1998).

The second supervisor layer was coded as a genetic algorithm (Byron and Allen, 1999). In this module, the selection of translation layer modules to execute was hard-coded for the evaluation corpus, but pronoun resolution modules and methods for combining their results were activated and de-activated by the genetic algorithm. Using pronoun resolution accuracy as the fitness function, the algorithm learned an optimal combination of pronoun resolution modules.

3.2 Translation layer

Translation layer modules are responsible for all syntactic and semantic analysis of the input text. There are a number of design features that must be controlled in this layer, such as how the discourse structure affects antecedent accessibility and which surface constituents trigger DEs. All these design decisions should be implemented as independent modules so that they can be turned on or off for particular experiments.

Our experiments created translation modules for two evaluation corpora: written news stories from the Penn Treebank corpus (Marcus et al., 1993) and spoken task-oriented dialogues from the TRAINS93 corpus (Heeman and Allen, 1995). The input format and features added onto DEs from these two corpora are very different, but by encapsulating the translation layer, the same pronoun resolution code can be used for both domains. In both of our experiments only simple noun phrases in the surface form triggered DEs.

Treebank texts contain complete structural parsers, POS tags, and annotation of the antecedents of definite pronouns (added by Ge et al. 1998). Because of the thorough syntactic information, DEs can be attributed with explicit phrase structure information. This corpus contains unconstrained news stories, so semantic type information is not available. The Treebank translator module adds the following features to each DE:

1. Whether its surface constituent is contained in reported speech;
2. A list of parent nodes containing its surface constituent in the parse tree. Each node's unique identifier encodes the phrase type (i.e. VB, NP, ADJP);
3. Whether the surface constituent is in the second half of a compound sentence;
4. The referent's animacy and gender from a hand-coded agreement-feature database.

A second translation module was created for a selection of TRAINS93 dialogue transcripts. The input was POS-tagged words with no structural analysis. Other information, such as basic punctuation and whether each pronoun was in a main or subordinate clause, had previously been hand-annotated onto the transcripts. We also created an interface to the semantic type hierarchy within the Trains system and added semantic information to the DEs.

Common DE attributes for both corpora:

1. Plural or singular numeric agreement;
2. Whether the entity is contained in the subject of the matrix clause;
3. Linear position of the surface constituent;
4. Whether its surface constituent is definite or indefinite;
5. Whether its surface constituent is contained in quoted speech;
6. For pronoun DEs, the id of the correct antecedent (used for evaluation).

3.3 Anaphora resolution layer

Modules within this layer can be coded to resolve a variety of anaphoric phenomena in a variety of ways. For example, a particular experiment may be concerned only with resolving pronouns or it might also require determination of coreference between definite noun phrases. This layer is reminiscent of the independent anaphora resolution modules in the Lucy system (Rich and LuperFoy, 1988), except that modules in that system were not designed to be easily turned on or off.

For our testbed, we implemented a variety of pronoun resolution techniques. Each technique

Pronoun resolution module	Activated for Treebank	Activated for TRAINS93
Baseline most-recent technique that chooses closest entity to the left of the pronoun	X	X
Hobbs naive algorithm	X	
Choose most recent entity that matches sub-categorization restrictions on the verb		X
Strube's s-list algorithm (Strube, 1998)	X	X
Boost salience for the first entity in each sentence	X	X
Decrease salience for entities in prepositional phrases or relative clauses	X	
Increase the salience for non-subject entities for demonstrative pronoun resolution (Schiffman, 1985)		X
Decrease salience for indefinite entities	X	X
Decrease salience for entities in reported speech	X	
Increase the salience of entities in the subject of the previous sentence	X	X
Increase the salience of entities whose surface form is pronominal	X	X

Table 1: Pronoun resolution modules used in our experiments

can run in isolation or with the addition of meta-modules that combine the output of multiple techniques. We implemented meta-modules to interface to the genetic algorithm driver and to combine different salience factors into an overall score (similar to (Carbonell and Brown, 1988; Mitkov, 1998)). Table 1 describes the pronoun resolution techniques implemented at this point, and shows whether they are activated for the Treebank and the TRAINS93 experiments. Although each module could run for both experiments without error, if the features a particular module uses in the DE were not available, we simply de-activated the module. When we migrate the TRIPS system to a new domain this year, all these pronoun resolution methods will be available for comparison.

4 Summary

This paper has described a framework for reference resolution that separates details of the syntactic/semantic interpretation process from anaphora resolution in a plug-and-play architecture. The approach is not revolutionary, it simply demonstrates how to apply known software engineering techniques to the reference resolution component of a natural language understanding system. The framework enables comparison of baseline techniques across corpora and allows for easy modification of an implemented system when the sources of information available to anaphora resolution change. The architecture facilitates experimentation on different mixtures of discourse context and anaphora resolution algorithms. Modules written within this framework are portable across domains and language genres.

References

Donna K. Byron and James F. Allen. 1999. A genetic algorithms approach to pronoun resolution. Technical Report 713, Department of Computer Science, University of Rochester.

Jaime G. Carbonell and R.D. Brown. 1988. Anaphora resolution: a multy-strategy approach. In *COLING '88*, pages 96–101.

George Ferguson and James F. Allen. 1998. Trips: An intelligent integrated problem-solving assistant. In *Proceedings of AAAI '98*.

Niyu Ge, John Hale, and Eugene Charniak. 1998. A statistical approach to anaphora resolution. In *Proceedings of the Sixth Workshop on Very Large Corpora*.

Peter A. Heeman and James F. Allen. 1995. The Trains spoken dialog corpus. CD-ROM, Linguistics Data Consortium.

Jerry Hobbs. 1986. Resolving pronoun reference. In *Readings in Natural Language Processing*. Morgan Kaufmann.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Ruslan Mitkov. 1998. Robust pronoun resolution with limited knowledge. In *Proceedings of ACL '98*, pages 869–875.

Elaine Rich and Susann LuperFoy. 1988. An architecture for anaphora resolution. In *Conference on Applied NLP*, pages 18–24.

Rebecca Schiffman. 1985. *Discourse constraints on 'it' and 'that': A study of language use in career-counseling interviews.* Ph.D. thesis, University of Chicago.

Michael Strube. 1998. Never look back: An alternative to centering. In *Proceedings of ACL '98*, pages 1251–1257.

Joel R. Tetreault. 1999. Analysis of syntax-based pronoun resolution methods. In *Proceedings of ACL '99*.