Larger-first partial parsing

2003

Sebastian Alexander Van Delden

Find similar works at: http://stars.library.ucf.edu/rtd

University of Central Florida Libraries http://library.ucf.edu

Part of the Electrical and Computer Engineering Commons

STARS Citation


This Doctoral Dissertation (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Retrospective Theses and Dissertations by an authorized administrator of STARS. For more information, please contact lee.dotson@ucf.edu.
LARGER-FIRST PARTIAL PARSING

By
Sebastian Alexander Van Delden

2003

UCF
LARGER-FIRST PARTIAL PARSING

by

SEBASTIAN ALEXANDER VAN DELDEN
B.S. University of Central Florida, 1999
M.S. University of Central Florida, 2001

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the School of Electrical Engineering and Computer Science in the College of Engineering and Computer Science at the University of Central Florida Orlando, Florida

Fall Term
2003
ABSTRACT

*Larger-first* partial parsing is a primarily top-down approach to partial parsing that is opposite to current *easy-first*, or primarily bottom-up, strategies. A rich partial tree structure is captured by an algorithm that assigns a hierarchy of structural tags to each of the input tokens in a sentence.

Part-of-speech tags are first assigned to the words in a sentence by a part-of-speech tagger. A cascade of Deterministic Finite State Automata then uses this part-of-speech information to identify syntactic relations primarily in a descending order of their size. The cascade is divided into four specialized sections: (1) a *Comma Network*, which identifies syntactic relations associated with commas; (2) a *Conjunction Network*, which partially disambiguates phrasal conjunctions and fully disambiguates clausal conjunctions; (3) a *Clause Network*, which identifies non-comma-delimited clauses; and (4) a *Phrase Network*, which identifies the remaining base phrases in the sentence. Each automaton is capable of adding one or more levels of structural tags to the tokens in a sentence. The larger-first approach is compared against a well-known easy-first approach. The results indicate that this larger-first approach is capable of (1) producing a more detailed partial parse than an easy first approach; (2) providing better containment of attachment ambiguity; (3) handling overlapping syntactic relations; and (4) achieving a higher accuracy than the easy-first approach. The automata of each network were developed by an empirical analysis of several sources and are presented here in detail.
To my loving wife Elizabeth, the center of my universe and pinnacle of my devotion and happiness. Also to my mother Wilma whose endless caring and encouragement has guided me through life’s challenges, sculpting me into the person that she knew I could be.
ACKNOWLEDGMENTS

First and foremost I would like to extend a special thanks to my adviser, Fernando Gomez. Your guidance and commitment to excellence has inspired me to strive for perfection in all that I undertake.

I would also like to thank the National Aeronautics and Space Administration for their partial funding of this research through several grants over the years (NASA grants 16-40-201 and 16-40-202).
# TABLE OF CONTENTS

LIST OF TABLES ............................................................................................................ x
LIST OF FIGURES ........................................................................................................... xi
LIST OF ABBREVIATIONS .............................................................................................. xiv

1 INTRODUCTION ........................................................................................................... 1
   1.1 Methodology ........................................................................................................ 3
   1.2 Motivation ............................................................................................................ 6
      1.2.1 Comma Analysis .......................................................................................... 6
      1.2.2 Richness ....................................................................................................... 8
      1.2.3 Part-of-Speech Tag Ambiguity .................................................................... 9
      1.2.4 Overlapping Syntactic Relations ................................................................ 10
   1.3 Overview of Dissertation ..................................................................................... 11

2 PART-OF-SPEECH TAGGING ..................................................................................... 12
   2.1 Introduction ......................................................................................................... 13
   2.2 Part-of-Speech Tagsets ...................................................................................... 14
   2.3 Stochastic ........................................................................................................... 16
   2.4 Rule-Based ......................................................................................................... 22
      2.4.1 Transformation-Based Rule Learning ......................................................... 24
   2.5 Other Approaches .............................................................................................. 28
      2.5.1 Neural Networks ......................................................................................... 28
      2.5.2 Hybrid Environments ................................................................................. 31
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5.3 Support Vector Machines</td>
<td>33</td>
</tr>
<tr>
<td>2.5.4 Memory(Example)-Based</td>
<td>35</td>
</tr>
<tr>
<td>2.5.5 Decision Trees</td>
<td>36</td>
</tr>
<tr>
<td>2.5.6 Maximum Entropy</td>
<td>36</td>
</tr>
<tr>
<td>2.6 Comparing Part-of-speech Taggers</td>
<td>37</td>
</tr>
<tr>
<td>2.7 Summary and Conclusions</td>
<td>39</td>
</tr>
</tbody>
</table>

3 THE SYNTACTIC RELATION SET ......................................................... 41

3.1 Pre-Processing ............................................................................ 43
3.2 Commas ....................................................................................... 43
  3.2.1 Speech Commas ...................................................................... 45
  3.2.2 Series Commas ..................................................................... 46
  3.2.3 Clausal Commas ................................................................... 47
  3.2.4 Enclosing Commas ................................................................ 49
3.3 Coordinate Conjunctions ................................................................ 51
  3.3.1 Clausal Conjuncts ................................................................ 52
  3.3.2 Phrasal Conjuncts ................................................................ 53
3.4 Clauses ....................................................................................... 54
3.5 Phrases ....................................................................................... 56
3.6 CASS Category Set ....................................................................... 56

4 THE LARGER-FIRST PARADIGM .......................................................... 59

4.1 Initial Design ............................................................................ 59
4.2 Conceptual Model ....................................................................... 62
4.3 The Algorithm ............................................................................ 69
4.4 Some Notation ............................................................................ 74
4.5 Related Research ....................................................................... 75
## LIST OF TABLES

2.1 POS Tagging Methods and their Representation of Information...............................39

3.1 Complete Set of Syntactic Categories.......................................................................42

3.2 Frequencies of Syntactic Relations Associated with Commas.................................44

3.3 Syntactic Categories Identified by the CASS System..............................................58

5.1 Comma Tagging - System Test Results.....................................................................107

5.2 Comma Tagging - Comma Tags with Level of Modification Indicators.....................112

5.3 Comma Tagging - Dutch System Test Results.........................................................121

6.1 Conjunction Tagging - Relaxed Analysis of Performance Results............................141

8.1 Results of the Evaluation of the Encyclopedia Encarta........................................157

8.2 Evaluation and comparison of the CASS and LAFI systems..................................163
# LIST OF FIGURES

1.1 The Larger-First Partial Parsing Approach ................................................................. 4
1.2 Partial Parsing Guidelines ......................................................................................... 5
1.3 Wall Street Journal Penn Treebank III Comma Frequencies ...................................... 7
2.1 Example Corpus used to Illustrated Stochastic POS Tagging ..................................... 16
2.2 Markov Chain from the Example Corpus .................................................................... 18
2.3 Hidden Markov Model of the Example Corpus ......................................................... 19
2.4 Transformation-Based Error-Driven Learning Strategy .............................................. 24
2.5 A Simple Neural Network Architecture used for POS Tagging ................................. 29
2.6 Elastic Neural Network used for POS Tagging ......................................................... 31
2.7 Binary Classification using Support Vector Machines .............................................. 34
4.1 Original Larger-First Partial Parsing Algorithm ....................................................... 60
4.2 Conceptual Model ...................................................................................................... 65-66
4.3 The Larger-First Partial Parsing Algorithm ............................................................. 69
5.1 Phrasal Subordinate Conjunctions ............................................................................. 80
5.2 Phrasal Prepositions ................................................................................................. 81
5.3 Direct Speech ........................................................................................................... 82
5.4 Indirect Speech ........................................................................................................ 84
5.5 List of Infinitival Clauses ......................................................................................... 86
5.6 List of Verb Clauses ................................................................................................. 87
5.7 List of Noun Phrases .................................................................88
5.8 Coordinated Verb Clause Enclosed by Commas .................................91
5.9 Infinitival Clause Enclosed by Commas ............................................92
5.10 Two Types of Relative Clauses Enclosed by Commas ..........................93
5.11 Subordinate Clause Enclosed by Commas ..........................................95
5.12 Reduced Subordinate Clause Enclosed by Commas ............................96
5.13 Time Noun phrases Enclosed by Commas ........................................97
5.14 Appositions .................................................................................98
5.15 Transitional Phrases .................................................................100
5.16 Prepositional Phrases Enclosed by Commas .....................................100
5.17 Coordinated Independent Clauses ..................................................101
5.18 Coordinated Noun Phrase Enclosed with Commas .............................102
5.19 Comma-Tagging - The Structure of the Co-occurrence Matrix .............104
5.20 Comma-Tagging - Co-occurrence Matrix Processing Algorithm ........105
5.21 Comma-Tagging - The Greedy Learning Algorithm ..........................106
5.22 Comma-Tagging - Non-Comma Relative Clause Automata for Dutch ....117
5.23 Comma-Tagging - Dutch Relative Clause Automata ..........................118
6.1 Coordinated Relative Clauses .....................................................127
6.2 Coordinated Subordinate Clauses ...................................................129
6.3 Coordinated Infinitival Clauses ....................................................130
6.4 Coordinated Gerund, Participle, or Reduced Subordinate Clauses ........131
6.5 Double and Single Coordinated Verb Clauses ...................................132
6.6 Coordinated Phrase Automata .....................................................133
6.7 Conjunction Tagging - Strict Analysis of Performance Results ........................................142
7.1 Subordinate Clause Automaton ..................................................................................144
7.2 Infinitival Clause Automaton .....................................................................................146
7.3 Relative Clause Automata ..........................................................................................147
7.4 Gerund, Participle or Reduced Subordinate Clause Automaton ..................................149
7.5 Prepositional Phrase Automaton ..................................................................................149
7.6 Time Noun Phrase Automaton ....................................................................................150
7.7 Noun Phrase Automaton ..........................................................................................151
7.8 Verb Phrase Automaton ............................................................................................154
7.9 Adjective and Adverbial Phrase Automata ..................................................................154
LIST OF ABBREVIATIONS
(Used to Define the Arc Labels of the Finite State Automata - see Section 4.4 for more details)

T: Any of the following tags can be present

W: Any of the following words can be present

PREV: The previous section of the sentence must contain any of the following tags

NEXT: The following part of the sentence must contain any of the following tags

LASTTAG: The last part-of-speech tag is any of the following

NEXTTAG: The next part-of-speech tag is any of the following

[TAG] A forward call is made to the $TAG$ automaton. Take arc if $TAG$ accepts.

<TAG> A backward reference is made to any tags assigned by the $TAG$ automaton

START The start of sentence marker

EOS The End Of the Sentence (T: .)

Default Always take this arc after attempting any other possible arcs

Italicization If a label is italicized, it is not assigned a structural tag

Comma A comma is present (T: ,)

Quote A quotation mark is present (T: “)

REL Refers to all of the relative clause automata: REL1, REL2, and REL3.

CO-LST-NP Refers to both list of noun phrases automata: CO-LST-NP1 and CO-LST-NP2

OR/AND Logical OR/AND

> Structural tag prefix indicating that the current token is grouped with the next

! Negates the meaning of an abbreviation
CHAPTER 1

INTRODUCTION

*We choose to do these things, not because they are easy, but because they are hard.*

- John F. Kennedy

The age of Artificial Intelligence is upon us. Computer systems and technology have undoubtedly transformed the world in which we live. We rely on them for almost every aspect of our daily activities. We are, however, not satisfied by the astonishing computational power of today’s super computers. We continually strive to make computing machines faster, smaller and most of all *smarter.*

Since John McCarthy coined its name in the summer of 1956 at Darmouth College, *Artificial Intelligence* has made giant strides forward, diverging off into several specialized areas of research, such as: Natural Language Processing, Computer Vision, and Machine Learning. This research focuses on a specific topic within the realm of Natural Language Processing (NLP).

Presently, NLP tools are being used by various types of industries, such as: web search engines, automated telecommunications services, natural language translators, word processing systems, database systems, and many other types of computer applications. A technique that can understand, or interpret, natural language sentences in an unrestricted domain would greatly enhance the capabilities of any NLP system.
An *interpreted* sentence is defined here as one in which all syntactic as well as semantic ambiguities have been resolved. How to approach this task is still a topic of debate. Some earlier work (notably Schank, 1975) focused on developing a semantic representation of natural language with a small set of primitives without considering syntactic information. Conversely others have attempted to use syntactic information alone to make difficult structural decisions. For example, Brill and Resnik (1994) and Hindle and Rooth (1993) use syntactic rules learned from a corpus to attempt to disambiguate prepositional phrase attachment. The philosophy taken here is one of *minimalist syntax*: syntax-based approaches first attempt to resolve as much ambiguity as possible while postponing decisions that should be made by considering the semantic information in the sentence. This approach can be implemented as follows:

Step 1) *Syntax* – identify as many syntactic relations and resolve as much syntactic ambiguity as possible using syntax-based approaches (van Delden and Gomez, 2003a);

Step 2) *Semantics* – resolve structural ambiguity and assign *meaning* to each of the identified syntactic components (Gomez, 2003 and 2001).

This dissertation addresses the syntactic issues at hand by describing a finite state approach to identifying the syntactic relations in a natural language sentence (Step 1). A plateau is defined to which a syntax-based approach should aspire but not exceed. A simple, decomposable system of specialized finite-state components is developed which ultimately produces a set of syntactic relations that can be exploited by a semantic interpreter.
1.1 Methodology

A growing trend in natural language processing is to decompose a parser into intermediate components. First, part-of-speech information is assigned to each word in the sentence. Next, a partial parse of the sentence identifying larger phrases and clauses is produced. Finally, a semantic analysis is performed to determine verb meaning, thematic roles, and resolve attachment issues.

Unlike part-of-speech tagging (Brants, 2000; Brill, 1994) and semantic interpretation (Goma, 2003 and 2001), no formal standard has been defined which establishes the detail or richness that a partial parser can and should achieve. This dissertation presents a **larger-first**, primarily top-down, approach to partial parsing which is opposite to current **easy-first**, or primarily bottom-up, approaches.

Abney (1996a, 1996b) defines an **easy-first** finite-state approach to partial parsing in which smaller syntactic relations, like noun and verb phrases, are identified first then combined to form larger syntactic relations, like prepositional phrases and relative and subordinate clauses. Levels of finite state automata are used to produce a partial parse. The output of level$_i$ is the input to level$_{i+1}$. For example, consider the sentence The woman in the lab coat thought you were sleeping. The chunks in the sentences are recognized as follows:

- **Level$_1$** [The woman] in [the lab coat] thought [you] [were sleeping]
- **Level$_2$** [The woman][ in [the lab coat]] thought [you] [were sleeping]
- **Level$_3$** [[The woman][ in [the lab coat]] thought][ [you] [were sleeping]]
The *larger-first* approach, on the contrary, identifies syntactic relations primarily in descending order of their size. A cascade of deterministic finite-state automata assigns a hierarchy of tags to the tokens in the sentence. The cascade can be divided into four specialized sections: 1) a *Comma Network*, which identifies syntactic relations associated with commas; 2) a *Conjunction Network*, which partially disambiguates phrasal conjunctions and fully disambiguates clausal conjunctions; 3) a *Clause Network*, which identifies non-comma delimited clauses; and 4) a *Phrase Network*, which identifies the remaining base phrases in the sentence. Figure 1.1 summarizes the larger-first approach.

![Figure 1.1 The Larger-First Partial Parsing Approach](image)

The words in an input sentence are first assigned part-of-speech tags by a part-of-speech tagger (Brill, 1994). This part-of-speech information is built in to the arcs of the automata. The entire system is declarative and the automata for each network are defined in text files. The output is a rich partial parse which differs from a full parse only by avoiding explicit attachment (structural) decisions. The automata presented in this dissertation were developed by an empirical analysis of several sources. They are not meant to be an exhaustive list of every possible syntactic structure in the English
language, but an indication of the structure of the most frequently occurring syntactic relations in unrestricted written text.

Three formal guidelines to partial parsing are set forth by this larger-first approach (hereafter referred to as the Guidelines) and are shown in Figure 1.2 below.

1) Explicit attachment decisions are always avoided.
2) Only comma information can be used to confine attachment ambiguity.
3) A syntactic relation may be a complement to, an attachment to, or in coordination with a peer syntactic relation or a relation within a preceding sub-clause.

**Figure 1.2 Partial Parsing Guidelines**

Explicit attachment decisions are always avoided because syntax alone is inadequate when making such decisions. Consider the following sentences: *Many foreigners came to Hawaii to work on the plantations* versus *Many foreigners came to Hawaii to work via transportation vessels*. It cannot be determined based on syntax alone where the final prepositional phrase should be attached. Therefore *on the plantations* and *via transportation vessels* are left unattached - they could be attached within the infinitival sub-clause or to another peer syntactic relation in the sentence, like the main verb *came*.

An important distinction made in this work is the difference between comma-delimited versus non-comma-delimited syntactic relations. The comma information in the sentence permits a far better containment of attachment ambiguity. *Containment of ambiguity* (Abney, 1997) refers to limiting the attachment sites of a syntactic relation to its consuming clause. For example, *Henry picked up the hammer, [REL which lay next to*
the nails on the table]. The attachment sites of the prepositional phrases next to the nails and on the table are limited to within the comma-delimited relative clause.

1.2 Motivation

Four advantages to this larger-first approach are discussed below, answering the following questions:

1) It has already been noted that comma information will be used to confine attachment ambiguity, but do commas appear often enough and in such crucial positions to warrant a special network of automata?

2) Can an easy-first system provide the detail that a larger-first partial parser is capable of producing?

3) Is the larger-first approach more suitable for handling the ambiguous Penn Treebank part of speech tags (Santorini, 1995) that are used by both systems?

4) Can the design of this larger-first approach identify both instances of overlapping syntactic relations?

1.2.1 Comma Analysis

The first step in larger-first partial parsing is to identify syntactic relations associated with commas (van Delden and Gomez, 2002 and 2003; Bayraktar et al., 1998) since commas are usually used to delimit or compose large syntactic relations. Jones (1994) notes that the comma is the most abundant punctuation mark in the Wall Street Journal Penn Treebank (Marcus et al., 1993). A closer analysis of the Wall Street Journal Section of the Penn Treebank III reveals that 65% of all sentences contain at least one comma
with: 20% containing two commas; 9% containing three commas; 4% containing four commas, and almost 2% containing five or more commas. Figure 1.3 summarizes the comma frequencies in the Wallstreet Journal Section of the Penn Treebank III.

![Comma Frequencies in the Wallstreet Journal Section of the Penn Treebank III](image)

**Figure 1.3** Comma Frequencies in the Wallstreet Journal Section of the Penn Treebank III corpus. Of the 52726 total sentences, 34224 contained at least one comma.

The syntactic roles of commas can be determined with a 95% accuracy when tested on this corpus (van Delden and Gomez, 2002). A specialized network of automata for commas is therefore a good foundation on which the rest of the partial parse should be built. Consider the following sentence:

Some seals, such as the leopard seal (*Hydrurga leptonyx*), are quite predatory, feeding on penguins, other birds that land on water, and other seals.

A partial parsing system must pay special attention to the roles that are being played by the commas in this sentence in order to realize that there is a reduced relative clause
(introduced by a comma) which contains a list of noun phrases that in turn contains another embedded relative clause. Such sentences are abundant in the Penn Treebank III and other real world texts.

1.2.2 Richness

Richness of a partial parse refers to the level of detail a partial parsing system produces. Attempting to disambiguate appositions from lists of noun phrases would violate the easy-first approach. Syntactically, an apposition is (usually) a noun phrase enclosed by commas that appositives a preceding noun phrase. This relation is syntactically smaller than a list of noun phrases which is comprised of at least three noun phrases, one or two commas, and a conjunction. Consider the following sentence: *John eats a banana, an apple, or a pear for breakfast.* *An apple* would be incorrectly identified as an apposition by the easy-first approach since syntactically smaller relations are identified prior to larger relations. This would occur whenever there is a list of at least three noun phrases with a comma before the conjunction. The easy-first approach cannot be extended to make this distinction since smaller relations are always identified prior to larger relations.

Unlike the larger-first approach, the easy-first approach is also incapable of being extended to disambiguate coordinate conjunctions. Conjunctions coordinating clauses are fully disambiguated, while conjunctions coordinating phrases are partially disambiguated. Partial disambiguation is defined as identifying the post-conjunct of a coordinate conjunction. For example, *We sold the car with the cloth interior and the truck after our boss left.* The conjunction (*and*) is identified as coordinating a noun phrase. However, no attempt is made to identify the pre-conjunct (*the car* or *the cloth interior*). Baker (1995)
suggests that coordination can be resolved by identifying the largest relation of similar syntax on either side of the conjunction. The larger-first approach is ideal for accomplishing such a task. Consider the following sentence: *The boys went to the beach and the girls went to the mall.* The larger-first approach identifies the larger syntactic relations on either side of the conjunction, determining that the conjunction coordinates two independent clauses. An easy-first approach, however, would incorrectly identify the conjunction as coordinating the smaller syntactic relations surrounding the conjunction – the noun phrases.

### 1.2.3 Part-of-Speech Tag Ambiguity

The larger-first approach is also preferred due to the ambiguity that can be present in part-of-speech tags. For example, the Penn Treebank Tagset (Santorini, 1995) provides one tag (IN') to identify either prepositions or subordinate conjunctions. Following an easy-first approach, will result in an error whenever a subordinate conjunction which could also be a preposition is present. For example, *John went to the black board after the teacher threatened to expel him.* In the easy-first approach, *after the teacher* would first be identified as a prepositional phrase. This error could possibly be changed later during processing. The larger-first approach identifies the syntactically larger subordinate clause *after the teacher threatened to expel him* first, no later changes are needed.

Even if a finer-grained tagset was able to distinguish between prepositions and subordinate conjunctions, a larger-first approach would have to be taken in order to assign these part-of-speech tags. In either case, a larger-first strategy would have to be employed to resolve such ambiguous cases.
1.2.4 Overlapping Syntactic Relations

Even though larger syntactic relations are identified prior to smaller ones, the larger-first approach is not strictly a top-down approach because of overlapping syntactic relations that are present in natural language. Abney (1997) also notes that following a strictly bottom-up (or top-down) approach is not desired. For example, consider the sentence *Beth bought a television, a DvD player that turned out to be broken, and ten DvD movies.* Note that the sentence contains a list of noun phrases which contains an embedded relative clause. Here the list of noun phrases is syntactically larger than the relative clause. However, consider the sentence *John bought a television that was equipped with a remote controller, a DvD hook-up, and HDTV capabilities.* Note that this sentence contains a relative clause with an embedded list of noun phrases. In contrast to the previous sentence, the relative clause here is syntactically larger than the list of noun phrases. Lists of noun phrases and relative clauses are examples for what is referred to here as overlapping syntactic relations.

The larger-first approach is capable of correctly identifying both situations by allowing the automata at different levels in the cascade to interact with each other (see Chapter 4). The easy-first system, however, is only able to identify one of the two situations of overlapping syntactic relations. Since there is limited interaction between the automata in the easy-first system, once a smaller syntactic relation is identified it cannot be correctly modified to include a larger syntactic relation.
1.3 Overview of Dissertation

Since part-of-speech tags serve as the basic units on which the arcs of the system's automata are taken, it is only necessary to give a comprehensive overview of part-of-speech tagging techniques. Chapter 2 therefore provides a thorough explanation of each part-of-speech tagging paradigm, including detailed descriptions of specific implementations of these paradigms. The most popular part-of-speech tagging strategies - Rule-based and Stochastic - are explained in detail. Several other approaches are also presented such as: Neural Network, Decision Tree, Maximum Entropy, Support Vector Machines, Memory(Example)-based, and Hybrid strategies.

Chapter 3 introduces the set of syntactic relations identified by the larger-first approach and compares them to the gold-standard tags set forth by the Penn Treebank Project (Marcus et al., 1993) and also to the syntactic relation set used by the easy-first system - CASS (Abney, 1996). Chapter 4 presents the larger-first approach in more detail and describes the algorithm used to assign the hierarchy of structural tags. Chapter 5 presents the comma network of automata. The concept of a finite state comma tagger is also presented in this chapter as well as a study that shows how the finite state comma tagger can be extended from English to the Dutch language. Chapter 6 presents the conjunction network of automata, and Chapter 7 the clause and phrase networks of automata. A detailed evaluation and comparison is performed between the larger-first and the CASS systems in Chapter 8. Finally, Chapter 9 concludes and summarizes the ideas presented in this work.
PART-OF-SPEECH TAGGING

Automatic text tagging is an important first step in discovering the linguistic structure of large text corpora. Part-of-Speech information facilitates higher level analysis, such as recognizing noun phrases and other patterns in text.

Cutting et al. (1992)

The initial step in larger-first partial parsing is to determine the parts of speech of the words in the sentence. This can be accomplished by assigning a part-of-speech tag to each word in the sentence. Since part-of-speech tags represent the basic elements on which the rest of the system relies, it is only necessary to give a comprehensive overview of part-of-speech tagging strategies.

After further introduction, this chapter first describes a few well-known tag sets. The two major part-of-speech tagging paradigms, stochastic and rule-based, are then discussed in detail. An overview of various other, less well-known strategies is presented next, such as: Neural Network, Hybrid, Support Vector Machines, Maximum Entropy, Decision Tree, and Memory(Example)-based strategies. Finally, a discussion of how part-of-speech taggers can be compared is presented, followed by a summarization and some concluding remarks. An analysis of part-of-speech tagging errors encountered during the implementation of this system is presented in the Evaluation Chapter (Section 8.3).
2.1 Introduction

For over three decades, automatic part-of-speech tagging has been a hot topic of research in Natural Language Processing and Artificial Intelligence. The task seems simple - assign a descriptive part-of-speech tag to each word in a sentence. Many words, however, have multiple part-of-speech categories, making this task extremely difficult, perhaps impossible, for computers to perfect.

A computer program which automatically assigns part-of-speech tags to the words in a sentence is called a *part-of-speech tagger*. Part-of-speech tagging research can be divided into two areas: (1) Using information from a correctly tagged corpus to tag new, unseen sentences - a *supervised* approach to part-of-speech tagging; and (2) Trying to induce a set of tags from an untagged corpus - an *unsupervised* approach to part-of-speech tagging. With the advent of large bodies of accurately tagged text (like the Penn Treebank (Marcus et al., 1993) and the British National Corpus (Leech and Smith, 2000), the focus of part-of-speech tagging research has been on the supervised part-of-speech tagging paradigm.

Only supervised taggers are described below since a pre-defined set of tags is used in this work. For more information on unsupervised part-of-speech tagging refer to Brill (1997), and Brill and Marcus (1992). Supervised taggers are separated into two main groups, *stochastic* (statistical) and *rule-based*, and a third group which encompasses other less popular approaches such as *Neural Network, Decision Tree, Support Vector Machines, Maximum Entropy, Memory(Example)-based, and Hybrid* strategies.

These part-of-speech tagging paradigms all rely on *Lexical* and *Contextual* information. Lexical information is specific to a single lexical item and is divided into
two types: (1) the likelihood of a part-of-speech tag for a known word; and (2) Affix and character clues for determining unknown words. Contextual information takes into consideration how a word is used in the context of other words and/or tags in a sentence. The part-of-speech tagging paradigms discussed here employ radically different techniques to acquiring, storing, and utilizing this information.

One could argue that the two main categories of part-of-speech tagging are whether the tagger is created by hand (linguistic approach) or automatically with data-driven techniques. Originally, only stochastic methods were data-driven, using statistics gathered from a corpus. However, transformation-based error-driven learning techniques (Ngai and Florian, 2001; Daelemans, 1999; Brill, 1995; Brill, 1994; Brill, 1992) have shown that rule-based systems can also be generated from a corpus (see Section 2.4.1 for details).

The part-of-speech tagger I chose for this larger-first partial parsing system is rule-based. The reasons for choosing this approach are: (1) Brill’s part-of-speech tagger (Brill, 1992 - a rule-based tagger - see Section 2.4.1) was available for download, (2) Brill’s tagger had been trained on the Penn Treebank and was ready to be used, (3) rule-based taggers are small, compact and portable, and (4) Brill’s tagger has been shown to achieve accuracy comparable to state-of-the-art stochastic taggers.

### 2.2 Part-of-Speech Tagsets

Besides indicating the major part-of-speech classes of words (noun, adjective, verb, etc...), part-of-speech tags usually provide some extra information within a major word class. For example, whether a noun is singular or plural. This is known as
morphosyntactic part-of-speech tagging. Several well-known sets of part-of-speech tags have been created and employed by different projects. These tag sets differ in syntax and granularity - the degree of morphosyntactic information they provide:

The **Penn Treebank Tagset**¹ (Marcus et al., 1993) – coarse granularity. More ambiguity is left in the tagset, for example, the IN tag can represent either a preposition or a subordinate conjunction.

The **Brown Corpus Tagset**¹ (Francis and Kucera, 1982) – medium granularity, consisting of about one hundred tags.

The **CLAWS7 Tagset**² (Wynne, 1996) – fine granularity. About two hundred tags are defined which provide more information that the above tagsets. For example, there are (excluding pronouns) twenty types of nouns defined, compared to ten in the Brown Corpus Tagset and only four in the Penn Treebank tagset.

The **ENGCG Tagset**¹ (Karlson, 1990) – differs from the above tagsets. Words are assigned a sequence of tags, each of which describe a different property.

Even though the Penn Treebank tagset provides the least amount of information, it is the set I chose to use in this system. The reasons for choosing this tagset are: (1) many large well-known corpora and natural language processing tools employing this tagset are available, (2) this tagset provides enough information to achieve a partial parse, and (3) a smaller tagset is easier to incorporate into the main component of the system (the finite

¹ The Brown, Penn Treebank, and ENGCG tagsets are available in van Halteren (1999).
² The CLAWS7 Tagset is available in Wynne (1996).
state automata). Appendix A lists the complete set of Penn Treebank tags. For a thorough description of the Penn Treebank tags and their usage, refer to Santorini (1995).

2.3 Stochastic

Statistical tagging techniques are centered around the manipulation of probabilities gathered from a correctly tagged corpus. As an ongoing example, consider the pre-tagged corpus of three sentences and two test sentences in Figure 2.1.

<table>
<thead>
<tr>
<th>CORPUS</th>
<th>The/DT book/NN is/VBZ on/IN the/DT table/NN ./i.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>I/PRP need/VB to/TO book/VB a/DT flight/NN ./i.</td>
</tr>
<tr>
<td></td>
<td>I/PRP buy/VB a/DT book/NN ./i.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TEST SENTENCES</th>
<th>I need the book for class.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>I need to book a flight.</td>
</tr>
</tbody>
</table>

**Figure 2.1** Example Corpus - a pre-tagged corpus of three sentences along with two test sentences.

Several types of probabilities associated with a specific word (W) or tag (T) can be directly generated from this corpus. For example when W = *book* and T = *NN*, the following probabilities can be calculated:

- **Word Probability** that *book* appears in the corpus:
  \[ P(W) = P(\text{book}) = \frac{\# \text{occurrences of } \text{book}}{\text{number words}} = \frac{3}{16} = 0.19. \]

- **Joint probability** that *book* appears in the corpus and is tagged *NN*:
  \[ P(W \& T) = P(\text{book} \& \text{NN}) = \frac{\# \text{occurrences of } \text{book with } \text{NN tag}}{\text{number words}} = \frac{2}{16} = 0.13. \]
Conditional probability that book is tagged NN in the corpus:
\[ \text{P}(T \mid W) = \text{P}(NN \mid \text{book}) = \frac{\text{P(book} \& \text{ NN)} \div \text{P(book)}}{0.13} / 0.19 = 0.68 \]

Lexical Generation(Output) Probability: given the NN tag, what is the probability that book is assigned to it:
\[ \text{P}(W \mid T) = \text{P}(\text{book} \mid NN) = \# \text{ times book is NN} / \# \text{ times NN appears} = 2/4=0.5 \]

Based on the conditional probability alone, we could estimate that the word book in the test sentences in Figure 2.1 can be assigned the NN tag, since it has the highest probability. This is called the Maximum Likelihood Estimator (MLE). As you can see this approach will only be correct for the first test sentence. Probabilities of the words in the surrounding context can be added to increase accuracy. For example, in our corpus, nouns (with the NN tag) are never preceded by to (TO):

The Bigram model: the conditional probability \( \text{tag}_i \) follows \( \text{tag}_{i-1} \):
\[ \text{P}(\text{tag}_i \mid \text{tag}_{i-1}) = \# \text{ times } \text{tag}_{i-1} \text{ is followed by } \text{tag}_i / \# \text{ times } \text{tag}_{i-1} \text{ occurs} \]
\[ \text{P}(NN \mid TO) = \# \text{ times NN is followed by TO} / \# \text{ times TO occurs} = 0 / 1 = 0.0 \]

Trigrams \( \text{P}(\text{tag}_i \mid \text{tag}_{i-2} \text{ tag}_{i-1}) \) – could also be used to generate the probability of a tag given the two previous tags. These are known as \( n \)-gram models, where \( n \) is the number of terms in consideration.

The ultimate goal here is to use these readily available probabilities to estimate the maximum probability of a sequence of tags, given a sequence of words:

\[ \text{P}(T_1, ..., T_n \mid W_1, ..., W_n). \]

This problem cannot be directly computed, but with the above probabilities, Bayes' formula can be used to estimate it indirectly:
\[
P(T_1, ..., T_n \mid W_1, ..., W_n) = \frac{P(T_1, ..., T_n) \times P(W_1, ..., W_n \mid T_1, ..., T_n)}{P(W_1, ..., W_n)}
\]

where \(P(T_1, ..., T_n) = \prod_{i=1}^{n} P(T_i \mid T_{i-1})\) and \(P(W_1, ..., W_n \mid T_1, ..., T_n) = \prod_{i=1}^{n} P(W_i \mid T_i)\).

Since this formula is being used to compare the likeliness of possible sequences of tags, the denominator can be dropped since it does not add any useful information. The numerator can be represented graphically as a Hidden Markov Model (HMM). To accomplish this, the bigram probabilities are first modeled in a transition network, where each node represents a tag and an arc indicates the probability that one tag follows another. This model is called a Markov chain. The Markov chain for the corpus in Figure 2.1 is shown in Figure 2.2. A path through the network can be compared to a sequence of tags.

![Figure 2.2](image)

**Figure 2.2** The Bigram Probabilities of the Example Corpus in Figure 2.1 represented as a Markov Chain. \(S\) is the start of a sentence marker.

Output probabilities (the Lexical Generated Probabilities) are then assigned to each node in the network to complete the HMM – Figure 2.3. For a sequence of tags, a path can be
taken through the network and the bigram and output probabilities multiplied together along the way - which generates a probability for the tag sequence. Since multiplication is commutative, the sequence of multiplications can be re-written to closely resemble the numerator of the Bayes formula.

*Hidden* is used because, for a specific sequence of words, it is not clear what state the Markov Model is in. For example, the word *book* could be generated from state *NN* with a probability of 0.67 or from state *VB* with a probability of 0.33 – which transition was taken is hidden. However, the probability produced by a sequence of words can easily be computed by multiplying the arc probabilities with the output probabilities.

![Diagram of Hidden Markov Model](image)

**Figure 2.3** The Hidden Markov Model generated from the Example Corpus in Figure 2.1.
Now let us re-visit the second test sentence above, in which book is ambiguous, using this HMM. book could be either a NN or VB, so for I need to book a flight, determine which sequence of tags is more likely. Multiplying only the bigram probabilities together yields:

\[ \text{PRP VB TO NN DT NN} = 0.67 \times 1.0 \times 0.33 \times 0.0 \text{(no transition)} = 0.0 \]

or

\[ \text{PRP VB TO VB DT NN} = 0.67 \times 1.0 \times 0.33 \times 1.0 \times 0.67 \times 1.0 = 0.148 \]

These probabilities are then each multiplied with the output probabilities of I need to book a flight to complete the numerator of the Bayes rule. However, at this point it is already clear that the second sequence of tags will be chosen, since the first yielded a probability of zero.

In the example above, the most likely of two patterns was chosen for I need to book a flight. This was to illustrate the use of the Markov model. An important aspect of the Markov model is that all possible sequences need not be enumerated. Only the most likely sequence at any given point needs to be observed. Starting from the beginning of the sentence, you step through each word, choosing only the most likely sequence for each ending word. This is known as the Viterbi Algorithm (Viterbi, 1967).

A common problem with stochastic approaches occurs when a low occurrence of data (or absence of data) causes false estimations — the sparse data problem. The worse case is the absence of a word or pattern, which forces the overall probability to zero. Smoothing techniques are usually employed by statistical taggers to avoid bad estimations when rare words or tag sequences appear. For example, the absence of the TO NN bigram in our corpus (which caused the overall estimate to become zero) could be considered a sparse
data problem. Even though it does not occur in our corpus, does not mean it is an impossible sequence, for example: *Closer to/TO home/NN, the survey produced more useful information*. To avoid these zero probabilities, a small weight could be added to the bigram probabilities, which ensures a non-zero overall estimation. More smoothing strategies are discussed in the implementations below.

In the stochastic approach, the *information is captured in tables of probabilities*. When trained on a large corpus, these tables become very large and difficult to work with. Another draw back to this approach is that linguistic knowledge captured by these probabilities is not obvious to a user.

**CLAWS1** (Constituent Likelihood Automatic Word-tagging System - Garside et al., 1987) and a **Parts Program** (Church, 1988) are two of the first well-known statistical based systems, recognizing the potential of tagging words based primarily on statistical information. These systems implement an *open* Markov model. The transitions between the tags and their frequencies are given explicitly, as opposed to HMMs where the transitions are kept hidden and only the lexical items are revealed.

**TnT** (*Trigram’n’Tags* – Brants, 2000) is an example of a stochastic tagger which uses the HMM model described above. As the name suggests, trigrams are used instead of bigrams to capture contextual information. Sparse data really becomes a problem when using trigrams. To avoid zero probabilities, TnT’s smoothing paradigm consists of a linear interpolation of unigrams, bigrams and trigrams:

\[
P(T_i | T_{i-2} T_{i-1}) = \lambda_1 * P(T_i) + \lambda_2 * P(T_i | T_{i-1}) + \lambda_3 * P(T_i | T_{i-2} T_{i-1})
\]

where \( \lambda_1 + \lambda_2 + \lambda_3 = 1 \)
The values of the $\lambda$s are context-independent, and can be generated from a corpus using uni-, bi- and trigram information.

Unknown words are given tag probabilities based on their suffixes. Starting with a suffix of length $m$, a linear interpolation of the frequency probabilities of this suffix is generated by successively shortening the suffix. For example, the probability of words ending in “ing” to have tag T would be generated by:

$$P(T \mid \text{“ing”}) = (P(T \mid \text{“ing”}) + \theta_1 * P(T \mid \text{“ng”}) + \theta_2 * P(T \mid \text{“g”}))/\theta_1 + \theta_2$$

The values of the $\theta$s are also context-independent, but the size of $m$ depends on the word in question. Refer to (Brant, 2000) for more details on choosing values for $\lambda$, $m$, and $\theta$. Capitalization is also used in determining unknown words, since English only capitalizes proper nouns (along with other words that start sentences).

The Xerox tagger (Cutting et al., 1992) is another stochastic tagger based on the HMM model. The Xerox tagger associates a probability with a word’s ambiguity class — the set of possible tags of the word. The ambiguity classes alone are first used to train a HMM, using the Baum-Welch algorithm (Baum, 1972). These sequences are then disambiguated by computing the maximal path through the HMM with the Viterbi algorithm. Unknown words are assigned probabilities based on suffix information, and unknown tag sequences are assigned small, non-zero values.

2.4 Rule-Based

Unlike stochastic taggers, rule-based taggers attempt to capture contextual information with a set of meaningful rules. For example, a contextual rule could be:
*IF* the current word is tagged as a verb *AND* the preceding word is a determiner

*THEN* change the current word's tag to a noun.

In the first rule-based systems, linguistic rules were hand written. This procedure is not only very time consuming, but requires much linguistic knowledge. TAGGIT (Greene and Rubin, 1971) was one of the first of such systems. Words were initially assigned a tag or a set of tags from a lexicon. There were about 3,300 rules, which were created manually from an empirical analysis of the Brown Corpus (Francis and Kucera, 1982). A local context window of five words was used.

Hindle (1989) was the first to attempt to automatically acquire rules from a corpus. After developing a system of hand written rules for over two years, he developed an algorithm which automatically generated a rule set that reduced the error rate by 50% when compared to the hand written rule set. Unlike the transformation-based tagger described in the next section, this approach generates a very large set of rules. For more information on how the rules were acquired, refer to Hindle (1989).

Roche and Schabes (1995) show that rule-based taggers can be implemented very efficiently with finite-state transducers. They describe how each rule from Brill's tagger (Section 2.4.1) can be represented as a non-deterministic finite-state transducer. These non-deterministic finite-state transducers can then be made deterministic, combined into a single transducer and represented in their minimal form. Even the lexicon is represented as a finite state automaton, reducing both look up time and space requirements.
2.4.1 Transformation-Based Rule Learning

A major revitalization of rule-based approaches came in 1992, when Eric Brill (1992) showed that a rule-based system could achieve performance comparable to state-of-the-art stochastic taggers. Furthermore, this high performance was achieved with a small rule set that was automatically acquired from a tagged corpus using Transformation-Based Error-Driven Learning (TBL – Ngai and Florian, 2001; Daelemans, 1999; Brill, 1995; Brill, 1994; Brill, 1992). Figure 2.4 shows how the TBL strategy works.

![Figure 2.4 Transformation-Based Error-Driven Learning Strategy.](image)

First, the text is annotated by some sort of initial state annotator. This could be anything from a stochastic tagger to Maximum Likelihood Estimates (see Section 2.3). This annotated text is then compared with the same text which has been manually annotated. An ordered list of transformations is learned which are then applied to the output of the initial state annotator to get closer to the manually tagged text. Given a set of possible
transformations, a greedy search is performed to find which transformation brings the initial annotator's output closer to the manually tagged text.

An **objective function** needs to be defined to choose the best transformation. For example, the objective function could be: (the number of corrections a rule makes) minus (the number of errors it causes). The transformation with the highest objective function score is then added to the ordered list. A transformation consists of: a *rewrite rule* and a *triggering environment*. For example, a rewrite rule could be *change tag from modal to noun* where the triggering environment is *the preceding word is a determiner*. To define a specific TBL system, one must specify the:

1. initial state annotator
2. possible transformations
3. objective function for choosing a transformation

**Brill's tagger** is a rule-based part-of-speech tagger. A version of this tagger, that has been trained using TBL on the Penn Treebank (Marcus et al., 1993), can currently be downloaded from Eric Brill's web site (http://www.cs.jhu.edu/~brill). The initial state annotator of this system assigns each word its most likely tag as indicated in a training corpus. In the lexicon, the most frequent POS tag along with other possible tags are listed for each word in the corpus. For example the lexical entry for *book* is

```
book NN VB
```

where NN is the most likely tag and VB is another possible tag. Two sets of transformations (rules) are then considered to complete tagging:
1) **lexical rule set** – determines tags of unknown words

2) **contextual rule set** – contextual rules which alter the tagging of a word tagged X to Y if:
   
   a) the word is not seen in the corpus OR
   
   b) the word was seen tagged Y in the corpus at least once

If a word is not in the lexicon, it is assigned the *NN* tag (noun), or *NNP* tag (proper noun) if it is capitalized. This is sort of a starting point for the lexical rules to try to determine the actual part-of-speech of the unknown word. The following templates are used by the lexical transformation rule set:

Change the tag of an unknown word (from X) to Y if:

1. Deleting the prefix(suffix) x, |x| ≤ 4, results in a word
2. The first (last) (1,2,3,4) characters of the word are x
3. Adding the character string x as a prefix (suffix) results in a word (|x| ≤ 4)
4. Word W ever appears immediately to the left (right) of the word
5. Character Z appears in the word

The actual syntactic tokens used in Brill’s tagger to represent each of these rules are *deletpref* (*deletesuf*), *haspref* (*hassuf*), *addpref* (*addsuf*), *godleft* (*goodright*), and *char*, respectively. These rules assign an unknown word a tag regardless of its current tag.

More lexical transformations are defined which are identical to the ones above except
they are only taken if the word currently has a specific tag. The syntactic tokens are also the same as above, except the prefix “f” is added.

Once the lexical rules have attempted to determine the part-of-speech tags for unknown words, the contextual rules are considered to improve accuracy. The following templates are used for the contextual transformation rule set:

**NON-Lexicalized**(only allowed to reference tags)

Change tag a to b when:

1) The preceding (following) word is tagged z
2) The word two before (after) is tagged z
3) One of the preceding (following) words is tagged z
4) One of the three preceding (following) words is tagged z
5) The preceding word is tagged z and the following word is tagged w
6) The preceding word (following) word is tagged z and the word two before (after) is tagged w.

**Lexicalized**(allowed to reference both words and tags)

Change tag a to b when:

1) The preceding (following) word is w
2) The word two before (after) is w
3) One of the two preceding (following) words is w
4) The current word is w and the preceding (following) word is x
5) The current word is w and the preceding (following) word is tagged z
The actual syntactic tokens used by Brill's tagger for each of these transformations are obvious in meaning, so I am not going to restate them here.

Both the lexical and contextual rules are located in text files which can be modified by a user. Once the tagger has been downloaded, it can be used as is, or re-trained on whatever corpus you are working with. Documentation is provided with the tagger that explains the training procedure in detail. Since the tagger was trained on the Penn Treebank (1.1 million words), it delivers fairly good results without having to re-train.

### 2.5 Other Approaches

Several different approaches to part-of-speech tagging are presented in the sub-sections below. The intention is to provide a general overview of the other tagging strategies that have been attempted.

#### 2.5.1 Neural Networks

The most popular neural network is the *Multilayer Perceptron* (*MLP* - Rumelhart et al., 1986). The network is composed of an input and output layer, and one or more intermediate or *hidden* layers. Each layer is represented with an array of *units*. Each unit has a *weight* (usually a real number) and an *activation* (usually 0 or 1) associated with it.
Each unit is connected to all units in an adjacent layer. Figure 2.5 depicts a simple three-level neural network\(^2\) that could be used for part-of-speech tagging.

![A Simple Neural Network Architecture that could be used in Part-of-speech Tagging.](image)

Initially the input layer can be assigned random weights. The values of each hidden unit \(i\) are computed by \(\Sigma_i a_i * w_{ij}\) where \(a_i\) are the incoming activation values to \(i\) and \(w_{ij}\) are the incoming weight values from the input layer. The output layer values are calculated in a similar fashion. An approximation of the desired weights can be found by: (1) repeatedly presenting input patterns to the network, (2) calculating the output values, and (3) then adjusting the weights so that error is suppressed. Back Propagation is a popular learning strategy used to adjust the weights. The actual output values are compared against the desired output values. The error between the actual output and the desired output is back propagated to the hidden and input units, strengthening weights which should have contributed more to the desired output and weakening those which contributed too much to the error. For example, consider Figure 2.5 again, in which we

---

\(^2\) Figure 2.5 was obtained from chapter 17.5 of van Halteren (1999)
only have three possible tags. Suppose we are trying to determine the part-of-speech tag of a word \(i\) in a sentence, and the network has already been trained to recognize that a word should be tagged noun if it follows an article and precedes a verb. If \(\text{word}_{i-1}\) is tagged A (article) and \(\text{word}_{i+1}\) is tagged V (verb), then the output unit corresponding to N (noun) should be given the highest activation. If another unit is activated, an error is back propagated through the network, so that the next time this pattern is presented to the network, the actual output would be closer to the correct output. The contextual rules are thus captured in the weights of the neural network.

Net-Tagger (Schmid, 1994a) is a MLP part-of-speech tagger which delivers results comparable the those of state-of-the-art statistical taggers. The network consisted of an input and an output layer (no hidden layers). Three preceding words, the focus word and the two following words were encoded in the input. As in Figure 2.5, each word requires a number of units corresponding to the number of possible tags. This yielded about 240 input units and 40 output units — since the Penn Treebank Tagset (aprox. 40 tags) was used. Lexical probabilities were assigned to the focus word and the two following words in the input layer. Since the preceding words have already been tagged, the activation output values (the tags that have already been determined by the network) are assigned to the preceding words. The probability lexicon and unknown word guesser are based on a system by Cutting et al. (1992).

Elastic-Neuro Tagger (Ma et al., 1999) performs part-of-speech tagging with variable lengths of context. This tagger is a 3-layer perceptron and achieves a high accuracy (94.4%) for tagging ambiguous words in small Thai corpus, where data is sparse. In this approach, the network is first trained using only the focus word input units.
A new perceptron is then created by incrementally growing the context in the input layer, and the new network is then re-trained. Figure 2.6 depicts this growth. The solid lines represent the original perceptron, and the dashed line show the input elements that are incrementally added until a desired number of words in the left and right context is reached.

![Diagram of the Elastic Neural Network]

**Figure 2.6** The Elastic Neural Network. The solid lines represent the original perceptron, and the dashed line show the input elements that are incrementally added.

### 2.5.2 Hybrid Environments

van Halteren et al. (1998) combined four different taggers which achieved better results than any one of the particular taggers. These taggers were based on HMMs (Section 2.3), Transformation Rules (Section 2.4.1), Memory-Based (Section 2.5.4), and Maximum Entropy (Section 2.5.6). Since the errors caused by each of these tagging approaches are somewhat un-correlated, a typical error made by one of the systems can be voted out by the others. There are several ways to combine the results of each individual tagger. *Simple voting* could be used – the tag with the most votes is chosen (In the case of a tie,
the winner was randomly chosen). Precision and recall information\(^1\) can be also used to weight the vote of each tagger. *Pairwise, Memory Based, and Decision tree* voting are also described by van Halteren et al. (1998). Each voting strategy out performed the individual taggers.

Tapanainen and Voutilainen (1994) discuss the combination of the rule-based EngCG tagger and the statistical Xerox tagger. EngCG is first used to resolve some ambiguities, and the Xerox tagger attempts to resolve remaining ambiguities.

**TAKTAG** (Lee et al., 1995) combined statistical and rule-based methods to tag Korean. A morphological analyzer first segments out the constitutional morphemes of the input text and assigns initial POS tag from a dictionary. A HMM tagger then takes the sequence of morphemes with initial tags and searches for the maximum probability tag sequence. A rule-based error corrector adopted from Brill (1992) then attempts to correct errors made by the HMM tagger due to the complex morphological structure of Korean.

Ma et al. (2000) combined a Neuro tagger (Section 2.5.1) and a rule-based tagger to tag Thai. The tagger achieved high performance when trained on only a small Thai corpus of 10K words. A Neuro tagger is first used to tag new sentences. A set of transformation-based rules are then employed, which reduces the error rate by almost 20%. This method performed much better than a HMM approach.

**WOTAN** (Bergman, 1994) combined HMM and Memory-Based approaches to tag Dutch. WOTAN was trained on the Eindhoven corpus. The main component is an HMM, and Memory-Based techniques are used to determine unknown words.

\(^1\) Gathered during training, precision measures which percentage of the tokens tagged X by a tagger are also tagged in the test corpus, and recall measures which percentage of the tokens tagged X in the test corpus are also tagged X by the tagger.
2.5.3 Support Vector Machines

Support Vector Machines (SVM) – see Cristianini and Shaw-Taylor (2000) for an introduction to support vector machines - have been used in several real-world applications, such as text categorization, hand-written character recognition, image classification, bio-sequences analysis, and, of course, part-of-speech tagging (Nakagawa et al., 2001; Gimenez and Marquez, 2003). SVMs are a form of a supervised machine learning algorithm for binary classification on feature vector space $x \in \mathbb{R}^l$. Consider the following hyperplane:

$$w \cdot x + b = 0, \quad w \in \mathbb{R}^l, b \in \mathbb{R}$$

The training data is defined as $\{(x_i,y_i) | x_i \in \mathbb{R}^l, y_i \in \pm 1, 1 \leq i \leq l\}$. Suppose that the hyperplane separates the training data into two classes such that:

$$y_i \cdot (w \cdot x_i + b) \geq 1$$

While several such separating functions exist (Figure 2.7, left side), SVMs find the optimal hyperplane that maximizes the margin in between hyperplane and nearest points (Figure 2.7, right side).
In an attempt to illustrate how this idea can be used to tag parts-of-speech, consider a training example \((x_i, y_i)\) where \(x_i\) is a vector of three features and \(y_i\) indicates if this feature space categorizes a noun \((NN \rightarrow a -1 value)\) or verb \((VB \rightarrow a +1 value)\):

\[ x_i = [\text{POS Tag Context} ; \text{Word Context} ; \text{Lexical Clues}] \]

\[ y_i = -1 \ (NN - a\ noun) \]

This training example could be plotted in a three dimensional figure (similar to the two dimensional graph in Figure 2.7) along with many other training examples from a corpus. The SVM would then learn the best separation (hyperplane) between features that indicate a noun is present and features that indicate a verb is present.

For linearly non-separable cases, feature vectors are mapped into a higher dimensional space by a nonlinear function. Note, however, that the basic SVM approach is only capable of distinguishing between two part-of-speech tags. A one-versus-rest (Weston and Watkins, 1999) approach can be taken to extend the SVM to classify \(k > 2\) part-of-speech tags by training \(k\) classifiers that say whether a feature space \(x\) belongs to \(tag_i, 1 \leq i \leq k, \) or \(tags_{1..i-1,i+1..k}..\)
2.5.4 Memory(Example)-Based

In a memory-based approach, new examples are compared with a set of previously encountered ones, as opposed to a set of rules which have been formed from previous examples. Each example has a label and a set of features. During training, the examples are incrementally presented to a classifier and added into memory. The distance between the elements in memory and a new element is determined with a similarity metric. The similar metric could merely be an overlap metric, where the number of common features are counted. Weights can also be used to introduce relevance among the features.

Memory-Based Tagging (MBT - Daelemans and Zavrel, 1996) is an implementation of this example-based approach. A tagger is created by extracting a lexicon, a known case base, and an unknown case base from a tagged corpus. During tagging, new words are referenced in the lexicon, and separated into known and unknown words. They are then retrieved from either the known or the unknown word case bases. Cases for known words consist of weighted information about the focus word and one word of right context, two left disambiguated words, and a corresponding category for the focus word. Cases for unknown words consist of three suffix letters, one prefix letter, one left disambiguated word, and one right context word (these features are also weighted). MBT suffers from both space and time complexity. However, Daelemans and Zavrel (1996) describes how IGTrees can be used to reduce memory requirements by 95 percent and case retrieval time by 100 to 200 percent.
2.5.5 Decision Trees

The Decision Tree part-of-speech tagging approaches (Daelemans, 1999; Marquez and Rodriguez, 1997) extract tree structures from training examples which are used to tag new examples. Internal nodes in the tree structure represent a test that is administered to a new example and the arcs represent the possible answers. So for a new example, a series of tests are administered and the answers to the tests form a path down through the tree until a leaf node that suggests the possible part-of-speech tag is reached. The decision tree is formed by repeatedly dividing the test example into categories of similar features.

A Probabilistic Decision Tree (Breiman et al., 1984) can be formed by having multiple part-of-speech categories (each with an assigned probability) in a single leaf node. For example, a path through the tree could lead to a leaf node containing the IN tag with a probability of .90 and the WDT tag with a probability of .10. TREETAGGER (Schmid, 1994b) and SPATTER (Magerman, 1994) are both implementations of probabilistic decision trees used to choose the most likely sequence of decisions for assigning part-of-speech tags and for forming a parse tree, respectively.

2.5.6 Maximum Entropy

The Maximum Entropy model has been used in several other areas of Natural Language Processing research besides part-of-speech tagging, including: language modeling (Lau et al., 1993), machine translation (Berger et al., 1996), prepositional phrase attachment (Ratnaparkhi et al., 1994), and word morphology (Della Pictra et al., 1995). Maximum Entropy part-of-speech taggers (Toutanova and Manning, 2000; Ratnaparkhi, 1996) are similar to statistical taggers presented earlier in Section 2.3. The model assigns a
probability for every tag $t$ in the set $T$ of possible tags given a word and its context $h$, which is usually defined as the sequence of several words and tags preceding the word. This model can be used for estimating the probability of a tag sequence $t_1...t_n$ given a sentence $w_1...w_n$:

$$P(t_1...t_n \mid w_1...w_n) = \prod_{i=1..n} P(t_i \mid t_1...t_{i-1}, w_1...w_n) = \prod_{i=1..n} P(t_i \mid h_i)$$

As with the HMM taggers in Section 2.3, tagging is the process of assigning the most likely tag sequence to a string of words. However, in the maximum entropy framework it is possible to easily define and incorporate much more complex statistics, and not be restricted to n-gram sequences. Under the Maximum Entropy formalism, the goal is to maximize the entropy of the distribution subject to the constraints in the training data.

### 2.6 Comparing Part-of-speech Taggers

Determining which method performs better is not a simple task. First of all, different systems use different part-of-speech tagsets. When one tagger has assigned an incorrect tag, another tagger could assign a more ambiguous tag. Second, linguists themselves tend to disagree on some part-of-speech tags. One linguist may find an error where another would say there is none. Third, even though one tagger may perform slightly better than another, the latter may be much easier to work with or modify. So there is a trade off between performance and usability.

Samualsson and Voutilainen (1997) compared the performance of the EngCG tagger (Section 2.2) and a state-of-art statistical tagger based on Church (1988). They report the
EngCG tagger performing far better than the statistical tagger. Skeptics argued that these results were due to: (1) ambiguities that are left in the tags by EngCG, (2) simplicity of the EngCG tag set, and (3) compromised integrity of the experiments. After a re-evaluation of the two taggers, Samualsson and Voutilainen (1997) argue that this is not the case. They conclude that the better performance of the EngCG tagger can be attributed to better lexical and contextual resources available to the EngCG system.

Zavrel and Daelemans (1999) recently compared the performance of seven different taggers when run on a Dutch corpus. The taggers were the D-Tale (a rule-based tagger), Dutchtable/PAROLE (a hybrid system – HMM and rule-based), Xerox (Section 2.3), Brill (Section 2.4.1), KEPER (a bigram tagger), CORRie (a trigram HMM tagger), WOTAN (Section 2.5.2), Memory-Based (Section 2.5.4), MXPOST (Maximum Entropy tagger), and TnT (Section 2.3). Each tagger was given the same task to complete, under the same conditions. They report TnT delivering the best results – a HMM model with a good smoothing technique for the handling of unknown words.

Charniak et al. (1996) compared the performance of various statistical tagging approaches to see which approach was best suited to provide part-of-speech information for a parser. Their models also included assigning multiple tags to each word – where each tag of a particular word is assigned a likelihood probability. They concluded that a single-tag Markov-model tagger was best suited for a parser. Assigning multiple tags to each word did not improve parsing accuracy. They also discovered that parsing accuracy does not increase when a parser assigns part-of-speech information itself.
2.7 Summary and Conclusions

Part-of-speech tagging has been the target of research for decades now. Many different approaches have been taken to solve this problem. First only manually created rule-based systems were available. Then with the advent of large manually tagged corpora, statistical models became the dominant approach. Finally with the development of transformation-based error-driven learning, there has been a renewed interest in rule-based systems. Currently, Statistical and Rule-based (generated with transformation-based learning) approaches are the most commonly used systems, both yielding similar state-of-the-art accuracies (96-97%). Several other approaches have also been attempted and are capable of producing state-of-the-art accuracies, including: Neural Network, Hybrid, Support Vector Machines, Decision Trees, Maximum Entropy, and Memory-based approaches.

Table 2.1 lists the part-of-speech tagging approaches that have been presented. Even though the methods of storing information differs dramatically from system to system, each approach uses supervised learning techniques to acquire lexical and contextual information from a correctly tagged corpus.

<table>
<thead>
<tr>
<th>Method</th>
<th>Information stored as</th>
</tr>
</thead>
<tbody>
<tr>
<td>Statistical (HMM)</td>
<td>tables of probabilities</td>
</tr>
<tr>
<td>Rule-Based</td>
<td>small intuitive linguistic rules</td>
</tr>
<tr>
<td>Neural Networks</td>
<td>weights and connections</td>
</tr>
<tr>
<td>Support Vector Machines</td>
<td>multiple binary classification hyperplanes</td>
</tr>
<tr>
<td>Maximum Entropy</td>
<td>tables of statistics and constraint functions</td>
</tr>
<tr>
<td>Decision Trees</td>
<td>nodes and arcs of a tree structure</td>
</tr>
<tr>
<td>Memory-Based</td>
<td>a collection of observed examples</td>
</tr>
<tr>
<td>Hybrid</td>
<td>a combination of the above methods</td>
</tr>
</tbody>
</table>
The primary advantage of supervised part-of-speech taggers is that they can automatically extract information from a pre-tagged corpus. This, however, could also be viewed as its primary disadvantage since one must have a large correctly tagged corpus in order to produce a tagger. The state-of-the-art tagging accuracies are achieved only when tested on a section of a corpus from which it has been generated. Fortunately these taggers still perform relatively well when used on sentences for which it has not been trained. However, many real-world texts, like the New York Times, Encyclopedia Encarta and the Wall Street Journal, contain sentences that average 25-30 words in length. If a tagger is achieving say a 95% accuracy then, on average, there would still be an error made in every sentence which could significantly degrade the performance of a secondary system (like a (partial) parser) that relies on part-of-speech tags. See Section 8.3 – in the Evaluation Chapter – for a more detailed discussion on the types of part-of-speech tagging errors that were encountered during the implementation of this system.

Future advances in part-of-speech tagging with hopefully provide us with a tagger that is very accurate across multiple domains without the need for re-training. This tagger would definitely enhance the practical value of any system that relies on part-of-speech tags.
CHAPTER 3

THE SYNTACTIC RELATION SET

...one result of the formal study of grammatical structure is that a syntactic framework is brought to light which can support semantic analysis.

- Noam Chomsky (1957)

Each syntactic relation is described here in detail, offering example sentences that illustrate their usages and boundaries. The syntactic relations are compared against the gold-standard constituents (hereafter referred to as the gold-set) set forth in the Penn Treebank III project (Bies et al., 1995; Marcus et al., 1994). The syntactic relations are broken down into four groups corresponding to the four networks of automata that identify them. The syntactic relations of the CASS system are also described in some detail. CASS is the implementation of the easy-first partial parsing approach and is compared against the larger-first approach in Chapter 8. The complete set of syntactic relations is shown in Table 3.1, organized by the network that identifies them. Within the comma and conjunction networks, the syntactic relations have been further categorized.

Table 3.1 The complete set of syntactic categories organized by the network that identifies them.

<table>
<thead>
<tr>
<th>Table 3.1 The complete set of syntactic categories organized by the network that identifies them.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pre-processing</strong></td>
</tr>
<tr>
<td>PSUB</td>
</tr>
<tr>
<td>PIN</td>
</tr>
<tr>
<td><strong>Comma Network</strong></td>
</tr>
<tr>
<td>CO-DIR</td>
</tr>
<tr>
<td>CO-IDIR</td>
</tr>
<tr>
<td><strong>Series Commas</strong></td>
</tr>
<tr>
<td>CO-LST-(INF VC NP)</td>
</tr>
<tr>
<td><strong>Clausal Commas</strong></td>
</tr>
<tr>
<td>CO-VC</td>
</tr>
<tr>
<td>CO-INF</td>
</tr>
<tr>
<td>CO-REL</td>
</tr>
<tr>
<td>CO-SUB</td>
</tr>
<tr>
<td>CO-RSUB</td>
</tr>
<tr>
<td><strong>Enclosing Commas</strong></td>
</tr>
<tr>
<td>CO-TNP</td>
</tr>
<tr>
<td>CO-APS</td>
</tr>
<tr>
<td>CO-TRN</td>
</tr>
<tr>
<td>CO-PP</td>
</tr>
<tr>
<td>CO-S</td>
</tr>
<tr>
<td>CO-NP</td>
</tr>
<tr>
<td><strong>Conjunction Network</strong></td>
</tr>
<tr>
<td>CC-REL</td>
</tr>
<tr>
<td>CC-SUB</td>
</tr>
<tr>
<td>CC-INF</td>
</tr>
<tr>
<td>CC-ING</td>
</tr>
<tr>
<td>CC-VS</td>
</tr>
<tr>
<td>CC-VC</td>
</tr>
<tr>
<td><strong>Phrasal Conjuncts</strong></td>
</tr>
<tr>
<td>CC-PP</td>
</tr>
<tr>
<td>CC-CNP</td>
</tr>
<tr>
<td>CC-NP</td>
</tr>
<tr>
<td><strong>Clause Network</strong></td>
</tr>
<tr>
<td>SUB</td>
</tr>
<tr>
<td>INF</td>
</tr>
<tr>
<td>REL</td>
</tr>
<tr>
<td>ING</td>
</tr>
<tr>
<td><strong>Phrase Network</strong></td>
</tr>
<tr>
<td>PP</td>
</tr>
<tr>
<td>TNP</td>
</tr>
<tr>
<td>NP</td>
</tr>
<tr>
<td>VP</td>
</tr>
<tr>
<td>ADV</td>
</tr>
<tr>
<td>ADJ</td>
</tr>
</tbody>
</table>
This is not intended to be a complete set of syntactic relations for the English Language, but it represents over 99% of the syntactic relations that were observed in the above sources. For example, a list of subordinate clauses is definitely a possible syntactic relation, but is not recognized here since it occurred so infrequently (less than .1% of lists) in the above sources. However, a new syntactic relation can easily be added by including an automaton in the cascade which recognizes it.

3.1 Pre-Processing

As a pre-processing step to the rest of the automata, phrases that are commonly used as prepositions and subordinate conjunctions are identified first. These phrases are not explicitly recognized in the gold-set and is done here merely as a simplification step for the automata that recognize prepositional phrases and subordinate clauses later in the cascade. Some common phrasal prepositions that are recognized include: such as, according to, along with, apart from, and with the exception of; and some phrasal subordinate conjunctions that are recognized include: even if, as though, even though, rather than, and so that. The complete set is given in Chapter 5.

3.2 Commas

Syntactic relations associated with commas are identified in the comma network. The focus of this larger-first approach is to handle large sentences with multiple commas. A well-defined set of automata has therefore been developed to identify these syntactic relations (see Chapter 5).
As shown in Figure 1.3 in the Chapter 1, commas are abundant in written text, but how exactly are these commas being used? A manually *comma-tagged* corpus of about 250K tokens (about 15 thousand commas) was compiled from articles randomly taken from all of the above mentioned sources. Table 3.2 summarizes the frequencies of the types of syntactic relations that are being represented with commas. This is an indication of how commas are used in an arbitrary written text. Almost 69% of the time a comma either coordinates items in a series, or delimits an apposition, prepositional phrase or relative clause. However, there is no clear-cut most likely syntactic category for commas.

Commas were also used to introduce the year part of a date (0.9%), coordinate adjectives in noun phrases (2.0%), and delimit adverbs (3.5%). Automata have not been explicitly defined to identify the syntactic relations associated with these commas: commas in dates and coordinating adjectives are recognized in the *time noun phrase* and *noun phrase* automata, respectively; and a sequence of one or more adverbs can be readily identified without using surrounding comma information if it is present. In total, this accounts for 99.3% of the commas in the corpus. The remaining 0.7% were used incorrectly or very infrequently.

**Table 3.2 Frequencies of Syntactic Relations Associated with Commas**

<table>
<thead>
<tr>
<th>Relation</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>CO-LST-(VC INF NP)</td>
<td>19.8%</td>
</tr>
<tr>
<td>CO-APS</td>
<td>16.5%</td>
</tr>
<tr>
<td>CO-TNP</td>
<td>0.7%</td>
</tr>
<tr>
<td>CO-TRN</td>
<td>3.2%</td>
</tr>
<tr>
<td>CO-PP</td>
<td>16.1%</td>
</tr>
<tr>
<td>CO-NP</td>
<td>3.1%</td>
</tr>
<tr>
<td>CO-VC</td>
<td>1.1%</td>
</tr>
<tr>
<td>CO-REL</td>
<td>16.2%</td>
</tr>
<tr>
<td>CO-SUB</td>
<td>7.4%</td>
</tr>
<tr>
<td>CO-INF</td>
<td>0.3%</td>
</tr>
<tr>
<td>CO-RSUB</td>
<td>0.3%</td>
</tr>
<tr>
<td>CO-DIR/CO-IDIR</td>
<td>2.7%</td>
</tr>
<tr>
<td>CO-S</td>
<td>5.5%</td>
</tr>
</tbody>
</table>
3.2.1 Speech Commas

Although not the focus of this research, direct and indirect speech (delimited with commas) are identified since commas play such a crucial role in their syntax. Direct speech is enclosed in quotation marks and can be placed before and/or after the speaker in the sentence, for example: [CO-DIR "After patrolling the neighborhood for hours," Officer Thompson noted [CO-DIR, "I felt it was entirely safe."]. A sentence containing indirect speech can have the same structure as the above example, except the quotation marks are omitted. Indirect speech, not delimited by commas, is not recognized here. For example, the indirect speech in the following sentence would be recognized as a complement clause later in the cascade: Officer Thompson said [SUB it was entirely safe]. Doran (1996) presents a detailed analysis of how various types of direct and indirect speech is punctuated in a sentence. She indicates that the comma which appears around direct speech is an important clue in identifying quoted speech. Such an in depth analysis, however, is not performed here. Commas delimiting speech are reduced into one of the two classes stated above. The gold-set identifies direct or indirect speech as S-TPC-#, where # is a reference number:

```
((SINV "
  (S-TPC-1 (NP-SBJ We)
    (VP have
      (NP (NP no useful information)
        (PP on
          (SBAR whether
            (S (NP-SBJ users)
              (VP are
                (PP-PRD at
                  (NP risk)))))))))
  "
  (VP said
    (S *T*-1))
  (NP-SBJ (NP James A. Talcott)
    (PP of
      (NP (NP Boston 's)
        Dana-Farber Cancer Institute)))) .)
```
The entire sentence is also recognized as an inverted sentence (S-INV) if the speech verb is placed before the speaker as in the example above. Indirect speech is represented in a similar fashion.

3.2.2 Series Commas

Commas are usually used to delimit a syntactic relation. In this case, the boundary of the syntactic relation is marked by a comma, the start of the sentence, or the end of the sentence. However, when commas are used to coordinate items in a series, a boundary needs to be established. This boundary is established by avoiding any type of attachment decisions (Guideline 1 – see Chapter 1). For example, *Thomas purchased the house in the country, the yacht with the 20 foot sail, and the sports car with the sunroof*. Only the highlighted words are included in the list of noun phrases. *With a sun roof* is not included in the list of noun phrases, because syntactically we cannot determine where it should be attached. In this case it should be attached to *the sports car*, but if it is replaced by *with the money*, then the attachment should occur at the verb. The Guidelines are not violated - *with the sunroof* can be attached within a sub clause (the list of noun phrases) or a peer (the verb). Note, however, that some attachment ambiguity has been contained. Attachment sites for *with the 20 foot sail* are limited to within the marked list of noun phrases. Also, it cannot be correctly determined where the list of noun phrases starts, but we know *the country* is definitely included in it. A close semantic analysis is needed to determine *house, yacht and car* are the three head nouns in coordination here.

CO-LST-VC is used to represent a series of verb clauses, for example: *Many foreigners [CO-LST-VC opened small businesses, joined fishing crews, or worked] as*
miners. No distinction is made between a list of gerund phrases and past participle phrases. The boundaries again are determined by following the guidelines. As miners is therefore not grouped in the list. CO-LST-INF is used to represent a series of infinitival clauses in a similar fashion.

Explicit categories are not defined in the gold-set for these syntactic relations. However, they are represented in terms of other pre-defined categories. For example: Thomas purchased [NP [NP the house in the country], [NP the yacht with the 20 foot sail], and [NP the sports car with the sunroof]]. The list of noun phrases is identified as a noun phrase which contains three noun phrases. Lists of verb and infinitival clauses are represented in a similar fashion.

3.2.3 Clausal Commas

A clausal comma is usually used to introduce/enclose a clause in a sentence, but it can also be used to conclude clauses that start a sentence.

CO-REL identifies a relative clause, as in: Henry III, [who succeeded Charles IX in 1574], feared the popularity of the Guise family. CO-SUB identifies a general subordinate clause that starts with a subordinate conjunction, as in: It made her the heir to the throne, [CO-SUB since George VI had no sons]. CO-VC identifies a coordinated verb clause that is enclosed by commas, as in: It is formed by the Mbomu and Uele rivers, [CO-VC and empties into the Congo]. CO-RSUB identifies a reduced subordinate clause with a present/past participle introductory verb, as in: [CO-RSUB By varying the refining processes], different kinds of asphalt may be obtained. Note that the corresponding non-comma-delimited syntactic relation in the clause network is ING
which could have multiple meanings. CO-RSUB’s meaning can be limited to reduced subordinate clauses here, since gerunds and complement clauses are not usually introduced by a comma. Finally, CO-INF identifies a purpose infinitival clause, as in Dewey believed that knowledge is a means of controlling the environment, [CO-INF hopefully to improve the quality of human life]. CO-INF’s meaning is restricted to purpose infinitival clauses since complement infinitival clauses are usually not introduced with a comma. The non-comma-delimited INF category in the clause network can be either a purpose or complement infinitival clause.

The gold-set identifies three types of infinitival clauses: Complement (Casey wants [S to throw the ball]), Purpose (Sue arrived early [S-PRP to get a good seat]), and Infinitival Relative (...a movie [SBAR to see]). Such detailed decisions are not made here since they require verb sub-categorization and semantic information.

Relative clauses are also categorized differently. In the gold-set, S-BAR is used for relative and subordinate clauses and RRC is used for reduced relative clauses in which no verb phrase is present, as in: ... title 110 not presently in the collection. Attachment of relative clauses by the gold-set in the Penn Treebank is indicated as follows:

(NP-SBJ (NP The person)
 (SBAR (WHNP-1 who)
  (S (NP-SBJ *T*-1)
   (VP threw
    (NP the ball))))))

where *T* is used to denote the movement of a constituent. S-BAR can take the following suffixes: SBAR-NOM (Nominal, marks free relatives: [SBAR-NOM What I really like] is chocolate); SBAR-ADV (Adverbial: You can leave [SBAR-ADV if you
really want to go]); SBAR-TMP (Temporal: Egg bread loses some zip [SBAR-TMP when the eggs come in 30-pound cans]). Finally, VP is used for CO-VC.

3.2.4 Enclosing Commas

Enclosing commas are used to enclose certain syntactic relations in a sentence. The boundaries of these syntactic relations are determined by commas and the beginning or ending of a sentence.

CO-APS identifies an apposition, as in: Catherine allied herself with Henry, [CO-APS the Duke of Guise ] . CO-TRN identifies transitional phrases. These phrases resemble prepositional phrases in syntax, but are often used to transition between sentences or ideas. Some transitional phrases are: in general, in addition, for example, and of course. CO-TNP identifies a time noun phrase that is enclosed in commas and starts a sentence. CO-PP identifies a prepositional phrase, as in: [CO-PP In the summer of 1971], Frank graduated from Harvard. CO-S introduces a new sentence or independent clause, as in: His early works often describe nature, [CO-S and the later ones describe the struggles and triumphs of the soul]. CO-NP identifies two noun phrases or prepositional phrases in coordination, as in: John was driving 100 mph, [CO-NP or 160 kph ], when he was pulled over.

The rationale for having unique comma categories for prepositional phrases, coordinate noun or prepositional phrases, and infinitival clauses (as well as the syntactic relations in the previous section) is so that the comma information can be used to determine the boundaries of the syntactic relations and contain attachment ambiguity.
When commas are not present, the boundary is determined by Guideline 1 – avoiding explicit attachment decisions.

The gold-set does not have a specialized tag for appositions. They are represented in a similar format to a list of noun phrases: *Catherine allied herself with [NP [NP Henry] , [NP the Duke of Guise ]]*. There is also no specialized category for transitional phrases. They are identified as prepositional phrases (PP) by the gold-set. The gold-set, however, distinguishes between several types of prepositional phrases by adding a suffix to the PP tag, for example:

- **PP-TMP** Temporal: ... *in September*
- **PP-LOC** Location: ... *on the Internet*
- **PP-DTV** Dative Object: *Aristotle gave the book [PP-DTV to Plato]*
- **PP-BNF** Benefactive: *Susan baked a cake for Doug*
- **PP-DIR** Direction: *I flew [PP-DIR from Tokyo] [PP-DIR to New York]*
- **PP-MNR** Manner: *She hit the nail [PP-MNR with the hammer]*
- **PP-PRP** Purpose: *the Dow Jones average went down, [PP-PRP due largely [PP to further selling [PP-LOC in UAL]]]*
- **PP-CLR** Closely Related to the verb: ... *donate your time [PP-CLR to a good cause]*
- **PP** Does not fall into these categories: *the cake was eaten [PP by Mary]*

The larger-first approach does not attempt to make these distinctions since the semantic interpreter (Gomez, 2001), to which the partial parse will serve as input, identifies such semantic roles.

Adverbial phrases are identified as ADVP in the gold-set. Similar to prepositional phrases, some further distinctions are made with suffixes: **ADVP-DIR** (Direction: *the*
average went [ADVP-DIR down]; ADVP-MNR (Manner: *She waited* [ADVP-MNR impatiently]); ADVP-TMP (Temporal: *You left* [ADVP-TMP early]). CO-NP is represented in the gold-set in a similar fashion to appositions and lists of noun phrases.

### 3.3 Coordinate Conjunctions

There are three types of conjunctions: 1) *Correlative* Conjunctions, 2) *Subordinate* Conjunctions, and 3) *Coordinate* Conjunctions. Correlative conjunctions, such as *either..or* and *neither..nor*, are not explicitly disambiguated here due to their relatively infrequent occurrence. Subordinate conjunctions have already been mentioned in Section 3.2.3 and are further discussed in the Section 3.4. Only coordinate conjunctions (hereafter referred to simply as conjunctions) are identified by this intermediate component. Only the *CC* tag is provided by the Penn Treebank Tagset to tag conjunctions, leaving a considerable amount of ambiguity present in the sentence for a parser. A syntactic relation set that fully disambiguates *clausal* conjunctions and partially disambiguates *phrasal* conjunctions in a sentence is defined here.

*Partial disambiguation* of conjunctions is defined here as identifying the post-conjunct of a coordinate conjunction, but not the pre-conjunct. The ending boundary of the post-conjunct is determined by following the Guidelines. *Phrasal* conjunctions are not fully disambiguated because when a noun or prepositional phrase is being coordinated there are usually several possible preceding coordination sites (pre-conjuncts). Choosing the correct one often requires semantic information. When a clause is in coordination, however, there is usually only one possible pre-conjunct, making the full disambiguation of coordinated clauses more accurate. When multiple clausal pre-
conjuncts are present, the rightmost is chosen as the coordination site. An empirical analysis reveals a high accuracy in this approach, however, erroneous groupings can be made (see Section 6.1 for more details).

Partially disambiguated syntactic relations are not needed in the gold-set, since conjunctions are fully disambiguated within the parse tree of the sentence. For example, in the following sentence taken from the Penn Treebank Manual, both conjunctions have been fully disambiguated since both the pre- and post-conjuncts have been identified:

(S (NP-SBJ (NP These girls)
  and
  (NP those boys))
  (VP (VP throw
    (ADVP well))
    and
    (VP catch
      (ADVP-MNR badly))))

Many coordinate conjunctions occur in close proximity to commas and are already partially disambiguated by the comma network. In fact an inspection of the manually comma-tagged corpus (described earlier in this chapter) reveals that over 30% of coordinate conjunctions are partially disambiguated by considering the comma information first.

3.3.1 Clausal Conjunctions

When two clauses that are not delimited with commas are in coordination, they are fully disambiguated. These clauses include: infinitival (CC-INF), relative (CC-REL), subordinate (CC-SUB), gerund/participle/reduced-subordinate (CC-ING), and verb (CC-VS) clauses. CC-VC is used to identify the post-conjunct verb clause in coordination. For example: *This morning I [ CC-VS drove to the library [CC-VC and found an interesting*
with a study guide. Once again the ending boundary is determined by avoiding explicit attachment issues. With a study guide is therefore not included in the verb clause, even though it should be attached to an interesting book. Making the attachment would have resulted in an error if the sentence had been This morning I drove to the library and found an interesting book with my uncle's car. In both cases, only and found an interesting book would be identified as the second verb clause that is being coordinated. The prepositional phrases are not included in the verb clause because syntactically it cannot be determined where they should be attached.

The other coordinated clauses all have a similar syntax. Coordinated relative clauses, for example, would be represented as follows: They bought a truck [CC-REL that is red and [REL that has many scrapes ] ] on its front bumper or They bought a truck [CC-REL that is red [CC-VC and has many scrapes ] ] on its front bumper.

3.3.2 Phrasal Conjuncts

There are two tags associated with noun phrases: (1) CC-NP for conjoining separate noun phrases; and (2) CC-CNP for identifying coordination within a single noun phrase that can be determined with syntax alone. For example, CC-NP would be used to conjoin the following two noun phrases: the police boat [CC-NP and race car].

Unfortunately there is only one pattern of coordination within a noun phrase (CC-CNP) that can be recognized accurately (95% - Resnik (1998)) based on syntax alone: number dissimilarity - for example: business [CC-CNP and marketing majors]. Otherwise, resolving coordination within a noun phrase requires semantic information (Resnik, 1998), for example: (bank and warehouse) guard, freshman ((business and
marketing) major), (food (handling and storage) procedures, ((mail fraud) and bribery) charges, and Clorets (gum and (breath mints)). Resolving these ambiguities is beyond the scope of this approach. The CC-NP category would incorrectly be assigned here, but could be later resolved using semantic information.

The gold-set uses a special category to identify coordination within a noun phrase: UCP (Unlike Coordinated Phrase): [NP [UCP federal and state] laws]. CC-PP coordinates two prepositional phrases, as in: Linda walked by the house [CC-PP and across the road].

It should be noted that two types of conjunctions are absorbed by syntactic relations later in Section 3.5. These are: coordinated adjectives in a predicate or noun phrase – [The long and tiresome journey] finally came to an end. and The journey was [long and tiresome].; and two coordinated adverbs, as in The counselor advised him [privately and confidentially]. Also, conjunctions used without a preceding comma are included in the CO-S syntactic relation. For example, Beth walked down the street [CO-S and Peter followed her in his car].

### 3.4 Clauses

These syntactic categories identify clauses that are not delimited by commas. Only post verbal noun phrases, lists of noun phrases or predicates are included when grouping these syntactic relations. The Guidelines in Figure 1.2 of Chapter 1 are followed closely when determining the boundaries of these syntactic categories.

For example, an infinitival clause is grouped as follows: Beth likes [INF to read a book] about airplanes. The post verbal noun phrase (a book) is included in the infinitival
clause, but *about airplanes* is not included even though it should be attached to *a book*. This does not violate the guidelines, since by Guideline (3) *about airplanes* can still be attached within the infinitival clause. Now consider the sentence: *John needs [INF to give the girl] on the bus a history book*. Even though *a history book* is also a post-verbal complement to *give*, it is not included in the infinitival clause because the explicit attachment decision of *on the bus* would need to be determined. Guideline (1) would be violated had the attachment decision been made. The relative clause groupings are made in a similar fashion by following the guidelines.

The SUB (Subordinate Clause) category is similar to the CO-SUB category in Section 3.2.3 except that these syntactic relations are not enclosed with commas so little containment of ambiguity can occur. SUB is also used to identify clause complements without an introductory subordinate conjunction, for example: *Mary said [SUB the teacher hit her]*. Again, in all of these syntactic relations, the ending boundary is determined by avoid explicit attachment decisions. In the gold-set, S is used to identify such syntactic relations: *Mary told Bill [S the teacher hit her]*.

Gerunds, reduced subordinate clauses and participle clauses (ING complement clauses) are also identified here, for example: *[ING Walking] is a good form of exercise.* and *Mr. Smith accused the students [of ING cheating ] on the test*. These syntactic categories as marked as VP in the gold-set and further annotated to identify their role in the sentence.
3.5 Phrases

These syntactic categories represent the smallest syntactic relations. Prepositional Phrases (PP) have already been described and compared to the gold-set in Section 3.2.4. Noun phrases are identified by the NP category and could contain adjectives coordinated with a comma or conjunction. No attachments are made to noun phrases. The NP category is also used in the gold-set, but it can also include a subject suffix: [NP-SBJ Peter] ate [NP the cake].

Adverbial phrases (ADV) and Adjective phrase (ADJ) identify a grouping of adverbs or adjectives respectfully, for example: You should call the police [ADV immediately] and Fishing is [ADJ fun]. These phrases can include a coordinate conjunction. The gold-set uses ADVP and ADJP to identify these phrases. Suffixes can also be added to these categories. The ADVP suffixes were shown in Section 3.2.4. ADJP can include the PRD (Predicate) suffix, for example: Fishing is [ADJP-PRD fun].

Verb phrases (VP) contain only verbs except when other constituents are placed within the verb phrase, for example: Beth [VP will hopefully attend] the conference in New Mexico and Beth [VP will, of course, attend] the conference in New Mexico. The same VP category is used in the gold-set.

3.6 CASS Category Set

A description of the syntactic categories identified by the CASS system is presented here in order to facilitate a comparison to the larger-first system in Chapter 8. CASS does not provide a formal specification of its syntactic categories. Therefore, this analysis is based on observations gathered while using the CASS system and is not meant to be a formal
specification. The major syntactic relations identified by the CASS system are shown in Table 3.3.

$N_x$, $v_x$, $p_p$, $r_x$, and $a_x$ represent syntactic relations very similar to the following categories defined above: NP, VP, PP, ADV, and ADJ, respectively. The main clause (C) category does not have a defined category here. The main clause is assumed to contain the top level constituents. $Infp$, $subc$, and $rc$ are also very similar to the following categories defined above: INF/CO-INF, SUB/CO-SUB, and REL/CO-REL, respectively. An exception being that reduced relative clauses with a past participle introductory verb are represented with $vnp$ and reduced relative clause with a present participle introductory verb are represented with a $vgp$ category. Here are some example usages: The electricity sale, [vnp scheduled for next year], is expected to raise 13 billion dollars; The stock, [vgp having lost much of its value], closed at $1.70 per share. Vgp is also used to represent reduced subordinate clauses (RSUB/CO-RSUB), for example: [vgp By varying the refining processes], different kinds of asphalt may be obtained.

Name and $inf$ are not identified by the larger-first approach, but could be extended to include these categories. $C-inv$ is used to identify an inverted clause, for example: I want to go home, [c-inv said the girl]. $Pp$-comp is an intermediate category used when a preposition that could also be a subordinate conjunction is present in the sentence. $Pp$-comp is first assigned to indicate that this could be a prepositional phrase or that start of a subordinate clause, for example: John went to the black board [subc [pp-comp after the teacher] threaten to expel him].
Finally, *ng* (noun grouping) is used to represent lists of noun phrases (CO-LST-NP), as well as appositions (CO-APS) and two coordinated noun phrases (CO-NP). As mentioned in Chapter 1, the CASS system cannot be extended to further disambiguate these syntactic relations.

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
<th>Closest Corresponding Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>nx</td>
<td>Noun phrase</td>
<td>NP</td>
</tr>
<tr>
<td>vx</td>
<td>Verb Phrase</td>
<td>VP</td>
</tr>
<tr>
<td>C</td>
<td>Main Clause</td>
<td>null</td>
</tr>
<tr>
<td>pp</td>
<td>Prepositional Phrase</td>
<td>PP CO-PP CO-TRN</td>
</tr>
<tr>
<td>pp-comp</td>
<td>Prepositional Phrase or start to a Subordinate Clause</td>
<td>N/A</td>
</tr>
<tr>
<td>c-inv</td>
<td>Inverted Clause</td>
<td>N/A</td>
</tr>
<tr>
<td>ax</td>
<td>Adjective Phrase</td>
<td>ADJ</td>
</tr>
<tr>
<td>infp</td>
<td>Infinitival Phrase</td>
<td>INF CO-INF</td>
</tr>
<tr>
<td>inf</td>
<td>Infinitive verb</td>
<td>N/A</td>
</tr>
<tr>
<td>ng</td>
<td>Lists of noun phrases/Two coordinated noun phrases/noun with apposition/Noun phrases containing of</td>
<td>CO-LST-NP CO-APS CO-NP</td>
</tr>
<tr>
<td>name</td>
<td>Proper Noun Phrase</td>
<td>N/A</td>
</tr>
<tr>
<td>vp</td>
<td>Verb clause</td>
<td>CO-VC CC-VC</td>
</tr>
<tr>
<td>vgp</td>
<td>Gerund/participle phrase/relative clause with present participle introductory verb</td>
<td>CO-ING CO-REL RSUB CO-RSUB</td>
</tr>
<tr>
<td>rx</td>
<td>Adverbial Phrase</td>
<td>ADV CO-ADV</td>
</tr>
<tr>
<td>subc</td>
<td>Subordinate Clause</td>
<td>SUB CO-SUB</td>
</tr>
<tr>
<td>timex</td>
<td>Time Noun Phrase</td>
<td>TNP</td>
</tr>
<tr>
<td>rc</td>
<td>Relative Clause</td>
<td>CO-REL REL</td>
</tr>
<tr>
<td>date</td>
<td>A Date</td>
<td>TNP</td>
</tr>
<tr>
<td>vnp</td>
<td>Reduced Relative clause</td>
<td>CO-REL REL</td>
</tr>
</tbody>
</table>
CHAPTER 4

THE LARGER-FIRST PARADIGM

When I read a sentence, I read it a chunk at a time.
- Steven Abney (1994)

The larger-first partial parsing paradigm is explained in this chapter. First, an initial design of the larger-first approach is discussed, including why the approach evolved into its current state. Second, a conceptual model representing the design and ordering of the automata is presented. Third, the algorithm, which processes the cascade of automata and constructs the partial tree structure of an input sentence, is described in detail. Some notation that will be used in the automata of the following three chapters is presented next. Finally, this chapter concludes with an overview of research (other than Abney’s work which was described in Chapter 1) that is currently being performed in partial parsing.

4.1 Initial Design

Originally (van Delden and Gomez, 2003), the automata of the larger-first approach were designed to utilize only part-of-speech information. The arcs of the automata were taken only on part-of-speech tags and, in some cases, lexical items. This initial development achieved a performance very similar to its current state presented in this dissertation.
However, there were some drawbacks to this initial system: (1) there was considerable duplication of work; (2) the automata did not have a straightforward design and seemed rather complex; (3) the speed of the system was slowed because the algorithm had to process multiple levels of tags in the sentence; and (4) a pre-processing step interfered with the identification of some overlapping syntactic relations.

The original larger-first partial parsing algorithm is shown in Figure 4.1. Each automaton was processed on every level of tags. Initially, there is only one level of tags—part-of-speech tags assigned to every token by a tagger. Each specialized network was capable of introducing one or more new levels of structural-tags to the sentence. The networks were still considered in descending order of the size of the syntactic relations that they identify: comma, conjunction, clause, then phrase.

```
for each network_i
  for each automaton_ij that assigns structural tag_j
    for each level_k
      try automaton_ij at each position n on level_k
      if automaton_ij accepts at n + m
        insert structural tag_j at level_k to the designated input tokens and continue processing at the position n + m + 1 of level_k
      otherwise
        continue processing at position n + 1 of level_k
```

Figure 4.1 The Original Larger-First Partial Parsing Algorithm.

Each automaton identified only the syntactic relation it was designed for. For example, the relative clause automaton would group the tokens that formed a relative clause, but would not identify the syntactic relations inside of it, for example, noun and verb phrases. This was the job of the smaller noun and verb phrase automata later in the cascade. The smaller automata would have to be processed on two levels in the sentence: the remaining
part-of-speech tags on the sentence level; and the new level introduced by the relative clause. However, implicitly these syntactic relations have already been identified when the relative clause was identified. In the new version of the approach, the relative clause is identified simultaneously with its internal partial tree structure.

The complexity of the algorithm is dependant on the number of levels ($m$) of structural tags assigned by the algorithm and the number of words ($n$) in the sentence. The number of automata is pre-determined and so only adds a constant time complexity. However, since usually $n >> m$, the complexity of the algorithm depends really only on $n$. Therefore, the speed of the algorithm is linear because it depends only on the size of the sentence. However, the overhead of processing each automaton on multiple levels added a high constant time complexity and thus slowed the overall performance of the system. Furthermore, as $m$ approaches $n$ (for smaller sentences) the time complexity becomes $O(n^2)$. The new version of the algorithm does not consider multiple levels of tags and reduces the processing time of a single sentence considerably.

There were two pre-processing automata in the original approach which first grouped non-comma-delimited relative and infinitival clauses, since they often interfered with the acceptance of the comma automata. This pre-processing step produced incorrect groupings when the non-comma-delimited relative and infinitival clauses contained comma-delimited syntactic relations as in: *John bought a television [REL that had ] [LST-NP a remote controller, a DvD hook-up, and HDTV capabilities ]*. The correct grouping should have been: *John bought a television [REL that had [LST-NP a remote controller, a DvD hook-up, and HDTV capabilities ]]*. Furthermore, these pre-processing automata represented a flaw in the larger-first cascade, and should have been in the
4.2 Conceptual Model

An important feature that was added to the initial version of the larger-first approach is the ability of the automata in the cascade to interact with each other. The arcs of the automata are taken on part-of-speech tags, but an arc of an automaton can also be taken by: (1) making a forward call to another automaton later in the cascade, and (2) making a backward reference to the structural tags that have already been assigned by automata earlier in the cascade. These two capabilities allow for a much better representation of the syntactic structures being identified as well as a better time complexity of the algorithm.

A forward call to another automaton is similar to how a Recursive Transition Network, or RTN (Woods, 1970; Winograd, 1983), has the ability to call another RTN. For example, a sentence RTN can call a noun phrase RTN to recognize a noun phrase in the sentence. However, there are several differences between RTNs and a larger-first cascade. The first being robustness. There is no single breaking point in the larger-first approach since it is a cascade of several automata. For example, if the relative clause automaton fails to recognize a relative clause, elements inside and surrounding that syntactic relation will still be identified and the output will still contain a partial tree structure. A second difference is that there is no need for either direct/indirect recursion in the automata of the cascade. Recursion is built into the RTNs so that each possible
parse tree can be recovered. For example, the prepositional phrase RTN can call the noun phrase RTN which in turn can call the prepositional phrase RTN. This (indirect) recursion is avoided since no attachment is performed by the larger-first approach and since backward references will allow some recursion to be indirectly captured (see below). A third difference is that since attachment issues are avoided, only one possible (partial) parse is always created.

In some cases, the need for recursive capabilities may appear to be necessary. However, the larger-first cascade is capable of identifying such structures while avoiding recursion by making a backward reference to a syntactic relation already identified in the cascade. For example consider the sentence: Peter wants to go to the beach, the mall by Mary's house, and the club later tonight. It would appear that the prepositional phrase automaton would call the list of noun phrases automaton which in turn would have to call the prepositional phrase automaton again. However, since large, comma-delimited syntactic relations are identified first, the list of noun phrases will have already been identified before the prepositional phrase automaton is processed on the sentence level. The list of noun phrases automaton will make a forward reference to the prepositional phrase automaton (as well as the noun phrase automaton) so that the following grouping will be made at that point: Peter wants to go to [LST-NP [NP the beach] , [NP the mall] [PP by [NP Mary's house] ] , and [NP the club ] ] later tonight. The prepositional phrase automaton makes a backward reference to the structural tags assigned by the list of noun phrases automaton so that the prepositional phrase can be recognized: Peter wants to go [PP to [LST-NP [NP the beach] , [NP the mall] [PP by [NP Mary's house] ] , and [NP the club ] ] ] later tonight. The prepositional phrase (which contains a prepositional
phrase within a list of noun phrases) has been recognized without the need for a recursive call. A prepositional phrase and a list of noun phrase are examples of *overlapping* syntactic relations. Either can be subsumed by the other. The larger-first approach correctly identifies both possibilities with forward calls and backward references.

The entire larger-first cascade of automata can be represented graphically with their associated forward calls and backward references. Figure 4.2 offers this graphical representation. The syntactic relations in the graph are ordered exactly how they are processed—top to bottom, left to right. This graph depicts a conceptual model of the entire larger-first paradigm. There is no indication here how these syntactic relations are identified. Figure 4.2 is a concise representation of how all the automata interact with each other.

Boxes are used to represent a call to an automaton. A box within another box represents a forward call to an automaton later in the cascade. A syntactic relation surrounded by << >> represents a backward reference to a model that has already appeared in the cascade. For example, the PP model, on the second page of Figure 4.2 in the Phrase Network, makes a forward call to the NP model and a backward reference to the CO-LST-NP model.

Phrasal prepositions (PIN) and phrasal subordinate conjunctions (PSUB) are recognized first by the Pre-processing Network. Backward references are made to these automata by the prepositional phrase model (CO-PP, PP) and subordinate clause model (CO-SUB, SUB).
Figure 4.2 Conceptual Model
Direct (CO-DIR) and indirect speech (CO-IDIR) are identified next. Speech could contain any of the other syntactic relations in the model, however, since it is not the focus of this research (this work focuses on written text) only a few clauses and phrases are identified within the speech.

Lists of infinitival clauses (CO-LST-INF) are identified prior to lists of verb clauses (CO-LST-VC) since CO-LST-INF is a more specific type of verb clause list. Lists of noun phrases (CO-LST-NP) are then identified after lists of verb clauses.

Single coordinated verb clauses (CO-VC) enclosed by commas are recognized next. A list of verb clauses can actually subsume a CO-VC when a comma is preceding the conjunction. This is an example of the larger relation being identified prior to a smaller one. The time noun phrase (CO-TNP) and apposition (CO-APS) models are listed next. A CO-TNP could be mistaken for an apposition which starts a sentence. Since more information (in the form of lexical clues) are used to identify CO-TNP, its model precedes the CO-APS model. Also, both of these models follow the CO-LST-NP model since they are syntactically smaller.

Several clauses enclosed by commas are now identified in the clause section of Comma Network. Note that forward calls made by these models are similar to their non-comma-delimited counterparts on the second page of Figure 4.2. However, the comma-delimited syntactic models provide a much better containment of ambiguity by using comma information to contain explicit attachment decisions (see their automata in Chapters 5 and 7).

Transitional phrases (CO-TRN) precede comma-delimited prepositional phrases (CO-PP). Syntactically, these relations can be the equivalent. However, transitional phrases
are a more specific type of syntactic relation and is therefore identified first using lexical clues.

Coordinated independent clauses (CO-S) are recognized at this point, but could have been recognized at the very beginning since an independent clause is the largest syntactic relation. However, this would require the CO-S model to contain forward calls to every other model in the cascade. Placed in the middle of the cascade, CO-S maintains coverage by making backwards references to all the preceding models while remaining computational more desirable. In the following chapter we will see that the automata for CO-S as well as other coordinated syntactic relations (CO-PP, CC-PP, and CC-SUB) must be carefully designed to allow the best interaction among them. For example, consider the following sentence: *Beth went to the mall during the holiday sale on Saturday, and on Sunday, she went to the beach with her family.* CO-PP will consume *and on Sunday*, but *and on Sunday* is actually the start of an independent clause. The conflict is resolved by only allowing CO-PP to assign structural tags to specific tokens that it consumes. In this case, only *on Sunday* is assigned structural tags so that the independent clause can still be identified.

The coordination within a noun phrase model (CC-CNP) is placed directly before the general coordinated noun phrases model (CC-NP), since it identifies a more specific pattern (see Section 6.2). The remaining models are all ordered in a general decrease of syntactic relation size. The order is not concrete, since, for example, the identification of a coordinated relative clause (CC-REL) has nothing to do with the identification of a coordinated infinitival clause (CC-INF). These models could easily be swapped without negative affect.
4.3 The Algorithm

The new larger-first algorithm is similar its precursor in Figure 4.1, except that automata are only processed on one level of tags and multiple layers of tags can be assigned by a single automaton. Figure 4.3 presents the larger-first algorithm.

Every automaton in the cascade is attempted at each position in the sentence. A single automaton call make several forward calls to other automata which in turn could make several forward calls to yet other automata. As input tokens are consumed, the implementation must keep track of which tags are being assigned to which input tokens. When an arc makes a forward call to an automaton and the automaton accepts, processing is returned to the original automaton - in the target state of the arc at the next position in the sentence.

```
for each automaton_j
  for each position n of the sentence
    try automaton_j at position n
    if automaton_j makes a forward call to automaton_k
      then tag_j = tag_j/tag_k is to be assigned
d to the tokens automaton_k consumes.
    if automaton_j accepts at n + m
    then insert tag_j in between the designated input tokens
      and their current assigned tag(s).
      Continue processing at position n + m + 1.
```

**Figure 4.3** The Larger-First Partial Parsing Algorithm

The > symbol is used as a prefix to indicate that the current token is grouped with the token that follows it. No > prefix marks the end of the syntactic relation. For example: *Susan/NP had/>VP walked/VP the/>NP dog/NP*. This notation has been used in several partial parsing systems (Ramshaw and Marcus, 1995; Voutilainen and Jarvinen, 1995) to
introduce a new, single layer of structural tags. However, this idea is extended to the next logical step - representing a partial tree structure of multiple levels within the structural-tags. To illustrate the entire process consider the following example:

John bought a television that has a remote controller, a DvD hook-up which is incompatible with Susan's DvD player, and HDTV capabilities.

The sentence is first assigned part-of-speech tags by a tagger:

John/NNP bought/VBD a/DT television/NN that/WDT has/VBZ a/DT remote/NN controller/NN ./, a/DT DvD/NNP hook-up/NN which/WDT is/VBP incompatible/JJ with/IN Susan/NNP 's/POS DvD/NNP player/NN ./, and/CC HDTV/NNP capabilities/NNS ./.

Each automaton corresponding to the syntactic relations in the conceptual model in Figure 4.2 is tried in the sentence. Since the automata are not presented until the next chapter, assume for now that the following calls and steps taken by the automata are valid. The first automaton of the cascade to accept is the list of noun phrases automaton (referred to here as LST-NP). LST-NP is first tried at positions 1 through 6 (John bought a television that has) which results in a halting state. So processing continues and LST-NP is tried at position 7. Assume LST-NP first makes a forward call to the noun phrases automaton (NP) which consumes a remote controller. Note that two levels of tags, LST-NP/NP, are to be assigned to a remote controller if the LST-NP automaton accepts. LST-NP then consumes the comma (which would only be assigned LST-NP) and calls NP once again which consumes a DvD hook-up. LST-NP now makes a call to the relative clause automaton (REL) which will make forward calls to the adjective (ADJ) and verb phrase (VP) automata. REL accepts after consuming which is incompatible. If LST-NP
accepts two tags would be assigned to which (LST-NP/REL), three tags to is (LST-NP/REL/VP), and three to incompatible (LST-NP/REL/ADJ). LST-NP now calls the prepositional phrase automaton (PP) which calls NP and with Susan’s DvD player is consumed. Finally, LST-NP consumes the comma, the conjunction, and then calls NP which consumes the final noun phase HDTV capabilities. LST-NP now accepts and assigns permanent tags in between the appropriate tokens and their part-of-speech tags in the sentence. The sentence after LST-NP accepts becomes (the new layers of structural tags introduced by LST-NP are highlighted):

<table>
<thead>
<tr>
<th>WORD</th>
<th>STRUCTURAL TAGS</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>NNP</td>
</tr>
<tr>
<td>bought</td>
<td>VBD</td>
</tr>
<tr>
<td>a</td>
<td>DT</td>
</tr>
<tr>
<td>television</td>
<td>NN</td>
</tr>
<tr>
<td>that</td>
<td>WDT</td>
</tr>
<tr>
<td>has</td>
<td>VBZ</td>
</tr>
<tr>
<td>a remote</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>controller</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>DvD hook-up</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>which</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>is incompatible</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>with Susan</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>’s DvD player</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>and HDTV</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>capabilities</td>
<td>LST-NP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>STRING</th>
<th>POS</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>NNP</td>
</tr>
<tr>
<td>bought</td>
<td>VBD</td>
</tr>
<tr>
<td>a</td>
<td>DT</td>
</tr>
<tr>
<td>television</td>
<td>NN</td>
</tr>
<tr>
<td>that</td>
<td>WDT</td>
</tr>
<tr>
<td>has</td>
<td>VBZ</td>
</tr>
<tr>
<td>a</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>remote</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>controller</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>DvD</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>hook-up</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>which</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>is</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>incompatible</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>with Susan</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>’s DvD player</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>and HDTV</td>
<td>&gt;LST-NP</td>
</tr>
<tr>
<td>capabilities</td>
<td>LST-NP</td>
</tr>
</tbody>
</table>
The remaining automata in the cascade are processed and the next to accept is REL. REL comes into a halting state when it is tried at the first four positions in the sentence (*John bought a television*). When REL reaches the fifth position, it consumes *that has* and then makes a backward reference to LST-NP by consuming the LST-NP tags. The new layer of REL tags are inserted in between the word and the structural tags that are already assigned. The output becomes:

<table>
<thead>
<tr>
<th>WORD</th>
<th>STRUCTURAL TAGS</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>NNP</td>
</tr>
<tr>
<td>bought</td>
<td>VBD</td>
</tr>
<tr>
<td>a</td>
<td>DT</td>
</tr>
<tr>
<td>television</td>
<td>NN</td>
</tr>
<tr>
<td>that</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>has</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>a</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>remote</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>controller</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>,</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>a</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>DvD</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>hook-up</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>which</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>is</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>incompatible</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>with</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>Susan</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>'s</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>DvD</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>player</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>,</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>and</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>HDTV</td>
<td>&gt;REL</td>
</tr>
<tr>
<td>capabilities</td>
<td>REL</td>
</tr>
</tbody>
</table>

Finally, the remaining noun and verb phrases are identified by NP and VP in the final stages of the cascade. The final output is shown below which can easily be converted to its bracketed form:
John bought a television that has a remote controller, a DVD hook-up which is incompatible with Susan's DVD player and HDTV capabilities.

<table>
<thead>
<tr>
<th>WORD</th>
<th>STRUCTURAL TAGS</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>NP</td>
</tr>
<tr>
<td>bought</td>
<td>VBD</td>
</tr>
<tr>
<td>a</td>
<td>DT</td>
</tr>
<tr>
<td>television</td>
<td>NN</td>
</tr>
<tr>
<td>that</td>
<td>WDT</td>
</tr>
<tr>
<td>has</td>
<td>VP</td>
</tr>
<tr>
<td>a</td>
<td>NP</td>
</tr>
<tr>
<td>remote controller</td>
<td>NP</td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>NP</td>
</tr>
<tr>
<td>DVD</td>
<td>NP</td>
</tr>
<tr>
<td>hook-up</td>
<td>NP</td>
</tr>
<tr>
<td>which</td>
<td>NP</td>
</tr>
<tr>
<td>is</td>
<td>NP</td>
</tr>
<tr>
<td>incompatible</td>
<td>ADJ</td>
</tr>
<tr>
<td>with</td>
<td>IN</td>
</tr>
<tr>
<td>Susan</td>
<td>NP</td>
</tr>
<tr>
<td>'s</td>
<td>POS</td>
</tr>
<tr>
<td>DVD</td>
<td>NP</td>
</tr>
<tr>
<td>player</td>
<td>NP</td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>and</td>
<td>CC</td>
</tr>
<tr>
<td>HDTV capabilities</td>
<td>NNP</td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
</tbody>
</table>
4.4 Some Notation

The automata which recognize the syntactic relations in the conceptual model of Figure 4.2 are presented in the following three chapters. Many abbreviations are used in labeling the arcs of the automata. The abbreviation list on page xiv should be used as a reference when reviewing these chapters. Some of this notation is explained here in more detail.

Before processing starts, the following token is placed at the start of the sentence: \textit{STAART/STAART}. This extra token (also used by Brill, 1994) can be used by the automata when the start of the sentence is an important clue when identifying a syntactic relation. Forward calls in the conceptual model are specified by placing square brackets around the tag of the automaton that is being called. A backward reference is made by specifying the structural tag that has been assigned by a prior automaton, for example: \textit{if the tag is $>$CO-LST-NP or CO-LST-NP then traverse arc}. Both tags can be referenced using greater than and less than symbols: $<$CO-LST-NP$>$.

Instead of a single automaton for each syntactic relation, several automata may be defined to reduce the overall complexity of the system. For example, there are three automata that recognize different types of non-comma-delimited relative clauses (REL). Even though they all recognize relative clauses, each automaton must have a unique name so that its particular automaton can be referenced by a forward call or backward reference if need be. Therefore, the automata would assign different tags such as REL1, REL2, and REL3 to the designated input tokens that they consume.

In some cases, tokens consumed by certain arcs of the automata are not to be issued structural tags. The labels of these arcs will be italicized. The purpose is to use the surrounding context to determine a syntactic relation without having to assign tags to the
surrounding context. So, if an automaton has an italicized arc, the tokens on it will be consumed, but no structural tags will be assigned to them.

Multiple conditions on an arc are implicitly separated by logical OR. For example, if an arc is labeled *Comma START*, then the arc can be taken on a comma or the start of sentence token. The *PREV*, *NEXT*, *LASTTAG*, and *NEXTTAG* conditions are secondary restrictions which are implicitly separated by logical AND from the other conditions on the arc. For example, if an arc is labeled *Comma PREV: VP* then the arc can be taken if a comma is present at the current position and a verb phrase is present in the previous part of the sentence. Logical *OR* and *AND* can also be explicitly used to define arc conditions.

Lexical items assigned to an arc are not case sensitive. For example, if an arc is labeled *W: today*, then it will be taken if the current word in the sentence is *today* or *Today*.

### 4.5 Related Research

There have been many proposed approaches to partial parsing: Finite State (Ait-Mokhtar, and Chanod, 1997; Abney, 1996a; Vilain and Day, 1996; Kupiec, 1993), Memory-Based (Daelemans et al. 1999; Tjong Kim Sang and Veenstra, 1999; Veenstra, 1998), Transformation-Based (Ramshaw and Marcus, 1995; Ramshaw and Marcus, 1994), Stochastic (Church, 1988); Linguistic (Voutilainen and Jarvinen, 1995; Voutilainen, 1993); and, most recently, combining different approaches (Dienes and Dubey, 2003; Frank et al., 2003; Park and Zhang, 2003; Schiehlen, 2003).
Voutilainen and Jarvinen (1995), and Voutilainen (1993) describe a detector of English noun phrases. New types of tags are added to the words in the sentence. For example, the chunk tag @>N is used for determiners and pre-modifiers, indicating they should group with the following noun head. A lexicon, which lists all possible chunk tags, along with hand-built constraint grammar patterns are used to produce a chunk of the noun phrases in the sentence.

Transformation-based learning has also been applied to text chunking (Ramshaw and Marcus, 1995; Ramshaw and Marcus, 1994). This approach is similar to NPTool (Voutilainen, 1993) in the sense that new tags are added to the words in the sentence to avoid bracketing issues. BaseNP chunks identified here include the initial portions of non-recursive noun phrases up to the head, including determiners but not including post-modifying prepositional phrases or clauses. The transformation-based learning algorithm which was described in Chapter 2.4.1 is used with a new set of rule templates.

More recently, the focus in the research community has shifted to learning a partial parser from a corpus. A study by Li and Roth (2001) shows that learning a shallow parser has several advantages over learning a full parser, for example: each layer of a shallow parser can be learned separately. They extracted the base phrases from a learned full parser and compare them to that of a learned shallow parser. However, I do not feel this is good comparison since the full parser is providing a much more detailed output which is being disregarded so that only the base phrases can be compared to the shallow parser’s output.

Munoz et al. (1999) presents a SNoW based learning approach to shallow parsing. The SNoW (Sparse Network of Winnows) learning architecture is a sparse network of
linear functions over a pre-defined or incrementally learned feature space. Using Inside/Outside predictors are compared against using Open/Close predictors for determining noun phrases and subject-verb combinations. Inside/Outside predictors are similar to the "->" notation used here: O – the current word is outside the pattern; I – the current word is inside the pattern; and B – the current word marks the beginning of a pattern which directly follows another pattern. For example, here is how Inside/Outside is used to identify noun phrases:

\[
\begin{array}{cccccc}
I & \text{went} & \text{to California} & \text{last} & \text{May} \\
I & O & O & I & B & I
\end{array}
\]

This notation is adequate when a single pattern is being identified in a sentence, but would not be appropriate for the larger-first system since many patterns are identified. Open/Closed predictors refer to placing brackets \([. . .]\) around the pattern. They found that the both methods perform about the same for identifying noun phrases, but Open/Closed out performs Inside/Outside for subject-verb patterns.

Learning approaches to memory-based shallow parsing have also recently been developed (van den Bosch and Buchholz, 2002; Tjong Kim Sang, 2002; Argamon-Engleson et al., 1999; Daelemans et al. 1999).

Argamon-Engelson et al. (1999) use a novel learning method for recognizing local sequential patterns. Positive and negative evidence from a training corpus is used to recognize a sequence. For example, is the following sequence of part-of-speech tags a noun phrase: \(DT \ ADJ \ ADJ \ NN \ NNP\)? This long pattern may not be in the corpus, however, smaller noun phrases that cover sub-sections of this pattern may be present, like the prefix \(DT \ ADJ \ ADJ \ NN\) NNP and suffix \(DT \ ADJ \ ADJ \ NN \ NNP\). When combined, these sub-sections offer positive evidence that the sequence is a noun phrase. Negative
evidence is generated from subparts in the raw data that do not have the right tag sequence.

Van den Bosch and Buchholz (2002) explore memory-based shallow parsing on the basis of words alone. Part-of-speech tags are used to overcome data sparseness, since a sequence of words is represented as a more general sequence of tags. However, with the abundance of training material currently available, van den Bosch and Buchholz suggests that this material be used directly, avoiding an explicit part-of-tagging step. Their results show that attenuated words (descriptive tags that are given to low-frequency or unknown words to prevent data sparseness) along with gold-standard part-of-speech tags achieves better results than words, or part-of-speech tags alone.
CHAPTER 5

COMMA NETWORK

Of all the punctuation marks the comma is the most flexible in the range of its use, and hence the most difficult to categorize.

- Greenbaum et al. (1985)

Because the comma (,) serves so many different purposes it is the most widely used of all punctuation marks. Its varied and distinct uses results in it being by far the most troublesome of the marks;

- Shaw (1969)

The omnipresence of commas in real world texts makes them impossible to avoid when processing natural languages. This chapter is, therefore, comprised of two techniques that disambiguate commas. First, the entire network of comma automata in the larger-first cascade is presented and described in detail. The automata of the network are partitioned into five stages and presented in the exact order that they are applied. The syntactic relations identified by these automata have already been defined in Chapter 3. Refer to the Abbreviation List on page xiv and Section 4.4 for descriptions of the labels that are placed on the arcs of the automata. Also, refer to Appendix C for the complete automata cascade.

Second, the isolated task of assigning a structural tag to commas alone (comma-tagging) is discussed and a method is presented that accomplishes this task. The idea of comma-tagging is also extended to tagging commas in the Dutch language. The comma-tagging systems for both English and Dutch are evaluated and results are given.
5.1 Pre-Processing

The first step in the larger-first cascade is to identify phrasal prepositions and phrasal subordinate conjunctions so that they can be treated as a single token, simplifying other automata that follow. Phrasal subordinate conjunctions (PSUBs) are identified first by the automaton in Figure 5.1.

![Figure 5.1 Phrasal Subordinate Conjunctions](image)

The automaton is completely lexicalized and will only accept on very specific phrases that are usually PSUBs. The path $ABD$ recognizes PSUBs of length two such as: *as though, even if, rather than, so that, now that, etc.* The path $ABCD$ recognizes PSUBs of length three, such as: *as long as, as soon as, and in order that.* Note that lexical items from the different phrases are clustered together around the same arcs in the automaton, which would allow the automaton to recognize nonsensical phrases such as: *rather of, in long that, etc.* This clustering merely simplifies the design of the automaton, and will not cause a problem in regular English text. It is assumed that this system will be applied to logically written text.

Like PSUB, the first phrasal preposition automaton (PIN1) in Figure 5.2 is completely lexicalized and lexical items from different phrases are clustered around the
arcs of the automaton. It recognizes PINs of three different lengths: length of two: *such as, according to, aside from, because of, instead of, etc*; length of three: *in favor of, in addition to, with respect to, by way of, etc*; and finally length of four: *with the exception of*. The second phrasal preposition (PIN2) automaton is more relaxed than PIN1 and identifies any other sequence of two prepositions as PIN2. This is a default assignment after PSUB and PIN1 have first been attempted. Also, if a coordinate conjunction is surrounded by two prepositions, the three words are grouped as a PIN2.

![Diagram of PIN1 and PIN2 automata]

**Figure 5.2 Phrasal Prepositions**

### 5.2 Speech Automata

Although not the focus of this research, four automata are defined to recognize *direct* and *indirect* speech at this point in the cascade. Figure 5.3 shows the two automata that recognize direct speech.
It is assumed that quotation marks (\textit{Quote}) are being used to enclose direct speech, however, this could easily be changed to another token. The first automaton (CO-DIR1) identifies the first block of contiguous direct speech and the second automaton (CO-DIR2) the second block if it is present. For example: \[ \text{[CO-DIR1} "After patrolling the park for several hours, " ] officer Smith stated, \[ \text{[CO-DIR2} "I was convinced that the perpetrator had fled the area." ] \] Arc CC in CO-DIR1 and BB in CO-DIR2 consume the noun phrase that represents the speaker. These are self arcs so that inverted constructions are also recognized, i.e. switching around officer Smith with stated in the above example. Note that some arc labels are italicized, indicating that the tokens they consume are not to be assigned structural tags. The automata have been lexicalized with \textit{communicate} verbs on arc CD in CO-DIR1 and arc AB in CO-DIR2. These arcs are taken if a communicate verb or any morphological derivation there of is present. The list of communicate verbs is shown in the side box and can easily be extended or restricted.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{figure53.png}
\caption{Direct Speech}
\end{figure}
Instead of maintaining a list of communicate verbs, the hypernymy of the lexical items in the sentence could be retrieved from WordNet (Miller, 1993) and searched for an instance of *communicate*. For example, consider the verb *say*. WordNet’s hypernymy (WordNet version 1.7.1) of Sense 10 of *say* is as follows:

```
Sense 10
say
  => covey, impact
    => communicate, intercommunicate
      => interact
        => act, move
```

*Communicate* is found at the second super-ordinate and would indicate that *say* is a communicate verb. However, when this approach was implemented, many false classifications were made due to the large number of verbs that have *communicate* in their hypernymy but are not used in direct and indirect speech. For example: *beep, snare, apply, bait, overburden*, and *bear down* are a few of the subordinates of *communicate* that would be inappropriate to express direct and indirect speech. A simple verb list is therefore maintained instead of incorporating WordNet into the arcs of the automata.

The two automata that recognize indirect speech are very similar to those in Figure 5.3 and are presented in Figure 5.4. The communicate verbs from Figure 5.3 are also referenced here.
The first automaton (CO-IDIR1) recognizes the first contiguous block of indirect speech, and the second automaton (CO-IDIR2) identifies the second block of indirect speech if it is present. CO-IDIR2 is complicated slightly to ensure that a verb phrase or subordinate clause is present in the second part of the indirect speech. This is to prevent relative clauses or appositions from being identified as CO-IDIR2. For example: [CO-IDIR1 I was convinced that the perpetrator had fled the area,] said Officer Smith, (who was) a ten year veteran of the L.A. police force. Also, CO-IDIR2 should only be assigned if there has already been indirect speech identified in the sentence. For example, [CO-IDIR1 After patrolling the park for several hours,] officer Smith stated, [CO-IDIR2 I was convinced that the perpetrator had fled the area]. If the indirect speech only follows the speaker, it is usually not enclosed in commas and would be recognized as a subordinate (complement) clause later in the cascade. For example: Officer Smith said [SUB that he was convinced] [SUB that the perpetrator had fled the area]. The speech automata here are only concerned with speech that is delimited by commas.
The speaker of the sentence can sometimes be separated from the communicate verb by a comma-delimited syntactic relation. For example, in the following sentence an apposition is inserted: *I was convinced that the perpetrator had fled the area, Officer Smith, a ten year veteran of the L.A. police force, said.* The above automata can easily be extended to handle such cases by inserting a self arc, labeled with forward calls to appositions and relative clauses, between the speaker and the communicate verb. Although important to note, this extension is not preformed at this point since its occurrence was very rare during testing.

### 5.3 Series Automata

The automata in this section identify series of three or more syntactic relations. Since any syntactic relation can be placed in series, this section is limited to only the most frequently occurring series that were found during testing. For example, lists of subordinate, relative and independent clauses, prepositional phrases, and adjectives were all encountered during testing. But because of their relatively low frequency (occurring less than 1% of all series), no automata have been defined here for them. However, the cascade could easily be extended to include these syntactic relations.

The first series to be identified is a list of infinitival clauses. Figure 5.5 presents one of the two automata (CO-LST-INF2) that accomplishes this task. The other (CO-LST-INF1 - not shown here) is very similar to CO-LST-INF2 except that a comma is present before the conjunction. Also, coordinated noun and prepositional phrases (CC-NP and CC-PP) are consumed by CO-LST-INF1 by arcs equivalent to arcs CC and DD in Figure.
5.5. Likewise, for the remaining Figures in this section, only one of the two automata will be shown.

![Figure 5.5 List of Infinitival Clauses](image)

The word to is required before the first infinitival clause of the list, but since it is usually omitted from the others that follow, it is placed on self arcs $DD$ and $FF$. Post-verbal syntactic relations are also placed on self arcs, since they are not required: *I like [CO-LST-INF2 to swim, run and hike]*. The final arc $GH$ consumes a post verbal noun phrase if one is present. If a noun phrase is not present, the arc is still taken, but the consumed token is not assigned a structural tag, for example: *I like [CO-LST-INF2 to swim in the ocean, run across the city and hike] up the mountain.*

The next series to be identified is a list of verb clauses. Figure 5.6 shows the automaton that recognizes lists of verb clauses with no comma preceding the conjunction. This is the largest of all the automata due to the different verb tenses that can appear in the list. It could be split into four simpler automata, one for each type of verb tense.
Figure 5.6 List of Verb Clauses

Arc AB acts a restriction so that relative clauses are not confused as the start of a list of verb phrases. For example, a incorrect grouping could be made in the following sentence if this restriction was not made: *The telescope, that Peter [ CO-LST-VC2 broke during the move, sat on the table for months and was finally thrown away ] not long ago.*

The first part of the label on arc AB states that the arc can be taken if a relative determiner is present and there is a verb in the preceding part of the sentence. This makes it possible to still identify a list of verb clauses when the verb of a relative clause does introduce the list, for example: *Peter broke the telescope, which (CO-LST-VC2 lay on the table for months, collected a lot of dust and was finally thrown out) not long ago.*

When each verb clause in the list is a present participle (VBG), the series could be a lists of gerunds, complement clauses, or reduced subordinate clauses. For example, *(CO-LST-VC2 Walking, hiking and swimming ) are all great forms of exercise.* or *Our
employee took pride in (CO-LST-VC2 painting the rooms, restoring the roof, and tiling the kitchen). No distinctions are made between these syntactic relations.

Finally, the last type of series identified here is a list of noun phrases. The automaton in Figure 5.7 identifies a list of noun phrases with no comma preceding the conjunction.

![Figure 5.7 List of Noun Phrases](image)

Relative and infinitival clauses, as well as prepositional phrases, can be contained with in the list. However, a problem occurs when the initial noun phrase has a relative clause or prepositional phrase attached to it. For example, the start of the following list of noun phrases would be incorrectly identified: The Democrats who voted for the bill were John Breaux of [CO-LST-NP2 Louisiana, Dianne Feinstein of California and Ron Wyden ] of Oregon. A solution to this problem is to create another CO-LST-NP automaton in which every noun phrase in the list must have a prepositional phrase or relative clause attached to it. This new automaton would be placed directly before the one in Figure 5.7 and would correctly handle the above example. An empirical analysis revealed that, when the first noun phrase in a list has at least one prepositional phrase attached to it, the rest usually do also. Adding the extra automaton resolved 92% of all ambiguous cases encountered during testing.

However, some problematic situations cannot correctly be resolved here since they require semantic information. For example, an incorrect grouping will be made in the
following sentence: *Beth brought the strawberries that were freshly picked by [CO-LST-NP2 the neighbors, the bananas, and the apples]*. Semantics is needed to realize that the *strawberries* is actually the first item in the list. Such lists cannot correctly be identified here, but fortunately they occurred very infrequently during testing.

Errors may also occur when distinguishing between a list of noun phrases and an apposition. Whenever an apposition contains a coordinate conjunction, there is the possibility of confusing it with a list of noun phrases. For example, *The assignment was given to John Smith, president of the board and general manager of all restaurants in that area.* The ambiguity can be resolved by introducing a new apposition automaton to look for the following pattern:

\[
\text{proper-noun} \text{, noun-phrase(not proper)}
\]

or

\[
\text{noun-phrase(not proper)} \text{, proper-noun}
\]

where the WordNet (Miller, 1993) hypernyms of the head noun in noun-phrase must contain the super-concept "person", "region" or "organization". The motivation behind this automaton is the fact that a proper noun is usually used to name a *person, place, or organization*. Because at least one of the noun phrases must be proper, this solution corrects most errors without producing many of its own, with a correction to error ratio of about 100:1 during testing. This automaton would be added directly before the list of noun phrases automaton in the cascade, and would have resulted in a correct grouping of the above example. Although easy to incorporate, this automaton has not been added to
the larger-first cascade since it requires semantic labels to be placed on the arcs of the automaton. A strictly syntax-based cascade is maintained here.

Finally, another ambiguity that is not resolved is when a list of noun phrases is confused with a single noun phrase containing a list of pre-noun modifiers. For example, a list of post-verbal noun phrases is identified in the following sentence when actually there is only one post-verbal noun phrase: *The terrorists targeted [CO-LST-NP2 the FBI, CIA and Capitol buildings]*. This example could be corrected by noticing the syntactic number dissimilarity, and would result simply in designing another automaton (actually this would be an extension of the CC-CNP automaton in Chapter 6) that would recognize such patterns as single noun phrases. Again, this will not resolve the noun phrases that do not contain syntactic dissimilarity - semantics is required.

### 5.4 Clausal Automata

The clause automata make up the next section of the cascade. Only clauses that are delimited with commas are recognized here. Unlike the previous automata, comma information is used explicitly here to help contain attachment ambiguity and determine the boundaries of the syntactic relations.

The verb clause automaton is presented first in Figure 5.8. It recognizes a coordinated verb clause that is enclosed by commas. The syntactic relation must be introduced by a comma followed by a conjunction and then a verb phrase. Since a comma (or semi-colon or EOS - End-Of-Sentence) is being used to determine the final boundary, many other syntactic relations can be contained within the verb clause (arc DD).
As with most of the automata in this chapter, the commas themselves are not assigned structural tags because they may interfere with the acceptance of another automaton. For example, in the following sentence, if the final comma is tagged then the apposition automaton may not be able to recognize the apposition: *Peter went to the mall that recently opened four new stores, [CO-VC and bought a diamond bracelet for Mary], his wife of seven years*. Even though the apposition is not contained within the verb clause, the Guidelines (set forth in the Chapter 1) are not violated since the apposition can be attached within a preceding clause.

![Figure 5.8 Coordinated Verb Clause Enclosed by Commas](image)

Figure 5.8 Coordinated Verb Clause Enclosed by Commas

Figure 5.9 presents comma-delimited infinitival clauses which are recognized next. These are purpose infinitival clauses, since complement infinitival clauses are not usually enclosed by commas. Again, many syntactic relations are contained by using comma information and the commas are not assigned structural tags.
Comma-delimited relative clauses are now identified. Figure 5.10 presents the automata that recognize two different types of relative clauses. The first automaton (CO-REL1) identifies reduced relative clauses in which the relative determiner and auxiliary verb are omitted. For example, *The offensive players on the team, [CO-REL1 called the attackers]*, *must wait for the opposition to let their guard down.* Every automaton in the cascade depends on the part-of-speech tags that are assigned by the part of speech tagger. The reduced relative clause automaton, however, is particularly sensitive to tagging errors since past tense verbs are often confused with past participles and vice versa. For example, the following sentence has one reduced relative clause introduced by the past participle verb *urged*: *John, [CO-REL1 urged on by his classmates]*, *walked up to the board*. *Urged* is in its past participle form which also happens to be the same as its past tense form, while *walked* is in its past tense form which also happens to be the same as its past participle form. An incorrect grouping will be made if the tagger confuses either of these verb tenses.

As with the previous automata in this section, many other syntactic relations are contained (arcs *CC* and *FF*). Also, since the CO-REL1 automaton must recognize and consume specific verb types (*VBN VBG* on arc *BC*), the verb phrase is not explicitly grouped since the verb automaton is not called. Therefore, arc *CC* is labeled with the particle tag (RP) that would normally be grouped with the verb phrase.
Reduced relatives that start a sentence are also recognized by following path $AB(BB)^*BC(CC)^*CD$ through the first automaton. For example, [ CO-REL1 Unconcerned by the growing treat of terrorists ] , the president allowed the games to continue.

The second automaton in Figure 5.10 identifies relative clauses which are introduced with a comma followed by: a Wh-Determiner or Wh-Pronoun; a preposition followed by a Wh-Determiner or Wh-Pronoun; or a noun phrase which is then followed by a preposition and then a Wh-Determiner or Wh-Pronoun. The following three sentences are examples of these cases, respectively: Mary wanted to study with John, who was currently out of town. Denmark has many busy seaports, of which Copenhagen is the most important. Frank read five books this year, the first of which he like the most. This automaton is relaxed (all forward calls to syntactic relations - even the verb clause - are
made by the self arc CC) since a comma followed a relative determiner is a strong indication that a relative clause is being introduced.

Finally, to conclude this segment on relative clauses and illustrate the complexity that these automata can achieve, the following sentences that were encountered during testing are presented:

\[
\begin{array}{l}
( \text{CO-REL1 Cut/VBN off/RP ( PP from/IN ( NP his/PRP$ base/NN ) ) ) ,/}, ( NP Darius/NNP ) ( VP fled/VBD ) ( ADV northward/RB ) ,/}, ( \text{CO-REL1 abandoning/VBG ( LST-NP1 ( NP his/PRP$ mother/NN ) ,/}, ( NP wife/NN ) ,/}, \text{and/CC ( NP children/NNS ) ) ( PP to/TO ( NP Alexander/NNP ) ) ) ,/}, ( \text{CO-REL2 who/VP ( VP treated/VBD ) ( NP them/PRP ) ( PP with/IN ( NP the/DT respect/NN ) ) ( NP due/JJ ) ( PP to/TO ( NP royalty/NN ) ) ) ./} \\
( \text{CO-REL1 Disillusioned/VBN ( PP by/IN ( NP the/DT impossibility/NN ) ) ( ING of/IN reconciling/VBG ( NP certain/JJ contradictory/JJ Manicheast/NNS doctrines/NNS ) ) ( CC-NP and/CC ( NP internal/JJ dissent/NN ) ) ,/}, ( NP Augustine/NNP ) ( CC-VS ( VP abandoned/VBD ) ( NP this/DT philosophy/NN ) ( CC-VC and/CC ( VP turned/VBD ) ) ) ( PP to/TO ( NP skepticism/NN ) ) ./} \\
( \text{CO-SUB When/WRB ( NP husbands/NNS ) ( CC-VS ( VP died/VBD ) ( CC-VC or/CC ( VP abandoned/VBD ) ( NP their/PRP$ families/NNS ) ) ) ,/}, ( NP women/NNS ) ( VP had/VBD ) ( NP no/DT choice/NN ) but/CC ( INF to/TO ( VP work/VB ) ) ) ,/}, ( \text{CO-REL1 opening/VBG ( NP a/DT shop/NN ) ( SUB if/IN ( NP they/PRP ) ( VP had/VBD ) ( NP the/DT capital/NN ) ( CC-VC or/CC ( VP working/VBG ) ) ( PP in/IN ( NP a/DT sweatshop/NN ) ) ( SUB if/IN ( NP they/PRP ) ( VP did/VBD not/RB ) ) ) ./} \\
\end{array}
\]

Comma-delimited subordinate clauses are identified next in the cascade. Figure 5.11 shows the corresponding automaton. A comma or the start of a sentence can introduce the subordinate clause which must be concluded by a period or another comma. The phrasal subordinate conjunction tags (PSUB) that were previously assigned by the PSUB automaton are referenced in arcs BB and BC. If a PSUB is present after the start of the sentence or comma, the automaton reaches state C. State C can also be reached if a single subordinate conjunction is present. Instead of referring to its ambiguous part-of-speech
tag (IN), a list of possible subordinate conjunctions (SC) is maintained and shown in Figure 5.11 also. This prevents prepositions that cannot act as subordinate conjunctions from mistakenly being treated as subordinate conjunctions.

![Diagram of comma-delimited subordinate clause]

**Figure 5.11** Comma-delimited Subordinate Clause

A subject is required and could be a single noun phrase or list of noun phrases - arc CD. Post subject modifiers could be present such as prepositional phrases or relative clauses. These syntactic relations would be consumed by arc DD. The verb phrase of the clause is then consumed by arc DE, and finally arc EE makes many calls to possible post verbal syntactic relations until a comma or End-Of-Sentence is reached.

A coordinate conjunction can be consumed by arc BB because it is sometimes placed at the start of a sentence (or independent clause) and may inhibit this automaton from accepting. For example, each of the following sentences contain subordinate clauses that are recognized here: *And [CO-SUB just after Peter finished re-wiring the entire system], the technician showed up.* and *The new entertainment system had not worked properly for almost two days, and [ CO-SUB just after Peter finished re-wiring the entire system], the*
technician showed up. The conjunction is not included in the subordinate clause so that the coordinated independent clause (in the second sentence) can still be recognized - see Section 5.5. Had the CC tag not been included on arc BB, the CO-SUBs would not have been recognized in the above examples. However, they would have been recognized later by the non-comma-delimited subordinate clause automaton (SUB), but the comma information would not be used at that point and so containment of ambiguity would be limited.

Finally, to conclude this segment on subordinate clauses and illustrate the complexity of the sentences that this automaton can achieve, the following sentence that was encountered during testing is presented:

And ( CO-SUB because/IN ( NP light/NN ) ( REL1 reflected/VBN ) ( PP from/IN ( NP a/DT large/JJ flat/JJ surface/NN ) ) ( PP ( PIN1 such/JJ as/IN ) ( NP water/NN ) ) ( CC-NP or/CC ( NP a/DT wet/JJ road/NN ) ) ( VP is/VBZ partially/RB polarized/VBN ) ) ,/ ( ADV properly/RB ) ( NP oriented/JJ Polaroid/NFP ) ( VP can/MD absorb/VB ) ( NP more/JJR ) ( PP than/IN ( NP half/NN ) ) ( PP of/IN ( NP this/DT reflected/JJ glare/NN light/NN ) ) ./.

The final syntactic relation to be identified in this section is the reduced subordinate clause (CO-RSUB), a close relative of CO-SUB. Figure 5.12 presents the CO-RSUB automaton. Unlike CO-SUB, there is no subject in the reduced clause for CO-RSUB to identify.

![Figure 5.12 Reduced Subordinate Clause Enclosed by Commas](image-url)
The clause must be introduced by the start of a sentence or a comma. The CO-RSUB automaton is more relaxed than CO-SUB because it refers only to the part of speech tags of a subordinate conjunction - IN and WRB. A comma (or start of sentence) followed by either of these tags and a present or past participle verb is a strong indication that a CO-RSUB is present. As with the reduced relative clause earlier in Figure 5.10, the verb phrase is not explicitly recognized since the automaton requires a particular verb tag to be present on arc CD. This once again accounts a possible particle tag (RP) to be consumed by arc DD.

5.5 Enclosing Automata

The Enclosing automata represent the final section of automata which identify comma-delimited syntactic relations. With the exception of independent clauses (CO-S), these comma-delimited syntactic relations usually do not contain verb clauses.

Time (or location) noun phrases (CO-TNP) are the first to be recognized. Figure 5.13 presents the CO-TNP automaton. This is a very specific, highly lexicalized automaton which only recognizes a time or location noun phrase that starts a sentence and is concluded by a comma. For example: Two weeks ago, ...; Ten seconds later, ...; Last Wednesday, ...; North of the river, ...; etc...

Figure 5.13 Time Noun phrases Enclosed by Commas
The reason for having a CO-TNP automaton is so that these noun phrases are not confused with appositions which are recognized next in the cascade by the automata in Figure 5.14. The first automata (CO-APS1) recognizes an apposition which is introduced by a comma and concluded by another comma or the end of the sentence. This apposition usually appositives a noun phrase that directly precedes it - recognized by arc AB. After the initial noun phrase of the apposition is recognized (on arc CD), several other syntactic relations can be contained by arc DD before the end of the apposition is reached. The second automaton (CO-APS2) recognizes an apposition that starts a sentence and appositives the noun phrase that directly follows it.

Figure 5.14 Appositions

The appositions in the following sentences were encountered during testing and correctly identified:

{ NP The/DT distinction/NN ) ( VP lies/VBZ ) ( PP in/IN ( NP the/DT fact/NN ) ) ( SUB that/IN ( NP realism/NN ) ( VP is/VBZ concerned/VBN ) ( ADV directly/RB ) ) with/IN ( REL2 what/WP ( VP is/VBZ absorbed/VBN ) ) ( PP by/IN ( NP the/DT senses/NNS ) ) ;/; ( CO-S ( NP naturalism/NN ) ) ,/; ( CO-APS ( NP a/DT term/NN ) ( REL1 more/RBR properly/RB applied/VBN ) ( PP to/TO ( NP}
There are certain situations in which a noun phrase that is enclosed by commas can incorrectly be identified as an apposition. By far the most common situation encountered is when a comma-delimited subordinate clause or prepositional phrase starts a sentence and is followed by the subject of the sentence which is then followed by another comma-delimited syntactic relation. For example, in the following sentence that was encountered during testing, the subject is mistaken for an apposition:

The easiest solution is to create a special purpose automaton that would be placed early in the cascade to recognize and prevent such patterns from incorrectly being identified.

Transitional phrases (CO-TRN) are identified next in the cascade by the automaton in Figure 5.15. A transitional phrase can be introduced by a comma or the start of the sentence and concluded by another comma. Transitional phrases of length two and three are recognized here, including such phrases as: for example, in addition, as a result, on the contrary, in other words, etc...
Figure 5.15 Transitional Phrases

The next automaton in Figure 5.16 recognizes prepositional phrases which are enclosed by commas (CO-PP). CO-TRN is placed directly before this automaton since syntactically CO-TRN is equivalent to CO-PP. CO-TRN is given precedence since it is a very specific, highly lexicalized automaton. Similar to CO-SUB in Figure 5.11 and CO-RSUB in Figure 5.12, a coordinate conjunction can be consumed in the beginning of a CO-TRN or CO-PP. As before, the conjunction is not assigned a structural tag.

Figure 5.16 Prepositional Phrases Enclosed by Commas

The final two automata of the comma network are shown in Figures 5.17 and 5.18 - coordinated independent clauses and coordinated noun phrases enclosed by commas. As previously noted in Chapter 4, the CO-S automaton should be placed at the very start of the cascade since it can contain every other syntactic relation. However, this would
require forward calls to every other automaton in the network. Computationally it was more desirable to place the CO-S automaton at this point at the cascade so that the right balance between forward calls and backward references could be achieved.

A coordinated sentence can be introduced by a conjunction, a comma and conjunction, or a semi-colon. In either case, an extra condition ensures that a verb phrase is in the preceding part of the sentence. The subject of the coordinated clause, which could be a noun phrase or list of noun phrases, is recognized by arc CD. Any pre-subject syntactic relations, such as CO-TRN, CO-SUB, or CO-PP, are consumed by the self arc CC. Post-subject modifiers or other syntactic relations are consumed by self arc DD. The main verb of the new clause is consumed by arc DE. Finally, any of the preceding or following automata in the cascade can be referenced by arc EE, before the End-Of-Sentence or a semi-colon is reached.

The following sentences are examples containing coordinated independent clauses that were encountered during testing and correctly identified:

```
( PP In/IN ( NP 1971/CD ) ) ( NP the/DT Progressive/NNS Conservatives/NNS ) ,/, ( CO-REL1 led/VBN ( PP by/IN ( NP Peter/NNP Lougheed/NNP ) ) ) ,/, ( VP were/VBD swept/VBN ) ( PP into/IN ( NP office/NN ) ) ,/, ( CO-S and/CC ( CO-PP in/IN ( NP 1982/CD ) ) ) ,/, ( CO-PP ( PP with/IN ( ADV only/RB ) ( NP three/CD ) ) ( PP of/IN ( NP its/PRP$ members/NNS ) ) ( REL1
```
It is important that the CO-S automaton precede the CO-NP (in Figure 5.18) so that the subject of an independent clause is not recognized as a CO-NP. For example, the following sentence contains a CO-S that would be mistaken as a CO-NP had the automaton ordering been switched: *The man, who lost his baggage, was stranded on the airport.* [CO-S and the woman, who only had a carry-on, is now at her hotel.]

**Figure 5.18 Coordinated Noun Phrase Enclosed with Commas**
5.6 Comma Tagging

Another way to approach the isolated task of interpreting commas is to assign structural tags to only the commas in the sentence – *comma-tagging* (van Delden and Gomez, 2003b and 2002). This is a simpler task since: (1) boundary identification of syntactic relations is no longer a factor; and (2) the ordering of the automata is irrelevant because a co-occurrence matrix can be automatically learned which identifies incorrect comma tags.

The automata of the comma network could be used *as is* to accomplish this task. However, there are some issues that need to be addressed. First, structural tags are only assigned to the commas that are parsed by the automata. Second, since contiguous blocks of texts are no longer tagged, "-BEG" and "-END" suffixes are added to the base tags of delimiting commas, instead of using the ">" notation which would no longer be appropriate. Third, some non-comma automata, such as the relative clause (REL) automata, noun phrase (NP) automata and others, would still need to be made accessible, since they are called by the comma network. This could be avoided by simply replacing the automata calls by part-of-speech tags since there is no need to recognize other non-comma syntactic relations. The complete set of comma-tags is offered later in Table 5.2 where it is extended to the Dutch Language.

An important aspect of the comma is that it can delimit numerous syntactic relations simultaneously. For example, *In the Fall of 1992, a great year for sports, my favorite team won the World Series.* Here the first comma concludes a prepositional phrase, but also introduces an apposition. If only the commas are being tagged, the system must have the ability to assign multiple tags to a single comma. The comma automata would assign
intermediate comma tags to each comma in the sentence. The tags from one automaton does not interfere with the acceptance of another automaton. The commas are usually over-tagged, since the automata are independent of each other. For example, consider the sentence: *John likes apples, oranges, and bananas.* Here the commas coordinate a series of noun phrases, but without any knowledge of the meaning of the words, they could very well be enclosing an apposition, i.e. *oranges* appositives *apples.* The automata will determine that the commas could either be coordinating a series of noun phrases or enclosing an apposition. The final decision is left to co-occurrence matrix.

5.6.1 The Co-Occurrence Matrix

After all the possible tags for a comma have been assigned by the automata, a co-occurrence matrix is considered to determine which commas are valid. The structure of the co-occurrence matrix is shown in Figure 5.19.

![Figure 5.19 The Structure of the Co-occurrence Matrix.](image)

The comma-tags are placed at the head of every column (*column-tag*) and the start of every row (*row-tag*). There are three possible values at an intersection of a column and row: ‘1’, ‘0’, or ‘>’. A ‘1’ at a column-row intersection indicates the column-tag and
row-tag can co-occur, and a ‘0’ means they cannot. The ‘>’ symbol is a special push-forward, which means if this column-tag is present with this row-tag, it should be moved (or pushed forward) to the next comma in the sentence. For example: *Fruit, including apples, oranges, and bananas, are a healthy source of vitamins.* The relative clause automaton is not able to determine that the comma in front of *oranges* is a list comma and therefore assigns it as the end of the relative clause. The following tags would be assigned by the relative clause and list-of-noun-phrases automata: *Fruit, (CO-REL-BEG) including apples (CO-REL-END CO-LST-NP), oranges (CO-LST-NP), and bananas, (UNDETERMINED) are a healthy source of food.* The push forward function would move the CO-REL-END tag to the third comma in the sentence. When the third comma is then inspected, the CO-REL-END tag would then be moved to the fourth comma, where it would remain. In the case of *I eat all kinds of fruit, including apples, oranges, and bananas.*, the CO-REL-END would be pushed completely out of the sentence.

This method works very well for relative clauses, as well as for appositions and prepositional phrases. The algorithm that reads the matrix is shown in Figure 5.20:

```
FOR EACH comma's set of possible comma-tags
    FOR EACH row_i in the matrix
        IF row-tag appears in current comma tag set
            FOR EACH column_j intersecting row_i
                IF intersection-value_{i,j} is 0
                    remove column-tag_j from possible tags
                ELSE IF intersection-value_{i,j} is >
                    move this column-tag_j to the next comma in the sentence
```

Figure 5.20 Co-occurrence Matrix Processing Algorithm
Creating this matrix can be very time consuming, but fortunately this process can be entirely automated. The actual values in the matrix (including the *push-forward* operator) can be automatically generated by recording which comma-tags co-occur in a comma-tagged corpus. A threshold can also be used here to eliminate commas that co-occur very infrequently.

A greedy learning algorithm can then be used to automatically determine the correct order of the rows in the matrix. The following files are needed: the manually comma-tagged corpus (CT) and the intermediate output of the automata on the same corpus (ICT) – the commas still have all the intermediate tags associated with them. The learning strategy is shown in Figure 5.21:

```
- determine the values of each row in the matrix by noting co-occurrences in CT.
- Initially the co-occurrence matrix(M) is empty.
- WHILE all rows have not been added to M:
  - FOR EACH remaining rowi:
    - add rowi to M and apply the Phase I algo. using M on ICT
    - Keep track of the number of errors that occurred by adding rowi (An error is recorded when a row-tag incorrectly removes a column-tag from the correctly tagged corpus CT).
    - The row that produced the least amount of additional errors is added to M.
```

Figure 5.21 The Greedy Learning Algorithm

### 5.6.2 Evaluation

Fifty random articles were chosen from the five previously mentioned sources in Chapter 3 (see Appendix B for complete article list) for testing. The system was also tested on Section 23 of the Wall Street Journal Penn Treebank III. This test data was not used in any way to design the automata. The results are presented in Table 5.1 below.
## Table 5.1 Comma Tagging System Test Results

<table>
<thead>
<tr>
<th>Sources</th>
<th>number articles</th>
<th>Tokens</th>
<th>avg. sen. length</th>
<th>number commas</th>
<th>Rule-based Tagger</th>
<th>Finite-State Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>WorldBook</td>
<td>10</td>
<td>7258</td>
<td>19</td>
<td>340</td>
<td>64%</td>
<td>92%</td>
</tr>
<tr>
<td>New York Times</td>
<td>10</td>
<td>9898</td>
<td>26</td>
<td>443</td>
<td>51%</td>
<td>91%</td>
</tr>
<tr>
<td>Wall Street Journal</td>
<td>10</td>
<td>12926</td>
<td>26</td>
<td>538</td>
<td>51%</td>
<td>92%</td>
</tr>
<tr>
<td>Britannica</td>
<td>10</td>
<td>8696</td>
<td>29</td>
<td>458</td>
<td>49%</td>
<td>92%</td>
</tr>
<tr>
<td>Encarta</td>
<td>10</td>
<td>12996</td>
<td>25</td>
<td>773</td>
<td>55%</td>
<td>92%</td>
</tr>
<tr>
<td>Penn Treebank 3</td>
<td>Sec. 23</td>
<td>39983</td>
<td>27</td>
<td>2512</td>
<td>56%</td>
<td>95%</td>
</tr>
</tbody>
</table>

The results show that the set of automata performs well regardless of corpus type. The 3% better performance on the Penn Treebank III was expected, because of its higher word tag accuracy. As a baseline to which our results can be compared, Brill’s tagger (Brill, 1994) was trained on each corpus and also applied to the test data. The performance of the automata is considerably higher than that of the rule-based tagger. The reasons for this low performance is two fold: (1) unlike the automata, the rule-based tagger cannot capture a large enough context to determine a comma’s meaning, and (2) the rule-based tagger cannot assign two or more tags to a token, which is required for commas.

Some erroneous situations merely call for an addition of a new tag to the arc of an automaton, in which case the automaton can easily be updated. Some errors are made by the co-occurrence matrix, removing the correct tag from the list of possible tags and leaving an incorrect one. This occurs when a situation arises that is opposite to what was learned by the greedy algorithm in Section 5.6.1. These errors can be reduced by training the algorithm on a larger corpus, or manually making a change to the co-occurrence matrix.
Some errors, however, cannot be fixed so easily, for example, consider the sentence: 

*Many of the executives were present at the meeting: John Smith, the CEO of the company, Elizabeth Ray, the vice president, and Harold Johnson, the head engineer.* The list of noun phrase automaton will be unable to identify this list because of the appositions that are enclosed with commas and located inside of the list. There is no simple way to solve this problem with this comma-tagging technique. The one solution is to design a more complex special purpose automaton to identify such lists. Note that without any other information, it is most likely that *John Smith* is *the CEO of the company* and *Elizabeth Ray* is *the vice president.* But this does not necessarily have to be true. More information about the company and its employees may indicate that *John Smith* is indeed not the *CEO of the company.* Decisions are made here based on the sentence level information that is provided – surrounding context is not considered.

Elliptical constructions will also cause the automata to make incorrect assignments, for example: 

*For example,/CO-PP-END Athens was famous for its decorated pottery,/CO-APS-BEG Megara for woolen garments,/CO-APS-END and Corinth for jewelry and metal goods.* The omission of the phrase *was famous from Megara for wooden garments* causes the apposition FSA to identify *Megora for wooden garments* as an apposition. A detailed analysis of the entire sentence is needed and is beyond the capabilities of this approach.

Few errors occur as side effects of the automata. A side effect is an error that is caused by an automaton, but is tolerated because the automaton is, in general, very accurate. For example, a comma that coordinates two adjectives is identified by the NP automaton as being part of a noun phrase. Although the accuracy of this automaton is
very high, it can produce an error in some situations. For example, *Even though the boy is happy, many teachers feel he would be better off in a different class.* The comma in *happy, many teachers* is incorrectly identified as coordinating two adjectives that are part of a noun phrase, when actually it is concluding a subordinate clause.

Despite these errors, the technique described here performs well on correctly tagged real world texts with a 95% accuracy, failing only on certain quite complex sentences. In addition to simple sentences, many complex sentences are handled very nicely. This section is concluded by listing a few of these sentences, which were taken from the Evaluation data (see article list in Appendix B).

Nurse sharks, slow-moving sharks that live mostly in warm, shallow water, grow to more than 4 m, or 13 ft.

Despite all the study, the problems have endured in Cincinnati, which, like Los Angeles, New York and other American cities, has had recurring racial problems involving its police force.

A black man is killed, an investigation is conducted, hearings are held, a report is written and then promptly forgotten.

The 1995 report by the city manager's review panel, which urged a renewed commitment to diversity in hiring, promotions and training, warned against lip service.

The woman, frightened, complied.

When viewed from above, their darker dorsal sides are difficult to distinguish from the ocean depths, and when viewed from below, their lighter ventral sides blend with the sunlit water above them.

5.7 Extension to Dutch

Here the feasibility of extending this comma-tagging approach to the Dutch natural language is analyzed. The following questions are answered:
- Are commas used to delimit a similar set of syntax relations in the Dutch language?

- If a comma-delimited syntactic relation occurs in both English and Dutch, is the syntax of the usage exactly the same?

- Does this finite state comma-tagging approach perform well on the Dutch language?

- How much effort is needed to extend this approach to Dutch?

Three levels of modification are defined to adapt the English comma-tagging automata to Dutch:

1. No modification at all, the automata can be used as is.

2. Translation of lexicalized arcs.

3. Re-organization of the automata due to the syntactic differences between English and Dutch.

Some syntactic relations can be recognized by the same automata in both English and Dutch because their part-of-speech tag patterns are similar. In such cases, no modification is needed to the English automata that recognize such syntactic relations.

Some automata, however, have been lexicalized to improve performance. Lexicalized automata cannot directly be employed by another language. A simple translation of the lexical term(s) that is (are) assigned to a transition is needed. However, a common problem in machine translation is that one lexical term in a language may result in two or more terms in another language. For example, an automaton is lexicalized to recognize that, if a sentence starts with *For example*, then *For example*, is definitely a transitional phrase that is being concluded by a comma. This phrase, however, would be translated to
the single word phrase *Bijvoorbeeld*, in Dutch. Simple translation of lexicalized arcs will not always suffice, some arcs may need to be expanded or collapsed.

The ordering of syntactic relations varies greatly in the English and Dutch languages. For example, Dutch prefers time, manner, place as in: *Hij gaat morgen (time) met zijn vrouw (manner) naar Leiden (place)*. While English prefers place elements before time elements: *He is going to Leiden tomorrow with his wife*. Verb syntax also varies greatly from English to Dutch when an auxiliary verb or modal is present, for example: *I must go to Leiden tomorrow*. Depending on style, this sentence is translated to Dutch as *Morgen moet ik naar Leiden gaan* or *Ik moet morgen naar Leiden gaan*, which translates directly back to English as *Tomorrow must I to Leiden go* or *I must tomorrow to Leiden go*, respectively. In some cases, an automaton, which captures the new syntactic structure introduced by the Dutch Language, must be created to supplement the existing English automaton. In other cases, the English automaton itself must be modified because its syntactic structure does not exist in Dutch.

The two-step approach to comma tagging is also desirable in Dutch. As in English, a single comma in Dutch can play more than one role. Furthermore, an empirical analysis reveals that co-occurrences in English are almost equivalent to those in Dutch. Co-occurrences are not *exactly* equivalent because some English comma-tags do not exist in Dutch. This phenomenon, however, does not adversely affect the performance of the matrix since the extra co-occurrence information is simply not used in Dutch. Therefore a matrix that is learned from an English corpus can be directly used by a Dutch comma tagger - no conversion work is necessary.
The two-step finite state approach seems to be a viable method for tagging commas in Dutch, but are commas being used to delimit or coordinate a similar set of syntactic relations in the Dutch language? Table 5.2 shows the set of comma tags that have been defined for the English language with a subscript that indicates the level of modification needed to extend the associated English comma-tagging automata to Dutch (see Section 5.7.1-5.7.3 for more details).

This tag set can be directly used in the Dutch comma tagger. The only exception of a comma that is not used in Dutch is the one that precedes the year part of a date. In English, day follows month in a date and usually has a comma before the year, i.e. January 15, 1997. In Dutch, however, month follows day and does not take a comma before the year part, i.e. 15 januari 1997. The date comma therefore does not exist in the Dutch language.

**Table 5.2 Comma Tags with Level of Modification Indicators.**

<table>
<thead>
<tr>
<th>Group</th>
<th>Tag</th>
<th>Possible Suffixes</th>
<th>Description(coordinates or delimits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Series</td>
<td>CO-LST(_{1,3})</td>
<td>-NP(_3), -VC(_3), -INF(_3)</td>
<td>Series of noun phrases; verb or infinitival clauses</td>
</tr>
<tr>
<td>Enclosing</td>
<td>CO-TNP(_2)</td>
<td>-BEG, -END</td>
<td>Time Noun Phrase</td>
</tr>
<tr>
<td></td>
<td>CO-APS(_1)</td>
<td>-BEG, -END</td>
<td>Apposition</td>
</tr>
<tr>
<td></td>
<td>CO-PP(_1)</td>
<td>-BEG, -END</td>
<td>Prepositional phrase</td>
</tr>
<tr>
<td></td>
<td>CO-NP(_1)</td>
<td>-BEG, -END</td>
<td>Coordinated noun phrase enclosed by commas</td>
</tr>
<tr>
<td></td>
<td>CO-TRN(_2)</td>
<td>-BEG, -END</td>
<td>Transitional phrase</td>
</tr>
<tr>
<td>Clausal</td>
<td>CO-REL(_3)</td>
<td>-BEG, -END</td>
<td>Relative clause</td>
</tr>
<tr>
<td></td>
<td>CO-INF(_3)</td>
<td>-BEG, -END</td>
<td>Infinitival clause</td>
</tr>
<tr>
<td></td>
<td>CO-SUB(_3)</td>
<td>-BEG, -END</td>
<td>Subordinate clause</td>
</tr>
<tr>
<td></td>
<td>CO-S(_2)</td>
<td>-BEG, -END</td>
<td>Independent clause or new sentence</td>
</tr>
<tr>
<td></td>
<td>CO-VC(_3)</td>
<td>-BEG, -END</td>
<td>Verb clause</td>
</tr>
<tr>
<td></td>
<td>CO-RSUB(_3)</td>
<td>-BEG, -END</td>
<td>Reduced subordinate clause</td>
</tr>
<tr>
<td>Speech</td>
<td>CO-DIR(_2)</td>
<td></td>
<td>Direct speech</td>
</tr>
<tr>
<td></td>
<td>CO-IDIR(_2)</td>
<td></td>
<td>Indirect Speech</td>
</tr>
</tbody>
</table>
There is also a difference in the types of syntactic relations that are assigned the CO-RSUB and the CO-REL tags. In English, reduced subordinate and relative clauses are missing the relative pronoun/determiner and auxiliary verb. The introductory verb of the clause is a present or past participle verb. For example: *While walking to school, he met his friend.* or *If opened, the box will explode.* In Dutch, however, there is no present participle verb form--- *ik zing* could mean *I sing* or *I am singing* depending on context. A progressive state could also be indicated with a helper verb and the main verb in infinitival form, for example: *He is sleeping* --- *Hij ligt (lies) te slapen (sleep).* Only the second example sentence above can be directly translated to Dutch: *Indien opengemaakt, gaat de doos ontploffen.* The first sentence could be translated as: *Terwijl (while) hij (he) naar school loopt (walk), ontmoette (met) hij zijn vriend,* but would never occur in the reduced form possible in English. The CO-RSUB tag and CO-REL tag (when used for reduced relative clauses) are, therefore, only assigned to reduced clauses introduced by a past participle verb.

A similar case can be made for appositions. Appositions often occur in the Dutch written language. However, in English, an apposition, that is concluded by a comma, can start a sentence and appositive the noun phrase that follows it. For example: *The best student in the class, John went to the black board.* *The best student in the class* appositives the noun phrase that follows it, *John.* In Dutch, if a syntactic relation other than the subject of the sentence starts the sentence, then it is usually followed first by the main verb and then the subject of the sentence. In Dutch, it would not sound right to place the subject directly after the apposition. Therefore such sentences are not usually
directly translated to Dutch. The apposition is placed after the noun phrase that is being appositive: *John, de bestte student in de klas, ging naar het bord.*

### 5.7.1 No Modification

Some of the comma tagging automata that were developed for the English language can be directly used without modification. The comma-tags assigned by these automata are used to delimit or coordinate: lists of noun phrases (CO-LST-NP), appositions (CO-APS), prepositional phrases (CO-PP), and coordinated noun phrases (CO-NP).

These automata are not lexicalized and do not contain verb clauses, For example, an automaton designed to recognize a list of noun phrases will do so based on part-of-speech information, regardless of the language: *I must speak with my parents, the teacher and the director of the school at once.* - *Ik moet meteen met mijn ouders, de leraar en de directeur van de school praten.* Even though syntactic order varies dramatically in this sentence from English to Dutch, the CO-LST-NP automaton shown earlier in Figure 5.7 in Section 5.3 will tag both of these commas correctly. Similar cases can also be made for the other automata that do not need modification.

### 5.7.2 Translation of Lexicalized Arcs

Some of the comma tagging automata that have been lexicalized cannot be directly applied to the Dutch language without some minor modifications. Four types of tags belong to this group: transitional phrases (CO-TRN), time noun phrases enclosed in commas (CO-TNP), indirect (CO-IDIR) and direct speech (CO-DIR). Extending the automata which assign these tags to Dutch, simply calls for a translation of the
lexicalized transitions, adding or removing arcs when necessary. For example, recall the
time noun phrase automata in Figure 5.13 of Section 5.5. This automata recognizes that
Two weeks ago, or Twee weken geleden, is a time noun phrase that starts a sentence. The
arc BC is taken only if a particular time or location word is at that position in the
sentence. In English these words include: today, hour(s), day(s), week(s), month(s),
year(s), north, south, etc. To apply this automaton to Dutch, these words must be
translated: vandaag, uur (uren), dag(en), week (weken), maand(en), jaar (jaren), noord,
zuid, etc. A similar case can be made for the remaining syntactic relations in this section.

5.7.3 Syntactic Re-organization

Most complications result from differences in verb syntax from English to Dutch. In
these cases, a re-organization of the automata is necessary for classification to be
accurate. The comma tags affected here delimit or coordinate the following clauses: lists
of verb and infinitival clauses (CO-LST-VC and CO-LST-INF), verb clauses (CO-VC),
relative clauses (CO-REL), subordinate clauses (CO-SUB), infinitival clauses (CO-INF),
and independent clauses (CO-S). New automata need to be defined by re-arranging the
arcs and possibly adding new states and transitions. Some of the English automata are
still used, but require an extra automaton to handle the new possible syntax introduced by
Dutch.

Non-comma automata that are called by the comma automata may need some re-
organization as well. Infinitival and relative clause automata are called regularly by the
comma automata. For example, In her haste to leave the store, Emma forgot her purse. -
In haar haast de winkel te verlaten, vergeet Emma haar portemonnaie. Te verlaten is recognized by the INF automaton which is called by the CO-PP automata.

Infinitival clause groupings can be accomplished directly by the English infinitival clause automaton. One of the relative clause automaton (REL2), however, requires an extra complication due to the different ordering of verb complements in English and Dutch. For simple past and present tense where no auxiliary or modal is present, the English relative clause automaton will work in the Dutch language. For example, *I bought a radio, a television that I returned this morning and a video machine. - Ik kocht een radio, een televisie die ik vanmorgen terugbracht en een video.* The first verb phrase after the relative determiner is identified as the introductory verb phrase of the relative clause. However, if an auxiliary verb or a modal is present, the verb phrase can be divided by post verbal noun or prepositional phrases. Consider the following example, *I bought a radio, a television that I must return to the store and a video machine. -Ik kocht een radio, een televisie die ik moet terug naar de winkel brengen en een video.* Note that in Dutch the prepositional phrase *naar de winkel* can be placed in between the modal and the main verb. The relative clause automaton must be extended to handle this possibility. In English, adverbs are commonly placed in between an auxiliary or modal and the main verb, but very seldom noun or prepositional phrases, so this extra complication is not needed in English. Figure 5.22 shows the relative clause automaton in English (top) and the new automaton that is added for Dutch (bottom).
An extra automaton is added instead of complicating the original one. Both automata are used in Dutch. When applying these automata, the new automata in Figure 5.22 (bottom) would be applied prior to the old one (top). If applied in the opposite order, the old automaton would still accept, inhibiting the new one. To revisit the previous example, the following words would be grouped by this automaton: *Ik kocht een radio, een televisie [REL die ik moet terug naar de winkel brengen] en een video machine.*

The automata that recognize commas enclosing relative and subordinate clauses also need modification. Figure 5.23 shows the original automata used in English to recognize commas that enclose relative clauses and the new set needed for Dutch. The two automata on the left side of Figure 5.23 (which were previously discuss in Section 5.4) are used to tag commas enclosing relative clauses in English.

The first automaton in the upper left half is still used in Dutch, and is supplemented by the first automaton in the upper right half. This new automaton also identifies commas that enclose a reduced relative, but the past participle verb is located at the end of the clause - which does not occur in English. For example: *The method, also called smelting, takes 2 hours.* - *De methode, ook smelting genoemd, duurt 2 uren.* The VBG tag remains
on arc $BC$ in the English automaton, but will never be used since, as previously mentioned, this Dutch does not support this construction.

The second automaton in the lower left half of Figure 5.23 is replaced by the automaton in the lower right half because, in Dutch, prepositional compounds replace phrasal relative pronouns or determiners. For example, *of which* - *waarvan*, *in which* - *waarin*, with *which* - *waarmee*, *upon which* - *waarop*, etc. These prepositional compounds simplify the implementation of the second automaton while preserving its capabilities: *I read 10 books, the first of which was interesting - Ik las 10 boeken, waarvan de eerst interessant was.*
Lists of verb clauses can be syntactically similar in English and Dutch, as in: *I kicked the ball to Jan, ran left towards the goal and waited a few minutes.* – *Ik schopte de ball naar Jan, rende links naar de goal en wachtte voor een paar minuten.* However, because of the new verb syntax capabilities in Dutch, additional automata must also be defined here. For example: *I must read the book, sell my bicycle and find my notebook.* – *Ik moet het boek lezen, mijn fiets verkopen en mijn schrift vinden.* If a modal or auxiliary verb is present, the main verb appears at the end of each verb clause in the sentence.

Furthermore, it needs to be noted that in Dutch a list of verb clauses can be very similar to a list of infinitival clauses. In English, *to* always precedes an infinitival clause, making it easier to distinguish an infinitival verb from a main verb. In Dutch, this distinction is not made in a list of infinitival clauses, for example: *I want to read the book, sell my bicycle and find my notebook.* – *Ik wil de boek lezen, mijn fiets verkopen en mijn schrift vinden.* Note that this Dutch sentence is syntactically almost equivalent to the previous one above - the only difference is the use of an auxiliary verb instead of a modal. If a modal is present, then this is a list of verb clauses, otherwise if there is an auxiliary verb present, this is a list of infinitival clauses.

Similar to lists of verb clauses, when a single verb clause is being coordinated by a comma and a conjunction, the syntax can also differ dramatically from English to Dutch when a modal or auxiliary verb is present. In English the introductory verb phrase is always at the beginning of the clause. In Dutch, however, it is possible for the main verb to be located at the end: *I have read the book, and seen the movie.* – *Ik heb de boek gelezen, en de film gezien.*
Purpose infinitival clauses are also written differently in Dutch as compared to English. The place-holder word *om* which translates to *in order to* is almost always used in Dutch. It is followed by post verbal noun and prepositional phrases and finally the infinitive. For example: *To climb the mountain, John must first buy good shoes.* - *Om de berg te beklimmen, moet John eerst goede schoenen kopen.* As with the relative clause automata, changes must be made to all of the automata in this section in order to capture the new syntactic structures possible in Dutch.

5.7.4 Evaluation

Brill's tagger was used to assign part-of-speech tags to the words in the test sentences. The tagger had been trained on a section of the Eindhoven Corpus (Boogaart, 1975) by Edwin Drenth and was available for download. The tagset used is similar to the WOTAN and WOTAN-II tagsets (van Halteren, 1999; Zavrel and Daelemans, 1999). The tags were translated to the Penn Treebank Tagset which is used by the arcs of the automata in this approach.

The comma tagging system was tested on the same section of the Eindhoven corpus, used by Edwin Drenth to train Brill's tagger, and on random articles taken from Rotterdam's online newspaper and three online encyclopedias:

- Sterrenkunde (www.astro.uva.nl/encyclopedie)
- Gezondsheid (www.gezondstegids.nl)
- Eletrotechniek (www2.ele.tue.nl/encyclopedie)
No changes were made to the automata once testing began. Results are shown in Table 5.3. Improper usage of commas in sentences were not included in the analysis. Interestingly, we found incorrect placement of commas to be similar in the English and Dutch languages. For example, *The tree that had stood for over a hundred years, was blown over by the hurricane.* - *De boom die voor honderd jaar stond, was omgewaaid door de orkaan.* In this example, a comma is used to conclude a relative clause but there is no introductory comma. This comma is therefore classified as being incorrectly used and would have not been included in the analysis.

**Table 5.3 Dutch Comma Tagging Results.**

<table>
<thead>
<tr>
<th>Source</th>
<th>Avg. Sentence Length</th>
<th>Number Commas</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eindhoven Corpus</td>
<td>20</td>
<td>6246</td>
<td>94.5%</td>
</tr>
<tr>
<td>S-Encyclopedia</td>
<td>27</td>
<td>244</td>
<td>93.4%</td>
</tr>
<tr>
<td>G-Encyclopedia</td>
<td>17</td>
<td>72</td>
<td>93.5%</td>
</tr>
<tr>
<td>Rotterdam Newspaper</td>
<td>22</td>
<td>62</td>
<td>95.2%</td>
</tr>
<tr>
<td>E-Encyclopedia</td>
<td>17</td>
<td>51</td>
<td>92.0%</td>
</tr>
</tbody>
</table>

Overall, the Dutch system performed about the same as the English version of the system. Typical errors that are made due to the limitations of the automata when tested in English were also made here (see Chapter 8 and Section 5.6.2 for more details). For example, consider the following sentence that was encountered during testing: *For stream 1 you use, for example, light green, and for stream 2 red.* - *Voor stroom 1 gebruik je bijvoorbeeld groen licht, en voor stroom 2 rood.* The system would identify the last comma as coordinating a prepositional phrase, when actually an independent clause containing an elliptical structure is being coordinated: *and for stream 2 you use red* - en
voor stroom 2 gebruik je rood. A detailed analysis of the entire sentence is needed to recognize elliptical structures and is beyond the scope of this finite-state approach.

A new source of ambiguity was, however, introduced by the Dutch Language that is not encountered in English – distinguishing between certain coordinated verb clauses and independent clauses. Consider the following sentence that was encountered during testing: *Opposing loads pull at each other, and so electric forces hold the whole world together.* - *Tegegestelde ladingen trekken elkaar aan, en zo houden elektrische krachten de hele wereld bij elkaar.* In this sentence, an independent clause is being introduced by a comma and a conjunction. Notice that in Dutch the verb is placed before the subject in the independent clause - *houden elektrische krachten.* This creates an ambiguity when a verb clause is being introduced and not an independent clause, for example: *Opposing loads pull at each other, and so hold the whole world together.* - *Tegengestelde ladingen trekken elkaar aan, en zo houden de hele wereld bij elkaar.* Verb sub-categorization knowledge must be considered to realize that the verb *houden* (hold) is transitive and cannot take two noun phrase objects. Unlike the WOTAN tagsets, however, the Penn Treebank tagset used here does not include this information. Had the WOTAN tagset been used in our system, the automata could have been modified to more accurately handle such situations.

Finally, this section is concluded by listing some example sentences encountered during the evaluation to illustrate the complexity of the sentences that can be correctly handled by this approach (The English translations here illustrate how the commas are being used and attempt to preserve the Dutch syntax).

The presence began to be noticed even outside of the building, in order words through the antennas on the roof, the call signs and the name of the club in front
of the windows, and through the longwire-antenna that runs from the roof of E-High to E-Low.

Zelfs buiten het gebouw valt de aanwezigheid al te merken, onder andere door de antennes op het dak, de roepletters en de naam van de club voor de ramen, en door de langdraad-antenne die van het dak van E-Hoog naar E-Laag loopt.

That is immediately more friendly to the environment, because satellites remain, after they have served their purpose, as rubbish in Space.

Dat is meteen milieuvriendlijker, want satellieten blijven, nadat ze dienst hedden gedaan, als afval in the ruimte achter.

If you pull your jacket over your recently washed, dry hair, you can even see the sparks right in front of your eyes.

Als je je trui over je net gewassen, droge haren trekt, kun je de vonkjes zelfs vlak voor je ogen zien.

From the moment of this “Big-Bang”, the origins of which we still do not precisely know, the universe started to expand and cool off, to the condition that we are currently in.

Vanaf het moment van deze "oerexplosie", waarvan men nu nog niet weet hoe deze precies heeft plaatsgevonden, is the heelal gaan uitdijen and dus gaan afkoelen, tot de toestand waarin wij het nu zien.

5.8 Related Research

Bayraktar et al. (1998) presents a detailed analysis of comma usage in the Wall Street Journal Section of the Penn Treebank corpus. A classification of structural usages is given which is similar to the structural tags presented in this work: elements in a series; sentence initial elements; sentence final elements; non-restrictive phrases or clauses; appositives; interrupters; and quotations. Syntactic patterns associated with each type of comma have been extracted from the corpus and the most frequently occurring ones are
included in an appendix at the end of the paper. The frequencies of each type of comma is also noted. Although the classification differs from the work presented here, the frequencies resemble those presented earlier in Table 3.2, for example:

<table>
<thead>
<tr>
<th>Elements in a Series:</th>
<th>Bayraktar et al. (1998)</th>
<th>This work (Table 3.2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentence Initial Phrases + Nonr. Phrases):</td>
<td>20.3%</td>
<td>19.8%</td>
</tr>
<tr>
<td>(similar to CO-PP and CO-TRN here)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sentence Initial Clauses + Nonr. Clauses):</td>
<td>19.8%</td>
<td>19.3%</td>
</tr>
<tr>
<td>(similar to CO-REL + CO-SUB here)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Appositions:</td>
<td>14.9%</td>
<td>23.6%</td>
</tr>
<tr>
<td>Quotations:</td>
<td>26.1%</td>
<td>16.5%</td>
</tr>
<tr>
<td>(Direct speech)</td>
<td>4.5%</td>
<td>2.7%</td>
</tr>
</tbody>
</table>

This, however, is a very rough comparison, since the classification schemes do not match. Also, the frequencies presented by this work were acquired from different sources across multiple domains. The frequencies by Bayraktar et al. (1998) are from a business journal alone.

The classification scheme of Bayraktar et al. (1998) was not used in this work because a more detailed description of the comma-related syntactic relations was desired. For example, an *Interrupter* in their classification could be referring to an adverbial phrase, a subordinate clause, a transitional phrase, or prepositional phrase that is breaking the flow of the sentence. Instead of identifying the syntactic relation as an interrupter it was preferred in this approach to better identify what the interrupter is - warranting the more detailed categories listed above.

Furthermore, Barakter et al. (1998) does not offer any type of approach to recovering the syntactic categories of the commas, only frequencies and their patterns are described.
On the contrary, two methods are presented in this work: (1) a larger-first cascade of finite automata which recover comma-related patterns and use the comma information to contain attachment ambiguity, and (2) a comma-tagging approach which combines finite state automata and a greedy learning algorithm.

Beeferman et al. (1998) describe a different comma-related task: a lightweight punctuation annotation system for speech called CYBERPUNC. The system attempts to restore punctuation marks such as commas, in speech that is dictated to it. Instead of using pauses in the acoustic stream, the probabilistic system relies only on lexical information. The comma frequencies and patterns from the Penn Treebank corpus are compiled so that a probabilistic tri-gram model can be used to restore the comma punctuation marks.
CHAPTER 6

CONJUNCTION NETWORK

Coordination is a pernicious source of structural ambiguity. - Philip Resnik

Identification of the appropriate conjuncts of the coordinate conjunctions in a sentence is fundamental to the understanding of the sentence. - Rajeev Agarwal and Lois Boggess

The automata that make up the conjunction network are presented in this chapter. First, the automata which fully disambiguate coordinated clauses are presented. The automata are of simple design. It is shown how the automata can be extended to avoid making errors in certain sentence constructions. Problematic examples are also discussed. The automata which partially disambiguate coordinated phrases are described next in Section 6.2. Refer to the Abbreviation List on page xiv and Section 4.4 for descriptions of the labels that are placed on the arcs of the automata. Also, refer to Appendix C for the complete automata cascade.

Some related research on conjunction disambiguation is presented next which serves as the foundation for the final section of the chapter - a hybrid approach to pre-conjunct identification that was developed along side this work.
6.1 Coordinated Clause Automata

Figures 6.1-6.5 present the automata that fully disambiguate coordinated clauses. These are non-comma-delimited clauses. Comma-delimited clauses are identified earlier in Section 5.4. The automata are all of simple design since they did not produce any errors during testing. However, incorrect groupings are possible with these automata. Consider the coordinated relative clause automaton (CC-REL) in Figure 6.1. If two relative clauses in a sentence are separated only by a coordinate conjunction, then it is very likely that these clauses are in coordination, especially if both are introduced with a relative determiner/pronoun. For example, the following sentence was encountered during testing:

```
( NP Various/NNP Semitic/NNP peoples/NNS ) ( CC-REL ( REL2 who/WP ( VP were/VBD influenced/VBN ) ) ( PP by/IN ( NP Sumerian/NNP culture/NN ) ) or/CC ( REL2 who/WP ( VP settled/VBD ) ) ) ( PP in/IN ( NP southern/JJ Mesopotamia/NNP ) ) ( VP adapted/VBD ) ( NP the/DT structures/NNS ) ( PP of/IN ( NP Sumerian/NNP religion/NN ) ) ( PP to/TO ( NP their/PRP$ own/JJ beliefs/NNS ) ) ) ( CC-NP and/CC ( NP practices/NNS ) ) ./. 
```

![Diagram of CC-REL automaton]

**Figure 6.1 Coordinated Relative Clauses**

If the relative determiner is omitted from the second relative clause, then it *less* likely that this verb clause is being coordinated with the preceding relative clause. However,
usually there is only one pre-conjunct site present and if there are multiple sites then usually the right most pre-conjunct site is the correct one. Consider the example sentence above once again. Had the second who been omitted from the sentence, the coordinated relative clause would still have been identified by the automaton in Figure 6.1 by calling the coordinated verb clause automaton (CC-VC) on arc BC: [ CC-REL [ REL2 who/WP [ VP were/VBD influenced/VBN ] ] [ PP by/IN [ NP Sumerian/NNP culture/NN ] ] [ CC-VC or/CC [ VP settled/VBD ]]]. This is a simple case since only one pre-conjunct site is present - were influenced. Now consider the following example: Peter just bought a computer [ CC-REL that has a fast processor [CC-VC and took it ] ] to school to show his friends. An incorrect grouping is made by the coordinated relative clause automaton because there are multiple pre-conjunct sites (bought and has) and the right most one (has) is not the correct pre-conjunct. Such errors can be corrected by verifying that the tenses of the coordinated verbs match - similar to how the list of verb clauses automaton in Figure 5.6 is designed. Notice in the above example that has is a third person singular verb while took is a past tense verb. Incorporating verb tense information would make all of the coordinated clause automata in this section more precise.

Coordinated subordinate clauses (CC-SUB) are recognized next by the automaton in Figure 6.2. The following sentence is an example encountered during testing in which this automaton correctly accepted:

(CC-SUB ( SUB Because/IN ( NP he/PRP ) ( VP became/VBD ) ( NP one/CD ) ) ( PP of/IN ( NP the/DT best/JJS lawyers/NNS ) ) ( PP in/IN ( NP Columbia/NNP County/NNP ) ) and/CC ( SUB because/IN ( NP the/DT county/NN ) ( VP was/VBD dominated/VBN ) ) ) ( PP by/IN ( NP the/DT Federalist/NNP Party/NNP ) ) ./. ( NP he/PRP ) ( VP was/VBD constantly/RB asked/VBN ) ( INF to/TO ( VP appear/VB ) ) ( PP in/IN ( NP court/NN ) ) ( PP against/IN ( NP Federalist/NNP lawyers/NNS ) ) ./.
Now consider the following example: *I saw Mary eat an orange and read a book.* This sentence is ambiguous, but the ambiguity is diverted to the part-of-speech tagger. If the tagger says that *read* is a present tense verb, then two coordinated subordinate clauses are grouped: *I saw [CC-SUB [SUB Mary eat/VB an orange] [CC-VC and read/TB a book] ].* If the tagger says that *read* is a past tense verb, then a single coordinated verb clause is identified: *I saw/VBD Mary eat an orange [CC-VC and read/VBD a book].*

However, some ambiguity simply cannot be resolved. Consider the following sentence: *We see/VB that the girls read/VB books and know/VB that the boys do/VB not.* Base verb tags(VB) would be assigned to each verb in the sentence, so the tagger does not resolve any ambiguity and verb tense information does not help in this case. Choosing the rightmost pre-conjunct is a possibility, but there is no way of knowing if this is actually the correct classification. Either grouping could be possible: *We see ... and (we) know ... or the girls read ... and (the girls) know ...*. Fortunately such sentences occur very infrequently.

Coordinated infinitival clauses (CC-INF) are identified next by the automaton in Figure 6.3. The following sentence is an example encountered during testing in which this automaton correctly accepted:
Verb tense agreement would not help to correctly determine the pre-conjunct site if the second to in the above sentence had been omitted. But since the rightmost pre-conjunct is the correct coordination site, the automaton in Figure 6.3 would be able to correctly identify the coordinated verb clause: [CC-INF [INF to/TO [VP avoid/VB] [NP out-of-home/JJ placements/NNS]] [CC-VC and/CC [VP reunify/VB] [NP children/NNS]]].

Coordinated gerunds, participle clauses (ing verb complements) or reduced subordinate clauses (ING) are identified next by the automaton in Figure 6.4. The following sentence is an example encountered during testing in which this automaton correctly accepted:

( NP The/DT broadcast/NN waves/NNS ) ( VP can/MD be/VB more/RBR efficiently/RB concentrated/VBN ) ( CC-ING by/IN ( ADV properly/RB ) ( VP arranging/VBG ) ( NP the/DT separate/JJ elements/NNS ) and/CC ( ING by/IN ( ADV properly/RB ) feeding/VBG ( NP signals/NNS ) ) ) ( PP to/TO ( NP the/DT elements/NNS ) ) ./.
Figure 6.4 Coordinated Gerund, Participle, or Reduced Subordinate Clauses

If the second preposition is omitted (by), verb tense information would make this automaton far more precise since only verb phrases with VBG tags would be coordinated. In this case, two coordinated reduced subordinated clauses are identified. However, coordinated gerunds are possible: [CC-ING [ING Hiking four miles [CC-VC and swimming three ] ] will be an incredible challenge; as well as ING complements: Peter keeps [ CC-ING [ING hoping for the best [CC-VC and preparing ] ] for the worst.

Finally, coordinated verb clauses (CC-VS and CC-VC) are recognized by the automata in Figure 6.5. The following sentences are examples encountered during testing in which these automata correctly accepted:

\[
( \text{NP Florida/NNP} ) ( \text{CC-VS (VP ratified/VBD ) (NP such/JJ ) (NP a/DT constitution/NN)} ) ( \text{PP in/IN (NP 1868/CD )} ) ( \text{CC-VC and/CC (VP accepted/VBD ) (NP the/DT Fourteenth/NNP Amendment/NNP )} ) ( \text{PP to/TO (NP the/DT Constitution/NNP )} ) ( \text{PP of/IN (NP the/DT United/NPP States/NNPS )} ), ( \text{CO-REL1 (VP guaranteeing/VBG) (NP civil/JJ rights/NNS) (PP for/IN (NP blacks/NNS))} ), ./.\]

\[
( \text{CO-PP (PP In/IN (NP some/DT species/NN))} ), (\text{CO-PP (PP (PIN1 such/JJ as/IN) (LST-NPI (NP guppies/NNS))}) , (\text{NP rockfish/JJ}) , (\text{NP females/NNS} ) ( \text{CC-VS (VP retain/VBP)} ) ( \text{NP the/DT eggs/NNS)} ) ( \text{PP in/IN (NP their/PRP$ bodies/NNS)} ) ( \text{CC-VC and/CC (VP accept/VB)} ) ( \text{NP sperm/NN)} ) ) ( \text{PP from/IN (NP males/NNS)} ) ./.\]
CC-VS encloses both verb clauses while CC-VC is called by CC-VS to enclose the second verb clause alone. Two automata are preferred so that CC-VC can also be used by the other automata in this and previous sections.

6.2 Coordinated Phrase Automata

The coordinated phrase automata are of simple design and are all presented in Figure 6.6. It is important that coordination within a noun phrase (CC-CNP) be resolved prior to coordinated noun phrases (CC-NP) since CC-CNP recognizes a larger and more specific pattern. Unlike coordinated verb clauses, a coordinated phrase usually has several possible pre-conjuncts and in most cases semantic information is required to determine the correct one. A noun phrase in coordination (CC-NP) is very common in written texts. The CC-NP automaton, therefore, accepted often during testing and the evaluation (see Chapter 8). The accuracy of this automaton is very high since the pre-conjunct (the hard part) is not determined and, at this point in the cascade, almost all of the other coordination ambiguities will have already been resolved. Even though simple, these
automata are helpful, being called by several preceding automata in the cascade and thus simplifying the overall complexity of the preceding automata.

![Diagram of coordinated phrase automata](image)

**Figure 6.6 Coordinated Phrase Automata**

### 6.3 Related Research

Unlike identifying the syntactic roles of commas, much work has been done in an attempt to classify conjunctions correctly (van Delden, 2002; Lefèva, 1998; Resnik, 1998 and 1995; Baker, 1995; Delisle and Szpakowicz, 1995; Agarwal and Boggess, 1992).

Agarwal and Boggess (1992) describe a now well-known algorithm that determines pre-conjunct size based primarily on semantic labels which have been assigned to the head nouns of the phrases in the sentence. The algorithm assumes that a conjunction conjoins two conjuncts: a *post-*conjunct which is located directly after the conjunction, and a *pre-*conjunct which is determined by searching backwards through the sentence, matching the semantic labels of phrases with the semantic label of the post-conjunct. This algorithm is simple and achieves relatively good results. However there are several
disadvantages to this approach: (1) the approach was evaluated on a closed domain - the Merck Veterinary Manual; (2) semantic labels must already be assigned to the noun phrases in the sentence; (3) a partial parse of the sentence must first be produced; (4) the post-conjunct does not always directly follow the conjunction; and 5) syntactic information is not utilized.

Resnik (1998, 1995) defines a similarity measure to resolve coordination ambiguities involving nominal compounds. For example, the semantic measure can be used to differentiate between the following two types of noun phrase coordination: *a (bank and warehouse) guard* versus *a (policeman) and (park guard)*. The approach generates a similarity measurement from the extent of the information that two concepts share. The similarity between two concepts is based on the information content of their lowest super-ordinate, referred to as lso(c₁, c₂). The semantic similarity between two concepts is then defined as:

\[
\text{sim}(c₁, c₂) = -\log p(\text{lso}(c₁, c₂))
\]

where \( p(c) \) is the probability of encountering an instance of synset \( c \) in some specific corpus.

Baker (1997) suggests choosing the largest candidate structure that occurs on both sides on the conjunction as the structure that is being coordinated. Consider the example: *John walked the dog and his sister took out the trash*. The following two partial parses are possible: *[John] [walked] [the dog and his sister] [took out] [the trash] and *[John] [walked] [the dog] and *[his sister] [took out] [the trash]*. The largest candidate structure is chosen to identify the meaning of the conjunction. In this case the second
partial parse is chosen. This idea is essentially captured by the larger-first partial parser. An easy-first system would not be appropriate for this task.

6.4 A Hybrid Approach

More recently, a hybrid approach to resolving coordination ambiguity has been proposed (van Delden, 2002). It can almost be described as a combination of the semantic approach by Agarwal and Boggess (1992) and the syntactic approach by Baker (1997) - see previous section. For some cases, syntactic information will suffice in the disambiguation analysis, but other times semantic information is essential to determining the correct meaning of the conjunction. For example, consider the sentence: Peter got a dog with a brown spot and a cat, and Mary got a parrot with green feathers. The second conjunction can be disambiguated using syntax alone due to similarity of the word patterns around the conjunction: Peter bought a dog and Mary bought a parrot. However, semantics is required to disambiguate the first conjunction. Semantically, cat should be more closely related to dog instead of spot. Therefore, the conjunction is coordinating [a dog with a brown spot] and [a cat], and not a dog with [a brown spot] and [a cat].

The algorithm is simple to implement and requires only part-of-speech information to be allotted to each word in the sentence. Part-of-speech tags can be assigned to each word in the sentence by Brill’s part-of-speech tagger. The hybrid approach presented here combines both syntax and semantics to determine the pre-conjunct size. An algorithm first assigns a score to each possible pre-conjunct window size based on the largest sub-sequence of part-of-speech tags that co-occur on both sides of the conjunctions – where a matching bigram of part-of-speech tags can be separated by a single tag. The pre-
conjunct window size scores are then updated by calculating the semantic similarity between the nouns which precede the conjunction and the first noun that follows the conjunction. Resnik's similarity measure (Resnik, 1998) which has been applied to the WordNet (Miller, 1993) ontology is used in the semantic analysis. The window with the highest score is declared the pre-conjunct window size.

6.4.1 Syntactic Analysis

Conjunctions can often be disambiguated based solely on syntactic information – the meanings of the words are not important. The objective is to recognize patterns of similar words on each side of the conjunction. Instead of considering the actual words in the sentence, part-of-speech tags are assigned to each word. Informally, the algorithm attempts to maximize the size of a sequence of similar part-of-speech tags on each side of the conjunction. For example, *Mary bought a car and Peter bought a motorcycle*. A sequence of four similar tags can be found on each side of the conjunction: *noun past-tense-verb determiner noun*. Based on this similarity we can conclude that the pre-conjunct's size is four, starting at the beginning of the sentence.

This approach, however, needs to be generalized so that an accurate score can be assigned when there is not an exact match. This is done by allowing single tags to occur in between a matching bigram sequence. Consider a similar sentence to the one above: *Mary bought a car last week, and Peter almost bought a red truck*. An exact subsequence tag match on each side yields a score of only one. However, allowing a single tag to occur between matching bigram tags yields a score of four: *Mary bought a red car last week, and Peter almost bought a truck*. Since the start of the matching pre-conjunct
is at the start of the sentence, the pre-conjunct window size is seven (or eight including the comma): *Mary bought a car last week,* and Peter almost bought a red truck.

This approach attempts to solve the problem that occurs when the post-conjunct does not directly follow the conjunction. For example: *Mary bought a car last Sunday, and on Monday morning, Peter bought a truck.* Note that a sequence of four tags is also matched here, and the pre-conjunct starts at the beginning of the sentence: *Mary bought a car last Sunday,* and on Monday morning, *Peter bought a truck.* Based on syntactic information we can conclude that the prepositional phrase that directly follows the conjunction is not the syntactic relation that is being coordinated. It is part of the larger independent clause: *on Monday morning, Peter bought a truck.* To improve the performance of this algorithm two special cases are added:

1. if an adjective/number/preposition directly follows the conjunction and an adjective/number/preposition directly precedes it then the pre-conjunct window size is one.

2. if a verb directly follows the conjunction then search backwards through the sentence for the first verb with a matching part-of-speech tag. This verb marks the start of the preconjunct. If no verb is found, proceed with the regular algorithm.

These special cases identify specific patterns around the conjunction and determine the pre-conjunct window size regardless of any other syntactic or semantic information. For example the following two sentences, which were taken from the Worldbook encyclopedia, illustrate rule one and rule two respectively: *Many Latin words helped shape [scientific] and legal terms. and Latin [is in the Indo-European family of languages] and is closely related to Celtic, Germanic, Greek, Sanskrit, and Slavic languages.* There are two reasons justifying these special cases: (1) Whenever there is
conjunction followed by a verb, there is high probability that the conjunction conjoins two verb clauses that have introductory verbs with similar part-of-speech tags - a similar case can be made for rule one; and (2) The semantic measure used in the next step does not provide a similarity measure for parts-of-speech other that nouns.

A score is assigned to each possible pre-conjunct window size. No final decision is made until a semantic analysis is performed (except if there is a special case)

6.4.2 Semantic Analysis

The algorithm used here to analyze the semantic similarity between the post-conjunct and pre-conjunct is very similar to Agarwal and Boggess (1992). The main differences being that only part-of-speech information is required and semantic labels are generated as the algorithm executes.

The semantic labels are generated using Resnik’s similarity measure (Resnik, 1998 and 1995) applied to the WordNet (Miller, 1993) ontology. There have been several other proposed semantic similarity or distance measures which utilize an ontology (Hirst and St-Onge, 1998; Leacock and Chodorow, 1998; Lin, 1998; Jiang and Conrath, 1997). Budanitsky and Hirst (2001) compared all five of these measures by examining their performance on a spelling correction system which detected malapropisms.

Resnik’s similarity measure was chosen because: (1) it yielded relatively good results in the comparison by Budanitsky and Hirst (2001); and (2) it yielded good results when applied to resolving coordination ambiguities in nominal compounds (Resnik, 1998). The SemCor corpus, which is a semantically tagged subset of the Brown Corpus (Francis and Kucera, 1979), is used.
The semantic analysis algorithm is as follows: (1) The first noun that occurs after the conjunction is chosen to be the post-conjunct; (2) Search backwards through the sentence calculating the semantic similarity between each noun and the post-conjunct noun; (3) the semantic similarity for a noun and the post-conjunct noun determines the semantic score for the pre-conjunct window starting at that noun. Each semantic score is recorded so that it can be combined with the corresponding syntactic score for the same window size to determine the final scores for each possible pre-conjunct window size.

Two special cases have also been added to the semantic analysis:

[1] if the noun is a pronoun, then use the word "person" to calculate semantic similarity.

[2] if the noun is a proper noun, then a) attempt to calculate semantic similarity using the proper noun, b) if the proper noun is not in the ontology then calculate semantic similarity with the word "person".

These special cases attempt to produce a similarity measurement even though the particular noun is not in the ontology. For example, consider the sentence, *I never went to school, but my brother attended regularly*. A syntactic analysis based on part-of-speech tags does not provide much information. The semantic analysis would fail in this particular case also, since pronouns are not represented in the WordNet ontology. But when *person* is used in place of *I*, a high semantic similarity is found between *person* and *brother*, and so based on semantic information the correct window size is determined.

A similar argument is made for rule two, except that certain proper nouns are represented in the WordNet ontology. For example consider the sentence: *However, many universities in Great Britain and other European countries still require Latin*. *Great Britain* is found in the WordNet ontology and a high semantic similarity is
generated between *Great Britain* and *countries*. However, many proper nouns, in particular personal names, are not in the ontology. If the proper noun is not found in the ontology, it is replaced with “person” and the semantic similarity is calculated.

Of course pronouns and proper nouns are not always *persons*. This is merely an attempt to generate some semantic information when none can be found due to the absence of a noun in the ontology. For example, *The ship will set sail in the morning, and she will remain at sea for forty days*. Here the pronoun *she* is referring to a *ship* and not a person. So the semantic similarity is calculated between *ship* and *person*, and a score is generated. This false semantic similarity, however, can only help the algorithm find the correct window size, because if the correct window size is not at this location in the sentence then the noun from correct window size should yield a much higher score with the post-conjunct noun.

### 6.4.3 Results

The algorithm was tested on twenty five articles taken from the following sources: the Britannica, Encarta and Worldbook Encyclopedias, the New York Times, and the Wall Street Journal. The results are shown in Table 6.1.

A relaxed criteria was first used to compile the results in Table 6.1. Since the algorithm only uses part-of-speech information, it has no information regarding the boundaries of the syntactic relations in the sentence. In Table 6.1, it is being assumed that if the algorithm returns a window such that the initial word is part of the syntactic relation that starts the pre-conjunct, then it has correctly determined the pre-conjunct. For example the pre-conjunct identified in the following sentence does not include the
determiner: *The [man went to the shop] and the woman stayed at home*. This occurs because *man* and *woman* are very closely related semantically, but the word *the* is not in the ontology. Such cases were taken to be a correct classification. Also, the pre-conjunct size in a list of elements was assumed to be correct if it fell within the boundary of the list.

**Table 6.1. Relaxed Analysis of Performance Results**

<table>
<thead>
<tr>
<th>Sources</th>
<th>number of articles</th>
<th>Tokens</th>
<th>avg. sen length</th>
<th>Number of Conjunctions</th>
<th>Correctly Determined</th>
</tr>
</thead>
<tbody>
<tr>
<td>WorldBook</td>
<td>5</td>
<td>4337</td>
<td>19</td>
<td>205</td>
<td>93.3%</td>
</tr>
<tr>
<td>New York Times</td>
<td>5</td>
<td>5621</td>
<td>26</td>
<td>142</td>
<td>88.2%</td>
</tr>
<tr>
<td>Vall Street Journal</td>
<td>5</td>
<td>7845</td>
<td>26</td>
<td>167</td>
<td>87.1%</td>
</tr>
<tr>
<td>Britannica</td>
<td>5</td>
<td>5036</td>
<td>29</td>
<td>194</td>
<td>92.7%</td>
</tr>
<tr>
<td>Encarta</td>
<td>5</td>
<td>7490</td>
<td>25</td>
<td>181</td>
<td>92.9%</td>
</tr>
</tbody>
</table>

Figure 6.7 shows a more strict analysis of the results. The exact pre-conjunct window size was determined 85% of the time - column 1 in Figure 6.7. This resulted mainly from determiners not being included. The pre-conjunct window size occurred within one token of the correct window size 90% of the time – column 2 in Figure 6.7.

Incorrect usages of conjunctions and ungrammatical sentences (Greenbaum et al., 1985; Warriner and Griffith, 1969) were disregarded from the tests. For example, *The party was fabulous, dancing and carousing until dawn*. The modifier, *dancing and carousing until dawn*, is misplaced since it does not modify anything in the sentence. Even though the algorithm would get this case correct, such examples of ungrammatical sentences were disregarded. The sentence could be correctly rewritten as: *We had a fabulous time at the party, dancing and carousing until dawn.*
Figure 6.7. Strict Analysis of Performance Results
CHAPTER 7

CLAUSE AND PHRASE NETWORKS

The final networks of the cascade are presented in this chapter. First, the automata of the clause network are presented. These automata are similar to their comma-delimited counterparts in the comma network. However, since comma information is not available, boundary identification and containment of attachment ambiguity differ considerably. Finally, the last network in the cascade is presented which identifies the smallest syntactic relations. Refer to the Abbreviation List on page xiv and Section 4.4 for descriptions of the labels that are placed on the arcs of the automata. Also, refer to Appendix C for the complete automata cascade.

7.1 Clause Network

The first automaton in the clause network identifies subordinate clauses (SUB) that are not enclosed by commas and is shown in Figure 7.1. Unlike the previous comma-delimited subordinate clause automaton, extra restrictions are placed on arcs $BC$ and $GE$ to ensure another verb clause is present in the sentence. Without these extra conditions, the automaton would accept if a preposition which could also be a subordinate conjunction was placed before the main verb in the sentence, for example consider the following simple sentence found during testing: Cultural peaks after 1200 AD had
reached the Eastern Woodlands. The tokens after 1200 AD had reached the Eastern Woodlands would have been grouped as a subordinate clause, had the restrictions not been put in place.

Figure 7.1 Subordinate Clause Automaton

Arcs CC and FF share the same label, as do DD and GG. As before, a list of subordinate conjunctions are referenced instead of relying only on the ambiguous IN part-of-speech tag. A phrasal subordinate conjunction (PSUB) can also introduce the subordinate clause or by looking for a verb followed by a clause, as in: Susan said [SUB Mary was coming home]. Note the initial verb phrase (said in the example above) is not included in the grouping. The subject of the clause can be a noun phrase or list of noun phrases, which can be followed by modifiers including coordinated noun or prepositional phrases. After the verb has been consumed, only a noun, list of noun phrases or predicate can be included in the SUB. If none of these are present, the automaton will go from state D or G to E on whatever token is present. However, this token will not be included in the SUB.
Since post-verbal noun phrases are grouped within the subordinate clause, the automaton may make an error when a subordinate clause introduces a sentence but is not concluded with a comma, as in: *Since Mary jogs a mile seems a short distance.* In this sentence, *a mile* is actually the subject of the main clause, but it will be grouped with the subordinate clause since it appears directly to the right of the verb. This error could be avoided by adding extra arcs to the automaton to ensure that a verb phrase does not directly following the noun phrase. However, this was not incorporated here since this error was not encountered during testing.

Verb sub-categorization information would not have been useful in the previous example since *jogs* can take a distance noun phrase as a direct object. However, it may be useful when an ambiguous subordinate conjunction is present. Consider the following sentences: *I saw the customer after you went looking for him* and *I thought the customer before you was a real jerk.* In the first sentence, the verb *saw* takes the noun phrase complement *the customer* and is then followed by a subordinate clause. This automaton would correctly recognize *after you went looking for him* as being a subordinate clause. The second sentence is syntactically equivalent and the same grouping would also be made - *before you was a real jerk* would be incorrectly identified as the subordinate clause. However, this would mean that the verb *thought* was taking a noun phrase complement. If verb sub-categorization information could have been used, this incorrect classification could have been avoided since the verb *to think* does not take a single noun phrase complement.

Another error may occur when multiple *IN* tags (preposition or subordinate conjunction) appear consecutively separated by noun phrases. For example, *I waited after*
work until nighttime before the client finally called. The difficulty lies in determining whether the subordinate clause starts at after, until or before – which could all be prepositions or subordinate conjunctions. In this case it begins at the final IN (before) in the sentence, but this is not always the case. A semantic analysis is needed to determine which IN actually starts the subordinate clause. The automaton in Figure 7.1 will always choose the first IN as introducing the subordinate clause.

Infinitival clauses are identified next by the automaton in Figure 7.2:

![Infinitival Clause Automaton](image)

Figure 7.2 Infinitival Clause Automaton

As with all the non-comma-delimited clauses, only a post verbal noun phrase, list of noun phrases or predicate is grouped with the clause.

The relative clause automata are presented next in Figure 7.3. For simplicity, they have been split into three separate automaton. The first two are similar to their comma-delimited counterparts, recognizing reduced relative clauses and relative clauses introduced by a relative determiner or pronoun.

Notice that the LASTTAG: NP restriction is placed on arc AB of REL1 instead of consuming the NP with an arc that does not assign a structural tag. The reason here is because REL1 is frequently called by other automata. For example, recall the list of noun phrases automaton (CO-LST-NP) in Figure 5.6. CO-LST-NP calls the relative clause
automata (REL) on arc *DD* after the noun phrase that it modifies has been consumed by arc *CD*. REL 1 would not accept if it had to consume the noun phrase since it is called after that position in the sentence by CO-LST-NP.

Besides regular relative clauses, REL 2 also identifies free relative clauses - a relative clause that does not modify anything in the sentence, for example: *[REL2 What Peter did] was terrible*. The final relative clause automaton (REL 3) identifies a reduced relative clause which has a new subject, for example: *The book [REL3 Peter bought] was lost.*

Figure 7.3 Relative Clause Automata

Notice that extra restrictions have been placed on this automaton (arcs *AB* and *HE*) which ensure that another verb phrase is present in the sentence. These restrictions prevent an
incorrect classification in sentences of the following form: *In the beginning [REL3 I went]*
\[ to school every day.\] Upon first thought, one may assume that a non-comma-delimited subordinate clause that introduces a sentence will also cause the same problem. But since the subordinate clause would have already been identified, this error is already avoided, for example: 
*SUB After [NP Peter] [VP bought] [NP the book] [NP I] [VP found] [NP it] [PP on sale].*

However, there is an error associated with distinguishing between a particular type of SUB and a REL3. Consider the following sentence: *Mary told Peter I was coming to dinner.* Notice the SUB automaton (in Figure 7.2) is not designed to recognize this subordinate clause because the pattern cannot be correctly determined without verb subcategorization information. REL3 will, however, (incorrectly) classify *I was coming* as a relative clause because a similar syntactic structure could very well be a REL3: *Mary found the book I was looking for in the library.* Again, verb subcategorization could be used here to realize that the verb *told* (from the first sentence) takes a noun phrase complement followed by a clause complement.

Finally, inverted sentences could also cause a problem for the REL3 automaton. For example, consider the following inverted sentence: *Your student Peter shall be.* REL3 will not generate an error on this particular sentence since it only has one verb phrase. Since such sentences occur so infrequently, there is no need to discuss it further here.

The automaton which identifies gerunds, participle clauses (complement -ing clauses) or reduced subordinate clauses (ING) is presented next in Figure 7.4.
Figure 7.4 Gerund, Participle or Reduced Subordinate Clause Automaton

ING can be introduced by the start if the sentence, a subordinate conjunction or a verb that is not an auxiliary verb. For example: [ING Hiking four miles] will be a challenge; Mary keeps [ING hoping] for rain; and [ING While walking] to school the boy lost his homework. Specific verb tags are referenced by arc BC and so particles must also be identified after that - arc CC. Also, if arc AB is taken on a verb, then this verb should not be an auxiliary verb, since it would most likely be included with the present or past participle that follows it.

7.2 Phrase Network

The final automata in the cascade are presented here in the phrase network. The first is the prepositional phrase automaton (PP) in Figure 7.5. A PP can be introduced by a preposition (IN) or phrasal preposition (PIN1 and PIN2 identified earlier in the cascade). The object of the PP can be a noun phrase, time noun phrase or list of noun phrases.

Figure 7.5 Prepositional Phrase Automaton
The time noun phrase automaton (TNP) in Figure 7.6 is next in the cascade. The automaton must precede the noun phrase automaton since it identifies a more specific type of noun phrase. Many of the arcs have been lexicalized to recognize such phrases as: today, yesterday, tomorrow, last month, next January, May 5th, 1977.

If the words today, yesterday or tomorrow are directly followed by a possessive ending (POS) then a TNP is not recognized. These words will be grouped with the words that follow them by the noun phrase automaton (NP) that is located next in the cascade. Here are some examples: [TNP Today] Peter will ask Mary to marry him; The mailman didn’t make any deliveries [TNP last week]; and [NP Tomorrow’s baseball game] has been cancelled.

A TNP could be mistaken for a NP when one of the lexical tokens is being used in a proper noun phrase, for example: USA Today sold over 14 million copies last year. Today in this sentence will be incorrectly identified as a TNP. A TNP could also be functioning as the subject of the sentence, in which case it should be identified as regular NP, for
example: *Today is a beautiful day for fishing.* To avoid these two errors, the TNP automaton could be extended to look for capitalization within the sentence and to see if a verb directly follows the TNP. If the TNP is capitalized and not at the start of the sentence, then it is not a TNP - solves first error. If a verb directly follows the TNP then it mostly likely is the subject of that verb - solves second error. This extension will solve many errors, but not all.

The noun phrase automaton (NP) is presented next in Figure 7.7.

![Figure 7.7 Noun Phrase Automaton](image)

Arc AE identifies single token NPs like *I/PRP, he/PRP, she/PRP,* and existential *there.* The arc path AD (DD)* DE identifies noun phrases with only nouns (possibly multiple) and no pre-noun modifiers or determiners, for example: *Peter/NNP, school/NN bus/NN, computer/NN book/NN, Dutch/NNP Publishing/NNP Company/NNP,* etc... Similarly, arc path AB BD (DD)* DE identifies noun phrases with only noun (possibly multiple) that are preceded by a (pre)determiner or possessive pronoun: *the/DT Dutch/NNP*
Publishing/NNP Company/NNP and my/PRP$ computer/NN book/NN. A slightly different path, \( AB \text{ BC (CC)}^\ast \text{ CD (DD)}^\ast \text{ DE,} \) identifies the same type of NPs except they now contain adjectives (possibly in coordination or being modified by adverbs), for example: the/DT very/RB unsuccessful/JJ Dutch/NNP Publishing/NNP Company/NNP and the/DT yellow/JJ and/CC red/JJ school/NN bus/NN. The CC-CNP tag on arc DD groups the latter part of a complex noun phrase together with the preceding part: [NP the/DT very/JJ Dutch/NNP Publishing/NNP Company/NNP and the/DT yellow/JJ and/CC red/JJ school/NN bus/NN]. Finally, the back arc DC is taken on a possessive ending (POS) and grouping the possession with the possessor: my/PRP$ uncle/NN Frank/NNP 's/POS motorized/JJ cement/NN mixer/NN.

There are several situations where this NP automaton will produce errors. First consider the following sentence: By 1950 many people had left the area. The problem occurs when a prepositional phrase introducing a sentence and containing a year was directly followed a noun phrase that was not a pronoun and did not contain a determiner. Grouping the pattern CD JJ NNS is not a bad choice, since such a pattern could very well be a valid noun phrase: 12/CD red/JJ apples/NNS. This very specific error could be fixed any adding a lexical feature to the automaton that looks for such a pattern containing a year part of a date.

A possible error that did not occur during testing could be made when two noun phrase objects are located next to each other. For example: Peter gave [NP Mary books]. Mary books will be incorrectly grouped as a single noun phrase. This is not a very bad decision since such a pattern (NNP NNS) could very well be a single noun phrase, for example: Peter gave [NP Calculus books] to Mary. As with previous examples in Section 7.1, this error could possibly be corrected by including verb sub-categorization
information in the automaton. A similar situation can be found in the following sentence: 
*I told Mary Peter was coming.* This situation is different than the subordinate clause problems discussed in Section 7.1. *Mary Peter was coming* will be identified as a subordinate clause because the NP automaton is unable to recognize there are actually two noun phrases and not one. Such a sequence is possible however: *I said Peter Henderson was coming.* Again, verb sub-categorization can be used here to realize *told* does not take a clause complement alone.

Another possible error that appeared once during testing was when a predicate was directly followed by a comma and a noun phrase, as in: *After the man turned green, many medics came to his aid.* The sequence *green, many medics* is mistaken as a noun phrase since *JJ, JJ NNS* is a likely noun phrase pattern.

Verb phrases (VP) are identified next by the automaton in Figure 7.8. Arcs *AB* and *BB* consume auxiliary and modal verbs which may be followed by a main verb - arc *BC*. Once in arc *C*, a particle can be consumed to reach state *D* or state *D* can be reached by any other token which is not assigned a structural tag. If the auxiliary or modal verb is the main verb, then arc *BD* can be taken so that automaton accepts. Transitional phrases (CO-TRN) and comma-delimited prepositional phrases (CO-PP) may be enclosed within the verb phrase. Finally, if no auxiliary or modal verbs are present, the path *AC CD* can be taken to identify a verb phrase. Here are some example verb phrases recognized by this automaton: *Peter [VP can [CO-TRN, of course, ] still take ] the exam; I [ VP woke up ] early this morning; I [ VP will ]; and Mary [ VP is ] short.*
Figure 7.8 Verb Phrase Automaton

This automaton will not correctly group particles with the verb phrase if they do not directly follow the verb phrase. For example, *Peter woke Mary up this morning*. The automaton cannot be extended to make this grouping since the verb phrase is no longer a contiguous block of tokens. If the particle is correctly identified by the part-of-speech tagger (as a RP), then the particle could be associated with its preceding verb phrase by a post cascade program.

Finally, the last automata of the cascade are presented in Figure 7.9. Adjective phrases (ADJ) and adverbial phrases (ADV) are grouped together. Coordinate conjunctions (CC) may be contained within both of these phrases. For example: *The journey was [ADJ long and tiresome]*, and *Peter adjusted the motor [ADV accurately and efficiently]*.

Figure 7.9 Adjective and Adverbial Phrase Automata
CHAPTER 8

EVALUATION

An evaluation of the larger-first approach is presented in this chapter. The evaluation is two fold. First, the larger-first approach is applied to one hundred randomly taken sentences from the Encyclopedia Encarta. The output of the system on these sentences is shown in Appendix D. Other examples of correct and incorrect sentences that were encountered during the development of the system (as well as ambiguous cases) can be found throughout Chapters 5, 6 and 7. Part-of-speech tagging errors were manually corrected during the evaluation so that the system’s performance could be measured on gold-standard (correct) part-of-speech tags.

Second, the larger-first approach is applied to the Wall Street Journal Section of the Penn Treebank III. Its output is compared against that of the CASS system – the implementation of the easy-first approach. The last section in this chapter describes part-of-speech tagging errors that were commonly made during the evaluation and how these errors affect the finite state cascade.
8.1 Encarta

One hundred random sentences were taken from the Encyclopedia Encarta (2001). This encyclopedia was chosen as a test bed because it is a well-written text containing fairly complex sentences. The sentences were first tagged with Brill’s tagger (Brill, 1994). However, if a sentence contained incorrect part-of-speech tags, they were corrected during the evaluation and the sentence was re-evaluated, to see how well the system performs on gold-standard (correct) part-of-speech tags. The entire set of one hundred sentences that have been partially parsed by the larger-first system is located in Appendix D. The number of test sentences was limited to one hundred so that they could all be included in this work (Appendix D). On average, each sentence contained 22 words. Results were evaluated using three performance metrics – Precision, Recall, and their harmonic mean (or F-score):

\[
\text{Precision} = \frac{\text{Number of correct proposed patterns}}{\text{Number of proposed patterns}}
\]

\[
\text{Recall} = \frac{\text{Number of correct proposed patterns}}{\text{Number of correct patterns}}
\]

\[
F_\beta = \frac{(\beta^2 + 1) \times \text{Recall} \times \text{Precision}}{\beta^2 \times \text{Precision} + \text{Recall}}
\]

\(\beta = 1\) was used. The results are shown in Table 8.1. The syntactic relations are listed in descending order of occurrence. Of the one hundred sentences, only eleven contained errors.
Table 8.1 Results of the Evaluation of the Encyclopedia Encarta.

<table>
<thead>
<tr>
<th>Syntactic Relation</th>
<th>Occurrence</th>
<th>Precision</th>
<th>Recall</th>
<th>F-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td>358</td>
<td>98%</td>
<td>98%</td>
<td>98%</td>
</tr>
<tr>
<td>PP</td>
<td>264</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
</tr>
<tr>
<td>VP</td>
<td>159</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CC-NP</td>
<td>38</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>ADV</td>
<td>25</td>
<td>92%</td>
<td>88%</td>
<td>90%</td>
</tr>
<tr>
<td>REL</td>
<td>22</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CO-REL</td>
<td>17</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CO-APS</td>
<td>15</td>
<td>92%</td>
<td>80%</td>
<td>86%</td>
</tr>
<tr>
<td>INF</td>
<td>15</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CO-LST-(VC INF NP)</td>
<td>12</td>
<td>85%</td>
<td>92%</td>
<td>88%</td>
</tr>
<tr>
<td>ADJ</td>
<td>12</td>
<td>83%</td>
<td>83%</td>
<td>83%</td>
</tr>
<tr>
<td>CC-VC</td>
<td>11</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CC-VS</td>
<td>10</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CO-PP</td>
<td>8</td>
<td>89%</td>
<td>100%</td>
<td>94%</td>
</tr>
<tr>
<td>SUB</td>
<td>7</td>
<td>100%</td>
<td>86%</td>
<td>93%</td>
</tr>
<tr>
<td>PIN</td>
<td>6</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>ING</td>
<td>5</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CO-SUB</td>
<td>4</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CO-S</td>
<td>4</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CO-COR</td>
<td>3</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CO-VC</td>
<td>3</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CO-RSUB</td>
<td>3</td>
<td>100%</td>
<td>67%</td>
<td>80%</td>
</tr>
<tr>
<td>TNP</td>
<td>2</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>PSUB</td>
<td>1</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CO-TRN</td>
<td>1</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CO-IDIR</td>
<td>1</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>CC-PP</td>
<td>1</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>CO-TNP</td>
<td>0</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>CO-INF</td>
<td>0</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>CO-DIR</td>
<td>0</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>CC-INF</td>
<td>0</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>CC-REL</td>
<td>0</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>CC-SUB</td>
<td>0</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>CC-ING</td>
<td>0</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>CC-CNP</td>
<td>0</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Eighty nine sentences were completely error free. Some syntactic relations had zero occurrences since only one hundred sentences were evaluated. The purpose of this experiment is to provide real world sentences that have been parsed by the system (see Appendix D) and is not meant to be a thorough evaluation of the Encyclopedia Encarta.

There was one occurrence of indirect speech. The system was unable to correctly identify it (corresponding to the zero scores in Table 8.1) because the speech contained comma-delimited syntactic relations which are not being recognized in speech since written text is the focus of this work. But the automata could easily be supplemented to accommodate more syntactic relations that occur in speech.

Some errors occurred when distinguishing between list of noun phrases and appositions. Whenever an apposition contains a coordinate conjunction, there is the possibility of confusing it with a list. For example, consider the following sentence from Appendix D:

```
( NP The/DT inhabitants/NNS ) ( PP of/IN ( NP Labrador/NNP ) ) ( VP included/VBD ) ( NP a/DT small/JJ number/NN ) ( PP of/IN ( NP Inuit/NNP ) ) ( PP along/IN ( NP the/DT northeastern/JJ coast/NN ) ) ,/, and/CC ( LST-NP2 ( NP two/CD closely/RB related/JJ Algonquian/NNP groups/NNS ) ,/, ( NP the/DT Naskapi/NNP ) and/CC ( NP the/DT Montagnais/NNP ) ) ,/, ( CO-REL2 who/WP ( VP were/VBD dispersed/VBN ) ( PP throughout/IN ( NP the/DT rest/NN ) ) ( PP of/IN ( NP the/DT land/NN ) ) ) ./
```

*Two closely related Algonquian groups, the Naskapi and the Montagnais* is incorrectly identified as a list of noun phrases when actually *the Naskapi and the Montagnais* appositives *two closely related Algonquian groups*. This error was discussed earlier in Section 5.3 and a solution was presented to resolve the problem. Had the solution been implemented, the apposition in the above sentence would have correctly been identified.
As with the comma tagger in Sections 5.6 and 5.7, elliptical constructions cause problems for the partial parser. Consider the following sentence taken from Appendix D:

```
( NP SIDS/NNP ) ( VP is/VBZ more/RBR ) ( NP common/JJ ) ( PP in/IN ( NP infants/NNS ) ) ( REL2 whose/WFS ( NP mothers/NNS ) ( VP are/VBP ) ) ( PP under/IN ( NP 20/CD years/NNS ) ) ( NP old/JJ ,/ , unmarried/JJ ) ,/, ( VP have/VBP had/VBN ) ( NP inadequate/JJ prenatal/JJ care/NN ) ,/, ( VP did/VBD not/RB breast-feed/VBD ) ( NP the/DT infant/NN ) ,/, ( CO-VC or/CC ( VP have/VBP ) ( NP more/JJR ) ( PP than/IN ( NP one/CD infant/NN ) ) ) ./.
```

This sentence contains a list of verb clauses which is part of the relative clause starting at whose mothers. The list of verb clauses is not identified for two reasons. First, an elliptical construction before unmarried omits the verb are. Lack of a verb in this position will make it impossible to recognize that this is a list of verb clauses. A much more detailed analysis of the entire sentence is needed and is beyond the scope of this approach. The second reason for failing on this sentence is over specialization. Recall, a restriction was placed on the list of verb clauses automaton: the verb phrase in each verb clause must have the same tense. Noted that is not the case in the example. Lifting this restriction, however, may cause the automaton to be over generalized, resulting in many errors.

In total, seven noun phrases were misidentified, all similar to the error in the following sentence. The sentence should start with the prepositional phrase By the mid-1980s and by followed by the noun phrase many authors.

```
( CO-PP ( PP By/IN ( NP the/DT mid-1880s/CD many/JJ authors/NNS ) ) ) ,/, ( CO-REL1 including/VBG ( NP B./NNP L./NNP Farjeon/NNP ) ( CC-NP and/CC ( NP Thomas/NNP W./NNP Speight/NNP ) ) ) ,/, ( VP were/VBD writing/VBG)(NP genuine/JJ detective/NN novels/NNS ) ./.
```

In this case, the pattern DT CD JJ NNS is two noun phrases, but in other situations, it could very easily be a single noun phrase, for example: the/DT 1880/CD good/JJ
sentences/NNS. Overall, however, noun phrases and prepositional phrases were identified with a high F-score: 98% and 99%, respectively.

Some syntactic relations were present in the evaluation sentences, for which there were no automata defined. There was one occurrence of a list of adjectives in a noun phrase, one occurrence of a list of prepositional phrases, and one occurrence of a list of independent clauses. These errors are only reflected in the total number of sentences that contained errors (11), and can be overcome by added automata to recognize them. As mentioned in Chapter 3, the syntactic relation set and automata that have been defined in this work represent the most frequently occurring patterns over an open domain and are not meant to be an exhaustive list.

Although this larger-first approach is incapable of resolving some syntactic issues, the results show that for the overwhelming majority of sentences (89/100 of the sentences in Appendix D) it performs very well. Consider the following two sentences taken from Appendix D:

( NP Other/JJ successful/JJ writers/NNS ) ( PP in/IN ( NP this/DT school/NN ) ) ,/ , ( CO-REL1 including/VBG ( LST-NP1 ( NP Catherine/NNP Aird/NNP ) ,/ , ( NP Reginald/NNP Hill/NNP ) ,/ , ( NP Patricia/NNP Moyes/NNP ) ,/ , and/CC ( NP June/NNP Thomson/NNP ) ) ) ,/ , ( VP have/VBP ) ( PP at/IN ( NP the/DT center/NN ) ) ( PP of/IN ( NP their/PRP$ works/NNS ) ) ( NP an/DT imperfect/JJ ) ( PP though/IN ( NP sensitive/JJ sleuth/NN ) ) ( REL2 whose/WF$ ( NP life/NN ) ( CC-NP and/CC ( NP attitudes/NNS ) ) ( VP are/VBP ) ) ( PP of/IN ( ADV almost/RB ) ( NP equal/JJ importance/NN ) ) ( PP to/TO ( NP the/DT mystery/NN ) ) ./. 

( NP Other/JJ useful/JJ medical/JJ substances/NNS ) ( REL1 now/RB manufactured/VBN ) ( PP with/IN ( NP the/DT aid/NN ) ) ( PP of/IN ( NP recombinant/JJ plasmids/NNS ) ) ( VP include/VBP ) ( LST-NP1 ( NP human/JJ growth/NN hormone/NN ) ,/ , ( NP an/DT immune/JJ system/NN protein/NN ) ( REL1 known/VBN ) ( PP as/IN ( NP interferon/NN ) ) ,/ , ( NP blood-clotting/JJ proteins/NNS ) ,/ , and/CC ( NP proteins/NNS ) ) ( REL2 that/WDT ( VP are/VBP used/VBN ) ) ( ING in/IN making/VBG ( NP vaccines/NNS ) ) ./.
In both sentences the list of noun phrases is contained nicely within the comma-delimited relative clauses. The list in the second sentence has an extra complication since it also contains a non-comma-delimited relative clause (*known*) within it. Both sentences also contain non-comma-delimited relative clauses. Note that these relative clauses end directly before the first post verbal prepositional phrase that they encounter (Guideline 1).

The following sentence also illustrates how helpful comma information is to containing ambiguity:

\[( \text{CO-PP} ( \text{PP In/IN } ( \text{NP large/JJ paintings/NNS} ) ) ( \text{REL1 often/RB encrusted/VBN} ) ( \text{PP with/IN } ( \text{LST-NP1 (NP straw/NN ,/, (NP dirt/NN ,/, or/CC (NP scraps/NNS)}) ) (PP of/IN (NP lead/NN)}) ) ,/, (NP Kiefer/NNP) (VP depicted/VBD) (ING devastated/VBN (NP landscapes/NNS) ) (CC-NP and/CC (NP colossal/JJ ,/, bombed-out/JJ interiors/NNS)}) )/.\]

The sentence is introduced by a comma-delimited prepositional phrase (CO-PP) which contains a relative clause (REL1) and three other prepositional phrases (PP), one of which also contains a list of noun phrases (LST-NP1). Also, at the end of the sentence a noun phrase containing coordinated adjectives is identified as being coordinated with another noun phrase.

The next sentence taken from the evaluation is a good example of how non-comma-delimited clauses can also contain ambiguity:

\[
\text{STAART/STAART (NP It/PRP) (VP seems/VBZ) (SUB that/IN (NP even/JJ actors/NNS)) (REL2 who/VP (VP speak/VBP) (NP AAVE/NNP)) (PP at/IN (NP home/NN) ) (VP recognize/VB) ) (PP on/IN (NP some/DT level/NN) ) (SUB that/IN (NP the/DT grammar/NN) (PP of/IN (NP the/DT vernacular/NN)) (VP would/MD not/RE be/VB understood/VBN) ) (PP by/IN (NP the/DT general/JJ public/NN) ) ./.}
\]

A subordinate clause is first encountered. The subject of this clause, however, has a relative clause attached to it. Since this relative clause occurs before the main verb of the
subordinate clause, it is contained. The prepositional phrase at home is also situated before the verb and so it is also contained within the subordinate clause. Post verbal prepositional phrases are not included, so the first subordinate clause ends directly before on some level. The second subordinate clause also contains the pre-verbal prepositional phrase of the vernacular.

One last sentence taken from the evaluation is shown below. As expected, many of the coordinated clauses from the conjunction network did not appear in the evaluation (CC-REL, CC-INF, CC-SUB, CC-ING), since these constructions usually have a much lower frequency when compared to the other syntactic relations. However, coordinated verb clauses did appear in about 10% of all sentences:

STAART/STAART ( NP The/DT GEIC/NNP ) ( CC-VS ( VP set/VBD up/RP ) ( NP a/DT temporary/JJ headquarters/NN ) ( PP during/IN ( NP the/DT Japanese/JJ occupation/NN ) ) ( PP of/IN ( NP the/DT Gilbert/NNP Islands/NNPS ) ) ( PP in/IN ( NP 1942/CD ) ) ( CC-VC but/CC ( VP moved/VBD ) ( NP the/DT administration/NN ) ( ADV back/RB ) ) ( PP to/TO ( NP Tarawa/NNP ) ) ( SUB after/IN ( NP the/DT United/NNP States/NNPS ) ( VP drove/VBD ) ( NP the/DT Japanese/NN ) ) ( PP from/IN ( NP the/DT Gilberts/NNS ) ) ( PP in/IN ( NP 1943/CD ) ) ).

Note that many prepositional phrases are contained within the first verb phrase and the conjunction. Although this syntactic relation was identified with a 100% precision during the evaluation, there is a possibility of incorrect groupings, for example: Peter just bought a computer [REL that [ VP has ] [NP a fast processor ] [CC-VC and wants ] ] to take it to the school to show my friends . In this case there are multiple pre-conjunct sites, bought and has. If this is the case, a rightmost grouping is made, which turns out to be incorrect in this sentence. However, since no errors were found during testing or the evaluation, the occurrence of such sentences are rare.
8.2 Comparison to CASS

The larger-first approach (referred to in this Section as LAFI - LArger-FIrst) was also evaluated on Section 23 of the Wall Street Journal Penn Treebank III (Marcus et al., 1993). The CASS system (version 1h) was also evaluated on this corpus. The results are shown in Table 8.2. CASS was compared against LAFI by following three evaluation criteria: 1) precision on the sentence level, 2) richness or detail of partial parse, and 3) containment of ambiguity.

Table 8.2 Evaluation and comparison of the CASS and LAFI systems. Section 23 of the Wall Street Journal Penn Treebank III was used as the test corpus.

<table>
<thead>
<tr>
<th>System</th>
<th>Precision</th>
<th>Richer Partial Parse</th>
<th>Better Containment of Ambiguity</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASS</td>
<td>87.5%</td>
<td>-</td>
<td>5.1%</td>
</tr>
<tr>
<td>LAFI</td>
<td>88.6%</td>
<td>31.3%</td>
<td>36.3%</td>
</tr>
</tbody>
</table>

8.2.1 Sentence Level Precision

CASS and LAFI identify many similar syntactic relations (see Chapter 3), motivating a comparison between the two approaches. However, LAFI identifies appositions and partially disambiguates conjunctions which CASS is incapable of identifying. Furthermore, the exact boundaries of the syntactic relations and structure of partial parse differs across systems. Sentence level precision is used here to generate a rough comparison of both systems based on their own criteria and syntactic relation set. Since the CASS system identifies a different set of syntactic relations than LAFI, it would be illogical to compare them on the syntactic relation level. Instead, an error was assigned if either system made a mistake on the sentence level. Precision is defined as the number of
correct sentences divided by the total number of sentences. As shown in Table 8.2, the evaluation yielded a similar precision for both systems: 87.5% for CASS and a slightly better 88.6% for LAFI. This precision is intended here only as an indicator that LAFI is capable of producing an accuracy comparable to the CASS system.

The main contribution of the larger-first approach is that it is capable of producing a richer partial parse and a better containment of ambiguity while maintaining an accuracy comparable to that of the easy-first approach.

An error was assigned if a sentence contained an erroneous grouping for which the system had a correct grouping. For example, consider the following sentence that was encountered during the evaluation (many groupings have been omitted for simplified viewing):

**CASS:** The group is led by [NP Jay Shidler], [NG chief executive officer of Shidler Investment Corp.] [PP in Honolulu], and [NP A. Boyd Simpson], [NG chief executive of the Atlanta-based Simpson Organization Inc.]

**LAFI:** The group is led by [NP Jay Shidler], [CO-APS chief executive officer of Shidler Investment Corp. in Honolulu], [CC-NP and A. Boyd Simpson], [CO-APS chief executive of the Atlanta-based Simpson Organization Inc.]

This output was classified as incorrect for the CASS system, but were correct for the LAFI system. An error is made by the CASS system because it does not group the two nouns that are being coordinated, but CASS *does* have a tag for grouping two noun phrases in coordination - *NG*. Our system makes a better classification, because the commas are first analyzed and the appositions recognized. The coordination network is then able to identify the conjunction as coordinating two noun phrases. As stated in Chapter 3, no attempt is made to determine the pre-conjunct of phrases.
8.2.2 Partial Parse Richness

The *richness* of a partial parse refers to the level of detail a partial parsing system produces. A system that disambiguates more syntactic relations than another system delivers a richer partial parse. Sentences that were correctly partially parsed by both systems' standards were analyzed to see which system created a more detailed partial parse. For example, if the following sentence had been correctly parsed by both systems, LAFI would produce a richer partial parse, because it identifies the apposition and the lists of noun phrases, where as CASS would identify two syntactic relations of the same type:

CASS: [NG Frank Dickson, the CEO of the company,] is in charge of all aspects of [NG sales, marketing, and productions].

LAFI: Frank Dickson, [APS the CEO of the company], is in charge of all aspects of [LST-NP sales, marketing, and productions].

As shown in Table 8.2, LAFI produces a richer partial parse on 31.3% of the test sentences, indicating that LAFI is capable of achieving a more detailed partial parse than CASS while remaining as precise as the CASS system.

8.2.3 Containment of Ambiguity

Sentences that were correctly partially parsed by both CASS and LAFI were also analyzed for the extent of containment of ambiguity. Since LAFI uses comma information to help limit attachment sites, containment of attachment ambiguity is often much better in sentences containing commas. For example:
LAFI: At those levels, stocks are set to be hammered by index arbitrageurs, [REL who lock in profits by buying futures [SUB when future prices fall]], and simultaneously sell off stocks.

The comma concluding the first prepositional phrase does not help contain ambiguity since only one phrase is being enclosed. More ambiguity is, however, contained by LAFI because of the two other commas in the sentence. These commas are identified as enclosing a relative clause. Since the relative clause contains a subordinate clause, attachment sites for the subordinate clause are limited to within the relative clause. This containment is not performed by the CASS system: At those levels, stocks are set to be hammered by index arbitrageurs, [REL who lock in profits by buying futures] [SUB when future prices fall], and simultaneously sell off stocks. LAFI, therefore, performs a better containment of ambiguity on such sentences.

As shown in Table 8.2, the LAFI system contained ambiguity of attachment sites better than the CASS system for 36.3% of the test sentences – primarily because of the information provided by the comma punctuation mark.

In some situations, CASS performs a right-most attachment of prepositional phrases. In these cases, the CASS system produces better containment of attachment ambiguity than LAFI (if the attachment is correct) since LAFI does not resolve any explicit attachment decisions. If an attachment is incorrect an error is generated (Section 8.2.1). The 5.1% better containment of ambiguity for CASS corresponds to the simple right-most attachment of prepositional phrases that is sometimes performed by the CASS system.
8.3 Part-of-Speech Tagging Errors

Brill’s tagger (Brill, 1994) was downloaded and used as is. It had been trained on the Wall Street Journal Section of the Penn Treebank (Marcus et al., 1993) and delivers fairly good results without having to re-train the tagger - a very time consuming process. Even if the tagger is trained on the corpus that is being used, there is still the possibility of tagging errors being made. Brill’s tagger (like other state-of-the-art taggers) can achieve an accuracy of 95-97% (about 19/20 words correct) when trained on a corpus. However, even this good performance could result in at least one error being made in every sentence since the length of the sentences in the Wall Street Journal Penn Treebank is 27 words. There were 22 words per sentence in the Encarta Evaluation (Section 8.1). Major tagging errors can be devastating to the larger-first cascade. Even some minor errors may have a terrible effect.

A Major tagging error is defined here as when a tag from one part-of-speech category is assigned to a word that belongs to a phrase that cannot contain that category. The most commonly occurring instances were:

- NNS and VBZ - plural noun versus 3rd person singular verb
- JJ and VBN/VBG - adjective versus present or past participle
- NN and VB - base noun versus base verb

There are several ways the tagger can make these errors. First, if the word is unknown, then lexical clues are used by the tagger to assign a part-of-speech tag. For example, consider the following sentence: The container houses many artifacts. In this case, houses
is used as a 3rd person singular verb. The tagger may say that it is actually a plural noun (because of the -s suffix) and then use contextual information to realize that it is actually a verb. If the tagger fails to realize this, an error will be made.

Second, another situation is when the word is known, but it requires a part-of-speech tag that has not been observed during training, i.e. the required target tag is not associated with the word in the lexicon. This results in terrible tagging errors. For example, consider the following sentence that occurred during testing: The pitted bearing needed to be replaced. The word pitted only has VBN and VBD (past tense) tags associated with it in the lexicon. Even though pitted is obviously not a verb in the above sentence, it will be tagged as one since the appropriate target tag (JJ) is not a possible tag according to the lexicon. Brill's tagger should be supplemented with a new contextual transformation that changes a part-of-speech tag whether or not the target tag is in the lexicon. This transformation would minimize such ridiculous errors as the one made above.

Third, another common error is when the target tag is in the lexicon, but is not the most likely tag, and an appropriate contextual rule has not been learned which would choose it for a new context that it is currently appearing in. In this case, the most likely tag is assigned which is, of course, not always correct. Any of these major tagging errors will result in an error in the partial parse.

A Minor tagging error is when a tag from one part-of-speech category is assigned to a word that belongs to a phrase that can contain that category, but the category is incorrect. The most commonly occurring instances were:
VBN and VBD - past tense versus past participle

JJ and NNP - adjective versus proper noun

VBN and VBD both comprise verb phrases and JJ and NNP both comprise noun phrases (JJ can also comprise a predicate). When their tags are confused, the partial parse may or may not contain an error - it depends on the particular situation. For example, in the following sentence, the verb phrase automaton will still correctly recognize the verb clause even though walked should be tagged VBN: I/PRP have/VB ./, of/IN course/NN ./, walked/VBD the/DT dog/NN ./ . Such tagging errors are frequently made when the past participle verb does not directly follow the auxiliary verb. However, in the following sentence, if the first verb (ran) is tagged VBD, then the relative clause automaton will not be able to identify it as introducing a relative clause: The horse ran past the barn fell. Furthermore, if the second verb (fell) is tagged VBN, then the relative clause automaton will incorrectly identify it as introducing a relative clause.

Similarly, the noun phrase automaton will not be able to recognize the first noun phrase in the following sentence, since at least one noun is required to be in a noun phrase: The/DT British/JJ agreed/VBD to/TO sign/VB the/DT treaty/NN ./ . British in this case is incorrectly tagged as a JJ, but JJ could be a possible tag for it: The/DT British/JJ army/NN agreed/VBD to/TO sign/VB the/DT treaty/NN ./ .

The automaton in the larger-first cascade could be modified to handle certain tagging errors. For example, if determiners 'a' or 'the' are followed by a verb, then include the verb in the noun phrase or if determiners 'a' or 'the' are followed by an adjective then this will be a noun phrase regardless of whether a following noun follows. However, this
would result in a confusion of two separate problems - part-of-speech tagging and partial parsing. A partial parser should focus on rules that assume the part-of-speech tags are correct.

Hopefully future advances in part-of-speech tagging will produce a tagger that is very accurate across multiple domains without the need for training. This tagger would definitely enhance the practical value of any system that relies on part-of-speech tags.
CHAPTER 9

CONCLUSIONS

This concise and final chapter offers a summarization of the ideas presented in this dissertation along with conclusions that can be drawn from them. Several conclusions are realized from this work:

- A larger-first approach to partial parsing is a viable alternative to easy-first partial parsing.

- Larger-first partial parsing can achieve a more detailed output by (1) identifying appositions; (2) fully disambiguating clausal conjunctions and partially disambiguating phrasal conjunctions; and (3) correctly classifying both instances of over-lapping syntactic relations.

- Three simple guidelines should be followed to achieve a deep partial parse based on syntactic information: (1) Explicit attachment decisions are always avoided; (2) Only comma information can be used to contain explicit attachment ambiguity; and (3) A syntactic relation may be a complement to, an attachment to, or in coordination with a peer syntactic relation or a relation within a prior sub-clause.
- Although hand-crafted based on an empirical analysis, a larger-first cascade of relatively few automata produces good results across multiple domains.

- Incorporating verb sub-categorization information in the arcs of the automata will help resolve many possible ambiguities between non-comma-delimited syntactic relations.

- Comma information can play an important role when identifying clause and phrase boundaries as well as containing attachment ambiguity.

- Comma tagging is a simpler task since: (1) boundary identification is no longer a factor and (2) a post automata co-occurrence matrix which is learned from a corpus permits un-ordering of the automata.

- Comma tagging can be extended from English to the Dutch language with three levels of modification: (1) no modification, (2) translation of lexicalized arcs, and (3) syntactic reorganization due to new verb syntax capabilities.

- A hybrid approach which uses both syntactic and semantic information to fully disambiguate coordinate conjunctions is capable of good results (over 90% accuracy) and requires only part-of-speech information and a semantic similarity measure.

- When evaluated, the larger-first approach produced a richer partial parse (31.3%) and better containment of attachment ambiguity (36.3%) while maintaining a slightly better sentence level accuracy (88.6%).
The next challenge is to link this partially parsed output to a semantic interpreter (Gomez, 2001) so that semantics can be used to resolve the remaining structural ambiguities as well as determining verb meaning and semantic roles of verbal complements. The ultimate result will be a formal system of independent components which will take a written sentence from an unrestricted domain and fully resolve all syntactic and semantic ambiguities.
APPENDIX A

THE PENN TREEBANK TAGSET
<table>
<thead>
<tr>
<th>CC</th>
<th>Coordinating Conjunction</th>
<th>PRP$</th>
<th>Possessive Pronoun</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD</td>
<td>Cardinal Number</td>
<td>RB</td>
<td>Adverb</td>
</tr>
<tr>
<td>DT</td>
<td>Determiner</td>
<td>RBR</td>
<td>Adverb comparative</td>
</tr>
<tr>
<td>EX</td>
<td>Existential there</td>
<td>RBS</td>
<td>Adverb superlative</td>
</tr>
<tr>
<td>FW</td>
<td>Foreign word</td>
<td>RP</td>
<td>Particle</td>
</tr>
<tr>
<td>IN</td>
<td>Prep or sub conjunction</td>
<td>SYM</td>
<td>Symbol</td>
</tr>
<tr>
<td>JJ</td>
<td>Adjective</td>
<td>TO</td>
<td>to</td>
</tr>
<tr>
<td>JJR</td>
<td>Adjective comparative</td>
<td>UH</td>
<td>Interjection</td>
</tr>
<tr>
<td>JJS</td>
<td>Adjective superlative</td>
<td>VB</td>
<td>Verb, base form</td>
</tr>
<tr>
<td>LS</td>
<td>List item marker</td>
<td>VBD</td>
<td>Verb, past tense</td>
</tr>
<tr>
<td>MD</td>
<td>modal</td>
<td>VBG</td>
<td>Verb, gerund or present participle</td>
</tr>
<tr>
<td>NN</td>
<td>Noun, singular or mass</td>
<td>VBN</td>
<td>Verb, past participle</td>
</tr>
<tr>
<td>NNS</td>
<td>Noun, plural</td>
<td>VBP</td>
<td>Verb, Non-3rd person singular present</td>
</tr>
<tr>
<td>NNP</td>
<td>Proper Noun, singular</td>
<td>VBZ</td>
<td>Verb, 3rd person singular present</td>
</tr>
<tr>
<td>NNPS</td>
<td>Proper Noun, plural</td>
<td>WDT</td>
<td>Wh-determiner</td>
</tr>
<tr>
<td>PD</td>
<td>Predeterminer</td>
<td>WP</td>
<td>Wh-Pronoun</td>
</tr>
<tr>
<td>POS</td>
<td>Possessive ending</td>
<td>WP$</td>
<td>Possessive Wh-Pronoun</td>
</tr>
<tr>
<td>PRP</td>
<td>Personal Pronoun</td>
<td>WRB</td>
<td>Wh-Adverb</td>
</tr>
</tbody>
</table>
APPENDIX B

COMMA TAGGING – ARTICLE LIST
The following list of fifty articles was used to evaluate the performance of the comma tagging system that was presented in Sections 5.6 and 5.7. The articles were randomly chosen from each source and are listed in the order they were chosen. Some of the encyclopedia articles were truncated to keep their size under 10k.

**New York Times (June 6, 2001)**

*Judge Refuses to Delay McVeigh Execution.* Terence Neilan.

*Jury Orders Philip Morris to Pay Ex-Smoker More Than $3 Billion.* Reuters.

*As Senate Shifts, Bush Is Expressing Optimism on Issues.* Frank Bruni.


*Now a Leader, Iverson Turns Image Around.* Chris Broussard.

*Amazon Is Planning to Sell PC’s for the First Time.* Saul Hansell.

*A New Flight Plan.* Opinion.

*A Mountain Road, Silent Monks and Noisy Old Racecars.* George P. Blumberg.

*'The Wild Blue Road’: As a Comrade, He Could Have Been a Contender.* Stephen Holden.

*Harold Ripley, Early Developer of Lens Implants, Dies at 94.* Anahad O’Connor.

**Wall Street Journal**

*Ten O’Clock Tech: Dust Off Your Home Videos.* Arik Hesseldahl.


*Lawyer’s New Service Targets VC Paper Trail.* Lisa Bransten.


*At Elite Universities, a Culture Of Money Highlights Class Divide.* Jonathan Kaufman. June 8, 2001

Home Values Should Grow During Next 10 Years. Chris Gay.


Out of Control. Claudia Rosett. June 14, 2001


Encyclopedia Britannica(2001)

Plants, George Fox, The Spinal Chord, Aircraft Carrier, Tennis, Saktism, Sailing, Printing, Hotel, George Bush Senior

Encyclopedia Encarta(2000)

Vietnam War, Scotland, Arthritis, Digestive System, Negro Leagues, Mormonism, Hemorrhagic Fever, Johannes Brahms, Telephone, Legislative Branch

WorldBook Encyclopedia(1994)

Dublin, Huguenots, David Hume, Las Vegas, Latin Language, Robert Edwin Peary, Sir Robert Peel, Stockholm, Stomach, Stonehenge
APPENDIX C

COMPLETE AUTOMATA CASCADE
... [CO-LST-NP1] NOT SHOWN ...

[CO-LST-NP2]

[CO-VC]

[CO-INF]

[CO-REL1]

[CO-REL2]
Sub-conjunctions (SC):
- while though then
- what why which
- that who whose
- how where

Wherever
- whoever
- whichever
- whatever

whether after
- before when
- whenever

although until
- because
- since if as
- even though once

Adverbial Words:
- month(s) week(s) day(s) year(s) today
- yesterday century(ies) decade(s)
- hour(s) minute(s) second(s) night(s)
- morning(s) evening(s) afternoon(s)
- Monday Tuesday Wednesday Thursday
- Friday Saturday Sunday
- January February March April May June
- July August September October November
- December January Feb Aug Sept Oct Dec Nov
- North South West East Southward Northward
- Eastward Westward Northwest Northeast
- Southwest Southeast Miles Kilometers
- STAART
- [NP] Comma
- [ADV] STAART
- [NP] Comma
- [NP] Comma
- [NP] Comma
- [NP] Comma
CLAUSE NETWORK

[SUB]

Sub-conjunctions(SC):
while though then what why which that whose how where
Wherever whoever whichever whatever
whether after before when whenever
although until because since if as
Eventhough once
APPENDIX D

ENCYCLOPEDIA ENCARITA – EVALUATION SENTENCES
NP Jerome/NNP David/NNP Salinger/NNP ( CC-VS ( VP was/VBD born/VBN ( CC-VC and/CC ( VP raised/VBN ) ) ) ( PP in/IN ( NP New/NNP York/NNP City/NNP ) ) ).

NP The/DT Catcher/NNP ( PP in/IN ( NP the/DT Rye/NNP ) ) ( VP is/VBZ narrated/VBN ( PP by/IN ( NP Holden/NNP Caulfield/NNP ) ) ) ( CO-APS ( NP a/DT 16-year-old/JJ boy/NN ) ( REL2 who/VP ( VP has/VBZ just/RB flunked/VBN ) ) ( PP ( PIN1 out/IN of/IN ( NP his/PRP$ third/JJ private/NNJ boarding/NN school/NN ) ) ) ) ( NP The/DT family/NN 's/POS saga/NN ) ) ( CO-REL1 colored/VBN ( PP by/IN ( NP the/DT suicide/NN ) ) ( PP of/IN ( NP the/DT precocious/JJ oldest/JJ son/NN ) ) ) ( CO-APS ( NP Seymour/NNP ) ( CO-VC ) ( VP informed/VBD ) ( PP by/IN ( NP Salinger/NNP 's/POS work/NN ) ) ( PP during/IN ( NP the/DT next/JJ decade/NN ) ) ).

NP The/DT four/CD blood/NN types/NNS ( NP are/VBP known/VBN ) ( PP as/IN ( LST-NP1 ( NP A/DT ) ) ( NP B/NNP ) ( NP AB/NNP ) ) ( and/CC ( NP O/NNP ) ) ).

NP Blood/NNP type/NN A/NN/NNP ( NP contains/VBZ ( NP red/JJ blood/NN cells/NNS ) ) ( REL2 that/WDT ( VP have/VBP ) ( NP a/DT substance/NN ) ) ( NP A/DT ) ( PP on/IN ( NP their/PRP$ surface/NN ) ) ).

NP In/IN ( NP 1895/CD ) ) ( NP he/PRP ) ( VP married/VBD ) ( NP Elinor/NNP White/NNP ) )

But/CC ( NP Frost's/NNP work/NN ) ( PP during/IN ( NP this/DT time/NN ) ) ( VP was/VBD associated/VBN ) ( PP with/IN ( NP that/DT ) ) ( PP of/IN ( NP the/DT Georgian/JJ poets/NNS ) ) ( CO-APS ( NP a/DT group/NN ) ( PP of/IN ( NP English/NNP writers/NNS ) ) ( REL2 whose/VP ( NP lyric/JJ poetry/NN ) ( VP celebrated/VBD ) ( NP the/DT English/NNP countryside/NN ) ) ) ).

NP In/IN ( NP the/DT title/NN poem/NN ) ( PP of/IN ( NP New/NNP Hampshire/NNP ) ) ) ( NP Frost/NNP ) ( VP makes/VBZ ) ( NP an/DT explicit/JJ statement/NN ) ( PP about/IN ( NP his/PRP$ beliefs/NNS ) ) ).

NP Housman/NNP ( VP was/VBD born/VBN ) ( PP in/IN ( NP Fockbury/NNP ) ) ) ( CO-APS ( NP Worcestershire/NNP ) ) ( CO-VC ) ( VP educated/VBN ) ( PP at/IN ( NP the/DT University/NNP ) ) ( PP of/IN ( NP Oxford/NNP ) ) ).

NP The/DT Acta/NNP Senatus/NNP ( CO-COR ) ( CO-VC ) ( VP included/VBD ) ( NP the/DT opinions/NNS ) ) ( PP of/IN ( NP the/DT chief/JJ speakers/NNS ) ) ( CC-NP ) ( NP the/DT final/JJ decision/NN ) ( PP of/IN ( NP the/DT Senate/NNP ) ) ).
The blindness reply,

they showed

is

by

the fact

that

camel

ate

good

was

growing

more

track

although

grass

was

growing

was

more

track

was

growing

in

thief

imprisoned

several

was

a

a

track

although

grass

was

growing

more

track

was

growing

was

a

a

track

was

growing

was

a

a

track

was

growing

was

a

a

track

was

growing

was

a

a

track

was

growing

was

a

a

track

was

growing

was

a

a

track

was

growing

was

a

a

track

was

growing

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track

was

a

a

track
Aird/NNP and/CC their/PRP$ works/NNS brass/NN instruments/NNS NP NP by/IN ensemble/NN ( ( ) ) ( ( ) ) ( Belgian/NNP Adolphe/NNP Sax/NNP newly/RB the/DT band/NN ( was/VBD probably/RB delayed/VB~ ) ) ( PP by/IN ( NP Dutch/JJ diplomat/NN Robert/NNP van/NNP van/NPP Gulik/NNP ) ) ) , ( CO-REL1 written/VBN ( PP by/IN ( NP Peter/NPP Lovesey/NNP ) ) ( CC-NP and/CC ( NP attitudes/NNS ) ) ( VP are/VBP ) ) ( PP of/IN ( ADV almost/RB ) ( NP equal/JJ importance/NN ) ) ( PP to/TO ( NP the/DT mystery/NN ) ) ) ./.

( ADV Also/RB ) ( PP of/IN ( NP special/JJ interest/NN ) ) ( VP are/VBP ) ( NP novels/NNS ) ( REL1 featuring/VBG ( NP the/DT 7th-century/JJ Chinese/JJ sleuth/NN Judge/NNP Dee/NPP ) ) , ( CO-REL1 written/VBN ( PP by/IN ( NP Dutch/JJ diplomat/NN Robert/NNP van/NNP van/NPP Gulik/NNP ) ) ) , ( CC-NP and/CC ( NP the/DT Victorian/JJ novels/NNS ) ) ( PP by/IN ( NP Peter/NPP Lovesey/NNP ) ) ( CC-NP and/CC ( NP Anne/NNP Perry/NP ) ) ./.

( NP The/DT former/JJ jockey/NN Dick/NNP Francis/NNP ) ( VP created/VBD ) ( NP a/DT number/NN ) ( PP of/IN ( NP detective/NN ) ) ( CC-NP and/CC ( NP thriller/NN heroes/NNS ) ) ( REL1 associated/VBN ) ( PP with/IN ( NP horse/NN racing/NN ) ) ./.

( NP The/DT emergence/NN ) ( PP of/IN ( NP the/DT modern/JJ band/NN ) ) ( VP was/VBD probably/RB delayed/VB~ ) ( PP by/IN ( NP the/DT slow/JJ solution/NN ) ) ( PP of/IN ( NP mechanical/JJ ) ) ( CC-NP and/CC ( NP acoustic/JJ problems/NNS ) ) ( PP in/IN ( NP the/DT design/NN ) ) ( CC-NP and/CC ( NP construction/NN ) ) ( PP of/IN ( NP a/DT low/JJ brass/NN wind/NN instrument/NN ) ) ./.

( NP The/DT carefully/RB prescribed/NN instrumentation/NN ) ( PP of/IN ( NP the/DT band/NN ) ) ( VP began/VBD ) ( INF to/TO ( VP expand/VB ) ) ( PP during/IN ( NP the/DT war/NN ) ) ( INF to/TO ( VP include/VB ) ( ADV newly/RB ) ( NP improved/JJ reed/NN ) ) ( CC-NP and/CC ( NP brass/NN instruments/NNS ) ) ( REL1 constantly/RB being/VBG ) ( VP developed/VBN ) ( PP by/IN ( NP such/JJ builders/NNS ) ) ( PP as/IN ( NP Belgian/NNP Adolphine/NNP Sax/NP ) ) ( CC-NP and/CC ( NP German/NNP Theobald/NNP Boehm/NNP ) ) ./.

( TNP Today/NN ) ( NP bands/NNS ) ( VP are/VBP ) ( NP the/DT country/NN 's/POS dominant/JJ form/NN ) ( PP of/IN ( NP large/JJ amateur/NN ensemble/NN ) ) , ( CO-PP ( PP with/IN ( NP a/DT high/JJ standard/NN ) ) ( PP of/IN ( NP performance/NN ) ) ( REL2 that/WDT ( VP can/MD challenge/VB ) ( NP any/DT amateur/NN musician/NN ) ) ) ./.
Newfoundland is bordered by the Atlantic Ocean on the east, and by the Gulf of Saint Lawrence on the west. The province is bordered on the south by the Gulf of Saint Lawrence, and on the north by the Strait of Belle Isle. The province has infertile soils and peat outcroppings, and its cold climate can be somewhat cooler than the interior because of the cold Labrador current.

Smaller plants include the pitcher plant, sheep laurel, blueberry, and snakehead. The coastal waters of Newfoundland and Labrador constitute one of the world's best fisheries, and many excellent harbors shelter small fishing fleets.

In the mid-1970s, higher fuel prices caused operating costs to rise sharply, further depressing the incomes of the fishermen. Other mineral deposits, notably high-quality uranium in Labrador, offshore petroleum, and natural gas have not been exploited because of the high costs of development.

The Province's population grew slowly from 12,000 in 1763 to 202,000 in 1891.
The iron-mining district around Wabush Lake accounted for about two-fifths of the total population. Ruins of a metal workers' shop and an anvil were littered with hundreds of slag and iron.

The hardy settlers remained, but thereafter without any aid from England.

In 1764, a year after peace was made, Sir Hugh Palliser became naval governor.

The pact, although never written, soon became a tradition in Newfoundland politics.

The commission, although made in 1871 to act in cooperation with a governor from 1934 to 1949, became a tradition in Newfoundland politics.

The Newfoundland government was completely dominated by Smallwood's Liberal Party for nearly two decades after 1949.

Wells retired in January 1996 and was succeeded by Liberal Brian Tobin.

A longtime supporter of civil liberties was instrumental in persuading President Randolph 1.
In/IN (NP large/JJ paintings/NNS) (REL1 often/RB encrusted/VBN) (PP with/IN (LST-NP1 (NP straw/N/, (NP dirt/NN), or/CC (NP scraps/NNS)) (PP of/IN (NP lead/NN))) ), (NP Kiefer/NNP) (VP depicted/VBD) (NP devastated/JJ landscapes/NNS) (CC-NP and/CC (NP colossal/JJ, bombed-out/JJ interiors/NNS)).

NP Critics/NNS) (VP saw/VBD) (NP this/DT recycling/NW) (PP of/IN (NP existing/JJ pictures/NNS) (REL1 known/VBN as/RF (NP appropriation/NN)) (PP as/IN (NP a/DT comment/NN)) (PP on/IN (NP the/DT saturation/NN)) (PP of/IN (NP contemporary/JJ culture/NN)) (PP with/IN (NP imagery/NN)) (REL1 circulated/VBN) (PP by/IN (NP the/DT print/NN)) (CC-NP and/CC (NP broadcast/N media/NNS)).

But/CC (NP much/NN) (PP of/IN (NP the/DT exuberance/NN)) (REL2 which/WDT (NP American/JJ neoexpressionism/NN) (VP was/VBD greeted/VBN)) (VP did/VBD not/RB survive/VB) (NP the/DT downturn/NN) (PP of/IN (NP the/DT art/NN market/NN)) (PP in/IN (NP the/DT late/JJ 1960s/CD)).

NP Plasmids/NNS) (VP carry/VBP) (NP hereditary/JJ information/NN) (PP in/IN (NP the/DT form/NN)) (PP of/IN (NP genes/NNS)) (CO-APS (NP the/DT basic/JJ units/NNS) (PP of/IN (NP inheritance/NN))).

NP Plasmids/NNS) (ADV thus/RB) (VP serve/VBP) (PP as/IN (NP convenient/JJ vehicles/NNS)) (ING for/IN transferring/VBG (NP genes/NNS)) (PP from/IN (NP one/CD organism/NN)) (PP to/TO (NP another/DT)).

NP Other/JJ useful/JJ medical/JJ substances/NNS) (REL1 now/RB manufactured/VBN) (PP with/IN (NP the/DT aid/NN)) (PP of/IN (NP recombinant/JJ plasmids/NNS)) (VP include/VBP) (LST-NP1 (NP human/JJ growth/NN hormone/NN), (NP an/DT immune/JJ system/NN protein/NN) (REL1 known/VBN) (PP as/IN (NP interferon/NN)) (NP blood-clotting/JJ proteins/NNS), and/CC (NP proteins/NNS) (REL2 that/WDT (VP are/VBP used/VBN)) (ING in/IN making/VBG (NP vaccines/NNS)).

NP Resistance/NN plasmids/NNS) (VP are/VBP currently/RB) (NP a/DT topic/NN) (PP of/IN (NP intense/JJ research/NN)) (PP (PIN1 because/IN of/IN) (NP the/DT growing/JJ problem/NN)) (PP with/IN (NP disease-causing/JJ bacteria/NNS)) (REL2 that/WDT (VP are/VBP) (NP resistant/JJ)) (PP to/TO (NP penicillin/NN)) (CC-NP and/CC (NP other/JJ commonly/RF used/RB used/JJ antibiotics/NNS)).

NP It/PRP) (VP is/VBZ) (NP a/DT member/NN) (PP of/IN (NP the/DT Commonwealth/NNP)) (PP of/IN (LST-NP2 (NP Nations/NNPS)), (NP an/DT association/NN) (PP of/IN (NP nations/NNS)) (REL2 that/WDT (VP includes/VBZ) (NP the/DT United/NNP Kingdom/NNP) and/CC (NP a/DT number/NN)) (PP of/IN (NP its/PRP$ former/JJ dependencies/NNS))).
About/RB (NP 1,000/CD Tuvaluans/NNPS) (CC-VS (VP live/VBP) (CC-VC (VP work/VB) (NP overseas/NN))) (PP in/IN (NP the/DT overseas/NN mining/NN industry/NN) (PP on/IN (NP Nauru/NNP)))) ./.

(NP Social/JJ life/NN) (VP centers/VBZ) (PP around/IN (CC-NP and/CC (NP family/NN gatherings/NNS))) ./.

(NP Income/NNP) (PP from/IN (NP a/DT trust/NN fund/NN)) (REL1 established/VBN) (PP by/IN (LST-NP1 (NP Australia/NNP))) (NP New/NNP Zealand/NNP) (CC-VS (NP the/DT United/NNP Kingdom/NNP)) (PP in/IN (NP 1987/CD)) (VP provides/VBZ) (PP about/IN (NP half/NN)) (PP of/IN (NP the/DT government/NN 's/POS recurring/JJ budget/NN requirements/NNS)) ./.

(NP A/DT shipping/JJ line/NN) (VP provides/VBZ) (NP limited/JJ international/JJ service/NN) (CO-S and/CC (NP a/DT small/JJ government/NN freighter/NN) (VP shuttles/VBZ) (PP among/IN (NP the/DT outer/JJ islands/NNS)) ./.

(NP All/DT citizens/NNS) (REL1 aged/VBN) (NP 18/CD) (CC-NP or/CC (NP older/JJR)) (VP can/MD vote/VB) ./.

(PP Between/IN (NP 1820/CD)) (CC-NP and/CC (NP 1870/CD American/NNP)) (CC-NP and/CC (NP British/JJ whalers/NNS)) (VP frequented/VBD) (NP the/DT islands/NNS) (CO-S and/CC (NP some/DT) (VP settled/VBD) (ADV ashore/RB)) ./.

(NP The/DT GEIC/NNP) (CC-VS (VP set/VBD up/RP) (NP a/DT temporary/JJ headquarters/NN) (PP during/IN (NP the/DT Japanese/JJ occupation/NN)) (PP of/IN (NP the/DT Gilbert/NNP Islands/NNPS)) (PP in/IN (NP 1942/CD)) (CC-VC but/CC (VP moved/VBD) (NP the/DT administration/NN) (ADV back/RB)) (PP to/TO (NP Tarawa/NNP)) (SUB after/IN (NP the/DT United/NNP States/NNPS) (VP drove/VBD) (NP the/DT Japanese/NN)) (PP from/IN (NP the/DT Gilberts/NN)) (PP in/IN (NP 1943/CD)) ./.

(PP At/IN (NP the/DT age/NN)) (PP of/IN (NP 13/CD)) (NP she/PRP) (VP went/VBD) (INF to/TO (VP live/VB)) (PP with/IN (NP her/PRP$ father/NN)) (PP in/IN (NP Nashville/NNP)) (CO-APS (NP Tennessee/NNP)) ./.

(PP In/IN (NP 1998/CD Winfrey/NNP)) (VP appeared/VBD) (PP as/IN (NP the/DT character/NN Sethe/NNP)) (PP in/IN (NP the/DT film/NN Beloved/NNP)) (REL1 which/WDT (VP was/VBD adapted/VBN) (PP from/IN (NP the/DT novel/NN)) (PP by/IN (NP Toni/NNP Morrison/NNP)) (SUB after/IN (NP Tarawa/NNP)) ./.

(NP Intergroup/NNP offices/NNS) (PP in/IN (ADV most/RBS) (NP urban/JJ areas/NNS)) (VP provide/VBD) (NP information/NN) (PP on/IN (NP times/NNS)) (CC-NP and/CC (NP places/NNS)) (PP of/IN (NP nearby/JJ meetings/NNS)) ./.
autopsy/NN decrease/VB under/IN infants/NNS the/DT victim/NN cases/NNS infant/NN is/VBZ conducted/VBN, ( CO-REL1 founded/VBN of/IN the/DT House/NNP with/IN the/DT and/CC susceptible/JJ ) ( CO-SUB ADV of/IN the/DT medical/JJ history/NN ) ( CO-APS NP the/DT infant/NN ) ( CC-NP and/CC ( PP of/IN ( NP parents/NNS ) ) ) , ( VP is/VBZ conducted/VBN ) ( PP in/IN ( NP all/DT suspected/JJ SIDS/NNP cases/NNS ) ) ./. 

( PP of/IN ( NP the/DT infant/NN ) ) , ( CO-REL1 including/VBG ( LST-NP1 ( NP a/DT complete/JJ autopsy/NN ) ) , ( NP examinaion/NN ) ( PP of/IN ( NP the/DT infant/NN s/POS sleeping/JJ environment/NN ) ) ,/ , and/CC ( NP review/NN ) ) ( PP of/IN ( NP the/DT medical/JJ history/NN ) ) ( PP of/IN ( NP both/DT the/DT victim/NN ) ) ( CC-NP and/CC ( NP parents/NNS ) ) ) ,/ , ( VP is/VBZ conducted/VBN ) ( PP in/IN ( NP all/DT suspected/JJ SIDS/NNP cases/NNS ) ) ./. 

( NP A/DT thorough/JJ examination/NN ) ( PP of/IN ( NP the/DT infant/NN ) ) ,/ , ( CO-REL1 including/VBG ( LST-NP1 ( NP a/DT complete/JJ autopsy/NN ) ) ,/ ( NP examinaion/NN ) ( PP of/IN ( NP the/DT infant/NN s/POS sleeping/JJ environment/NN ) ) ,/ , and/CC ( NP review/NN ) ) ( PP of/IN ( NP the/DT medical/JJ history/NN ) ) ( PP of/IN ( NP both/DT the/DT victim/NN ) ) ( CC-NP and/CC ( NP parents/NNS ) ) ) ,/ , ( VP is/VBZ conducted/VBN ) ( PP in/IN ( NP all/DT suspected/JJ SIDS/NNP cases/NNS ) ) ./. 

( NP A/DT thorough/JJ examination/NN ) ( PP of/IN ( NP the/DT infant/NN ) ) , ( CO-REL1 including/VBG ( LST-NP1 ( NP a/DT complete/JJ autopsy/NN ) ) ,/ ( NP examinaion/NN ) ( PP of/IN ( NP the/DT infant/NN s/POS sleeping/JJ environment/NN ) ) ,/ , and/CC ( NP review/NN ) ) ( PP of/IN ( NP the/DT medical/JJ history/NN ) ) ( PP of/IN ( NP both/DT the/DT victim/NN ) ) ( CC-NP and/CC ( NP parents/NNS ) ) ) ,/ , ( VP is/VBZ conducted/VBN ) ( PP in/IN ( NP all/DT suspected/JJ SIDS/NNP cases/NNS ) ) ./. 

( NP A/DT thorough/JJ examination/NN ) ( PP of/IN ( NP the/DT infant/NN ) ) , ( CO-REL1 including/VBG ( LST-NP1 ( NP a/DT complete/JJ autopsy/NN ) ) ,/ ( NP examinaion/NN ) ( PP of/IN ( NP the/DT infant/NN s/POS sleeping/JJ environment/NN ) ) ,/ , and/CC ( NP review/NN ) ) ( PP of/IN ( NP the/DT medical/JJ history/NN ) ) ( PP of/IN ( NP both/DT the/DT victim/NN ) ) ( CC-NP and/CC ( NP parents/NNS ) ) ) ,/ , ( VP is/VBZ conducted/VBN ) ( PP in/IN ( NP all/DT suspected/JJ SIDS/NNP cases/NNS ) ) ./.
implication/NN ) ) ( SUB that/IN ( NP this/DT language/NN ) ( VP was/VBD ) ( NP a/DT racial/JJ fact/NN ) ) ( PP ( PIN1 instead/RB of/IN ) ( NP a/DT social/JJ fact/NN ) ) ./.

( CO-PP ( PP In/IN ( NP each/DT column/NN ) ) ) ) ) ( NP there/EX ) ( VP are/VBP ) ( NP five/CD beads/NNS ) ( PP below/IN ( NP the/DT crossbar/NN ) ) ) ) ( CO-REL2 ( NP each/DT ) of/IN which/WDT ( VP represent/VBP ) ( NP one/CD unit/NN ) ) ), ( CC-NP and/CC ( NP two/CD beads/NNS ) ) ( PP above/IN ( NP the/DT crossbar/NN ) ) ) ) ( CO-REL2 ( NP each/DT ) of/IN which/WDT ( VP represent/VBP ) ( NP five/CD units/NNS ) ) ) ./.

( NP The/DT elemental/JJ unit/NN ) ( PP of/IN ( NP ecology/NN ) ( VP is/VBZ ) ( NP the/DT absolute/JJ charge/NN ) ) ( PP on/IN ( NP a/DT single/JJ electron/NN ) ) ( CC-NP or/CC ( NP proton/NN ) ) ./.

( CO-SUB If/IN ( NP a/DT current/NN ) ( PP of/IN ( NP 1/CD abampere/NN ) ( VP flows/VBZ ) ( PP in/IN ( NP a/DT wire/NN 1/CD centimeter/NN ) ) ) ( ADJ long/JJ ) ), ( NP the/DT wire/NN ) ( VP is/VBZ pushed/VBN ) ( ADV sidewise/RB ) ( PP with/IN ( NP a/DT force/NN ) ) ( PP of/IN ( NP 1/CD dyne/NN ) ) ( PP by/IN ( NP a/DT magnetic/JJ field/NN ) ) ( PP of/IN ( NP 1/CD oersted/NN ) ) ( REL1 acting/VBG ) ( PP at/IN ( NP right/NN angles/NNS ) ) ( PP to/TO ( NP the/DT wire/NN ) ) ./.

( NP The/DT SI/NNP unit/NN ) ( PP of/IN ( NP electrical/JJ work/NN ) ( VP is/VBZ ) ( NP the/DT watt/NN ) ) ./.

( ADV Thus/RB ) ./, ( NP a/DT micromicrofarad/NN ) ( VP is/VBZ ) ( NP a/DT trillionth/NN ) ( PP of/IN ( NP a/DT farad/NN ) ) ) ) ( NP a/DT microampere/NN ) ( VP is/VBZ ) ( NP a/DT millionth/JJ ) ( PP of/IN ( NP an/DT amper/NN ) ) ) ( NP a/DT millivolt/NN ) ( VP is/VBZ ) ( NP a/DT thousandth/JJ ) ( PP of/IN ( NP a/DT volt/NN ) ) ) ) ( NP a/DT millihenry/NN ) ( VP is/VBZ ) ( NP a/DT thousandth/JJ ) ( PP of/IN ( NP a/DT henry/NN ) ) ) ) ( NP a/DT kilowatt/NN ) ( VP is/VBZ ) ( NP 1000/CD watts/NNS ) ) ./, ( CO-S and/CC ( NP a/DT megohm/NN ) ( VP is/VBZ ) ( NP 1/CD million/cd ohms/NNS ) ) ./.

( NP Forests/NNS ) ( PP of/IN ( NP larch/NN ) ) ( CC-NP and/CC ( NP cedar/NN ) ) ( VP cover/VB ) ( NP more/JJR ) ( PP than/IN ( NP 40/CD percent/NN ) ) ( PP of/IN ( NP the/DT republic/NN ) ) ) ( CO-SUB while/IN ( NP grasses/NNS ) ) ( CC-NP and/CC ( NP other/JJ steppe/NN vegetation/NN ) ) ( VP dominate/VB ) ( PP in/IN ( NP the/DT plains/NNS ) ) ) ./.

( TRN For/IN example/NN ./, ) ( NP shamanist/NN traditions/NNS ) ( VP continue/VBP ) ( PP among/IN ( NP the/DT people/NNS ) ) ( PP despite/IN ( NP Soviet/JJ attempts/NNS ) ) ( INF to/TO ( VP abolish/VB ) ( NP religion/NN ) ) ./.

( NP Khakassia/NNP ) ( VP is/VBZ administered/VBN ) ( PP by/IN ( NP an/DT elected/JJ supreme/NL legislature/NN ) ) ( CC-NP and/CC ( NP an/DT elected/JJ council/NN ) ) ( PP of/IN ( NP ministers/NNS ) ) ./.

( PP In/IN ( NP October/NNP 1930/CD ) ) ( NP the/DT Khakass/NNP Autonomous/NNP Oblast/NNP ) ( VP was/VBD formed/VBN ) ( PP as/IN ( NP part/NN ) ) ( PP of/IN ( NP Krasnoyarsk/NNP territory/NNP ) ) ./.
But the Almoravids returned to Sevilla and deposed al-Mutamid.

He rebuilt churches and promoted education.

Martin was bishop of Tours in the 4th century, and was mentioned in the 6th century by the noted Frankish historian Gregory of Tours.
LIST OF REFERENCES


K Church. 1988. *A stochastic parts program and noun phrase parser for unrestricted text.* In Proceeding of the Second Conference on Applied Natural Language Processing, Austin, Texas, Pages 136-143.


G Hirst and St-Onge, D. 1998. Lexical Chains as representations of context for the


G Nunberg. 1990 *The Linguistics of Punctuation*. Center for the Study of Language and Information.


