ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

FACULTY OF INFORMATICS
DEPARTMENT OF INFORMATION SCIENCE

AN INTEGRATED APPROACH TO AUTOMATIC COMPLEX SENTENCE PARSING FOR AMHARIC TEXT

A thesis submitted to the School of Graduate Studies of Addis Ababa University in partial fulfillment of the requirements for the Degree of Master of Science in Information Science.

BY

DANIEL GOCHEL AGONAFER

JUNE 2003

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

FACULTY OF INFORMATICS
DEPARTMENT OF INFORMATION SCIENCE

AN INTEGRATED APPROACH TO AUTOMATIC COMPLEX SENTENCE PARSING FOR AMHARIC TEXT

BY
DANIEL GOCHEL AGONAFER

Signature of the Board of Examiners for Approval

_____	_____
_____	_____
_____	_____
_____	_____

Acknowledgment

My deepest gratitude goes to my parents, W/ro Amelework Amakelew and Ato Gochel Agonafer, my sisters Ehitagegnew, Haregewoin, and Netsanet, and my brothers Yitbarek and Addisu, and my aunt Bayush, who have all been my backbone in terms of providing me with moral and financial supports during my stay in the University.

My heartfelt gratitude goes to my advisors, W/rt Atelach Alemu, who was my driving force throughout this study, Ato Mesfin Getachew, for his commitment, and constant support, and Dr. Abebe G/Tsadik, without whom such a study would have been incomplete. They have been providing me constructive and invaluable comments which have brought life to this study. Without the presence of their constructive, critical and friendly suggestions, this project would have been more demanding.

I am also indebted to Prof. Baye Yemam, who shared me his precious time voluntarily and allowed me to get his linguistic advice periodically.

My special thanks goes to my friends, Ato Alem Aregawi and Ato Tessema Mamo for their professional and moral support.

My indebtedness goes to my beloved friend Mr. James Guterez (Frank), for his unreserved, all-rounded support and invaluable inspiration, and his wife Mrs. Guterez, for their moral support and enthusiasm. I would also express my gratitude to Engineer Sisay Denboba who technically assisted me during the coding of the algorithm.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Abbreviations and symbols used

Symbols

( ) What is inside is optional except those used to indicate cite (source) of documents used

\ To separate words from their tags

Abbreviations

N	Noun in all forms
NP	A preposition not separated from a noun
NC	A conjunction not separated from a noun
NV	Verbal nouns
NB	Noun prefixed with balä
V	Verb in all forms except auxiliary, compound and all forms of auxiliary and compound verbs
AUX	Auxiliary verbs and all their other forms
VCO	**C**ompound verbs
VP	A preposition not separated from noun
VC	A verb prefixed or suffixed by a conjunction
J , Adj	An adjective
JC	A conjunction not separated from an adjective
JNU	A numeral used as an adjective
JPN	A noun not separated from a preposition and that function as an adjective
JP	An adjective not separated from a preposition
PREP	A preposition
ADV	An adverb
ADVC	An adverb not separated from a conjunction
C	A conjunction
COMP	Complement

REL	Relative clause
ITJ	Interjections
PP	Prepositional Phrase
NP	Noun Phrase
VP	Verb Phrase
AdjP	Adjectival Phrase
AdvP	Adverbial Phrase
NL	Natural Language
NLP	Natural Language Processing
NLU	Natural Language Understanding

APPENDICES

ABSTRACT

Natural language processing is a research area which is becoming increasingly popular each day for both academic and commercial reasons. Higher NLP systems (e.g., machine translation) are materialized only when the lower ones (e.g., part-of-speech tagger, syntactic parser) are successfully built. This functional dependency exists even among the lower NLP systems. A morphological analyzer can be an important component for a part-of-speech (POS) tagger particularly in dealing with unknown words. A POS tagger, which is a system that uses various sources of information to assign possibly unique POSs to words, in turn, can be used as an input to a syntactic parser. Writers in the area of NLP argue that if the POS tagger is accurate, this method is an excellent one. This thesis can be taken as an attempt to integrate ideas and outputs of previously attempted Amharic NLP prototypes towards solving a bit further problem in the NLP of the language, i.e. automatic Amharic complex sentence parsing.

Syntactic parsing underlies most of the applications in natural language processing. Parsers are already being used extensively in a number of disciplines such as in computer science (for compiler construction, database interfaces, artificial intelligence, etc), and in linguistics (for text analysis, corpora analysis, machine translation, etc.).

Although there have been some comprehensive studies of Amharic syntax from a linguistic perspective, attempts for investigating it from a computational point of view is a very recent story. In this thesis, Amharic word and phrase classes, sentence formalisms, morphological properties peculiar to complex sentence formation in the language, and

attempts to extract such features that enable implementation of automatic Amharic complex sentence parser is presented.

The sample data used in this study has been taken from references that are widely used in the teaching-learning process of the language. This data has also been manually analyzed, tagged, parsed, and then used as a corpus to extract the grammar rules and to assign probabilities. Algorithms that can use the morphological, lexical and syntactic properties of the language have been customized and modified.

Experiments have been conducted in this study using the training set and test set. The first experiment was conducted on the part-of-speech tagger to see the state of its performance when a morphological analysis is embedded in it. The result of this experiment showed that the tagger attained 98.7% and 94% of accuracies on the training set and the test set, respectively.

The experiments on complex sentence parsing showed 89.6% accuracy result on the training set and 81.6% accuracy result on the test set prepared for this purpose.

CHAPTER ONE

INTRODUCTION

1.1 Background

A given human language, whether in the written or spoken mode, is a fundamental part of human communication. Any hope of providing computer systems that claim intelligence approaching that of a human, therefore, rests on the hope of providing communication in natural language. Machine translation of natural languages accurately and in real time is a typical example of such systems that activates researches in the area of computational linguistics.

Information retrieval systems is another most frequently mentioned driving force towards instilling computer systems with natural languages. Most scholars in the field of information retrieval (e.g., Salton and his co-workers) argue that a large part of the information stored in bibliographic retrieval systems consists of natural language data, i.e., in the form of texts and utterances of English, French, Amharic, etc. (Salton, 1983). They indicated that the use of natural language search statements, which is the preference of many users of the systems, could raise the effectiveness as well as the efficiency of the retrieval operations by making possible the formulations of precise requests that correctly reflect user needs and simplify the user-system interactions (Salton, 1989).

Accordingly, the general goal for most computational linguistics is to inspire the computer with the ability to understand and generate natural language so that eventually people can address their computers through text and/or speech as though they were addressing a person (Atelach, 2002). Thus, Natural Language Processing (NLP), also

called Computational Linguistics (CL), is that area of research and application that explores how natural language, which is entered into a computer system, can be manipulated and stored in a form that preserves certain aspects of the original (Harris, 1985). NLP has also been described as the ability of computers to generate and interpret natural language (Lenat et al, 1990). According to the recently published *Handbook of Natural Language Processing* (Dale, 2002), NLP is concerned with "the design and implementation of effective natural language input and output components for computational systems".

Therefore, in order to fully realize the aforementioned impressive capabilities of NLP, intensive research works should be conducted at different levels of natural language processing (NLP): phonological (i.e. sounds or combinations of sounds), morphological (processing of individual word forms), lexical (procedures operating on full words), syntactic (grouping the words of a sentence into structural units), semantic (adding contextual knowledge to the purely syntactic process in order to restructure the text into units that represent the actual meaning of a text), and pragmatic (using additional information about the social environment in which a given document exists). As Warner (1987) stated, the incorporation of information about the underlying organization of the data at various levels as described above helps in understanding the nature of the language under consideration and, therefore, to effect the desired systems.

Besides, NLP demands deep natural language understanding (NLU) and modeling the natural language so that computer programs that act appropriately on the information contained in the text or utterance of the language can be developed. However, the fact

that NL involves a large number of classes and relationships whose existence is not transparent from its surface structure has made NLP an extremely complicated task. Similarly, the assignment of words to their classes and identification of the relation they have with other words in a sentence, paragraph, or text is difficult (Mao, 1997; Mesfin, 2001; Kibur, 2002), adding to the complexity of the task of NLP.

Generally speaking, for computers to understand natural languages, they require systems that help them handle the myriad word classes and relationships involved in NL. To that end, works in computational linguistics and NLP systems are playing important roles in contributing to the development of such systems that process NL at different levels of complexity to come-up with a general NLU system (Allen, 1996). In this regard, there are systems developed for processing NL at phoneme, word, sentence, and pragmatic levels. Uibo (2001) as cited in Kibur (2002) has indicated that these systems are developed in such a way that the output of a lower system can serve as an input for the next higher level. For instance, the output of a part-of-speech tagger could serve as an input for syntactic analyzer (or parser), which, in turn, can serve as input for a semantic parser and automatic machine translation.

In fact, several NLP research results (such as Atelach, 2002; Genereux, 2001) indicate that syntactic parsing underlies most of the applications in natural language processing. Wilks and Spark Jones (1983) also indicated that a central component of most NLP systems is the parser, a computational process that takes in individual sentences or connected texts and converts them to some representational structure useful for further processing. According to Grishman, (1994), syntactic analysis is one of the foremost

tasks to be accomplished in NLP and its overall objective is to determine the meaning of a sentence. In practice, this involves translating the natural language input into a language with a simple semantics, which seeks the exact contextual meaning of a sentence or into a language, which can be interpreted by an existing computer system. In most systems, the first stage of this translation is syntactic analysis, i.e., the determination of a sentence structure.

The syntactic structure of a sentence indicates the way words in the sentence are related to each other. It also indicates how the words are grouped together into phrases, what words modify other words, and what words are of central importance in the sentence (Allen, 1995). This structure is commonly represented by means of a sentence diagram, where the syntaxes are encoded as a list of structure or a parse tree – a tree in which the nodes correspond to sentence constituents such as phrases (Rauch-Hindin, 1986; Shutzer, 1987).

In general, sentence parsing can be described as a method of analyzing the various parts of a string to determine whether or not the string is a sentence in the language. It is, therefore, a crucial task in NLP and a method that deals with the decomposition of a sentence into its major subparts, namely noun phrase (NP), verb phrase (VP), noun (N), verb (V), etc. In addition to this, parsing deals with a number of sub problems such as identifying constituents that can fit together, testing the compatibility of number and tense etc (Metzer et al, 1989 as cited in Atelach, 2002). A number of syntactic parsers have already been and are being developed for many languages in the world. There are also many systems (e.g. machine translation systems) that comprise sentence parser as

their core module, bearing out the fact that this particular NLU system is really indispensable for the materialization of further NLU systems. Thus, it is the purpose of this study to develop a syntactic analysis system useful for parsing Amharic sentences.

1.2 Statement of The Problem

According to the latest census results, Amharic is a mother tongue of more that 17 million people. The language is also used as a second language for over 5 millions people (ECSA[1], 1998). It has also been, for a long period, the principal literal language and medium of instruction and school subject in primary and secondary schools of the country. Moreover, it is the official working language of the Ethiopian Federal Government and five of the regional states, all of which make the language to be predominantly used in word processing activities in different offices. Furthermore, there are also a large number of documents, published and unpublished, written and recorded in Amharic.

There have been comprehensive studies of Amharic syntax from a linguistic perspective. However, attempts for investigating Amharic extensively from a computational point of view were almost nonexistent for a long time. Very recently, NLP for local languages such as Amharic has become an area of research interest and there are some studies that have been and are being conducted on NLP of Amharic. These include, for instance, 'an Amharic Word Parser' by Abiyot (2000), 'a Part-of-Speech Tagger (POST)' by Mesfin (2001), 'an Amharic Sentence Parser' by Atelach (2002), 'Statistical Morphological

[1] ***ECSA*** stands for Central Statistical Authority of the Federal Democratic Republic of Ethiopia

Analysis for Amharic' by Tesfaye (2002), and 'A Morphological Synthesizer for Amharic Perfectives' by Kibur (2002).

Although there is a lot to be done in order to solve the need for all sorts of NLP systems (like semantic analysis, syntactic parser, machine translation, information storage and retrieval systems, etc.) for Amharic, integration of previous studies in order to solve a bit further NLP problem of Amharic, for example, the development of complex sentence (text) parsing system for the language is among the tasks that demand priority. More specifically, there is morphological analysis study for Amharic and a POS tagger that is totally statistical. The effect of the morphological analysis on the performance of the POS tagger, and the sumtotal effect of these two systems on the performance of Amharic syntactic analysis, particularly, on Amharic complex sentence parsing is an active research area. This is largely due to the fact that a sentence parser is a central component of natural language processing, and is used to solve basic problems (such as language comprehension and production work) that most NLP applications face. In this regard, Atelach (2002) indicated that the absence of a syntactic parser for Amharic limits higher forms of NLP like semantic analysis, discourse integration, machine translation, spell checking and grammar, etc. for the language under consideration.

In spite of such decisive roles that a syntactic analyzing system may play for the development of subsequent NLP systems, it seems that there was no study conducted in this area of Amharic until very recently. To the best of the researcher's knowledge, the automatic simple sentence parser for Amharic developed by Atelach (2002) was the first in this regard. This prototype parser had an average of 85% accuracy in the test sets used

for the purpose of the study. Nonetheless, the study had the following limitations. First, the automatic sentence parser developed parses only 4-word long Amharic simple sentences. Second, the prototype developed was totally dependent on the statistical tagger. In addition to this, the bi-gram lexical co-occurrence, a technique used in the study to guess for unknown words, was found to be not promising, which the researcher herself indicated. Thirdly, the size of the sample sentences prepared and used for training and testing purposes in the study was small (i.e., 100 simple declarative sentences).

Furthermore, it is obvious that integration of related research works that have already been conducted and are being conducted in the area of NLP of Amharic is also important so as to ensure the continuity of such studies there by accomplishing the desired result. In the local case, for instance, as it has already been indicated, research works on a word parser for Amharic and on statistical morphological analysis for the same language were conducted. Also, the studies on the POS tagger and on the simple sentence parser for Amharic used purely statistical techniques in lexical and structural ambiguities resolution, respectively. However, combining the statistical techniques and the morphological analysis could produce a better result (Atelach, 2002).

Therefore, the major concern of the present study is to integrate some of the initiatives taken to develop NLU systems for Amharic in order to develop a bit further NLU system for the language. More specifically, this study aims at handling complex sentences of the language with particular emphasis on Amharic complex declarative sentences. Besides, it is the intent of this study to prepare and use corpus larger than those used in the previous work (possibly, 350 sentences collected from grammar reference books that are widely

used in the teaching-learning process of the language). Furthermore, the study attempts to incorporate statistical lexical co-occurrence and morphological properties of the language (such as inflections) in order to deal with structural ambiguity resolution.

1.3 Objectives of The Study

1.3.1 General Objective

The general objective of this study is to explore the possibilities of integrating previously attempted studies in the area of morphological analysis, POS tagging, and simple sentence parsing in order to be able to parse complex Amharic sentences that are composed of a simple noun phrase and a complex verb phrase (i.e., a phrase which a sentence is embedded in).

1.3.2 Specific Objectives

The specific objectives of the study include:

- Review the basic word categories, morphological properties, phrase structures, and the various kinds of sentences of Amharic with the aim of investigating patterns that allow computer representation;
- Collect sample complex sentences to be used in the experiment;
- Rebuild the database of the part-of-speech tagger (Mesfin, 2001) using the stems of words taken from the sample corpus and calculate the lexical and transitional probabilities for them.
- Design and develop a table containing four fields, namely stem category, prefix, suffix and new category.
- Generate the grammar rules appropriate for the language;

- Use a combination of statistical lexical co-occurrence techniques and morphological analysis in order to guess for unknown words;
- Customize the parsing algorithm that was used for automatic simple sentence parser for Amharic (Atelach, 2002);
- Develop a prototype parser that will implement the findings of the study and test it to measure its performance;
- Forward recommendations for further studies;

1.4 Methods

Obviously, developing an NLP system such as the one under consideration requires thorough investigation and identification of the properties of the target language. The following methods have, therefore, been used in order to achieve the objectives of this research.

1.4.1 Literature Review

Various appropriate and related literature resources, i.e. books, research reports, journal articles, manuals, and other published and unpublished documents including those from the Internet have been reviewed for the purpose of this study. All these helped the researcher understand both the issues regarding NLP, particularly automatic sentence parsing (e.g., approaches, techniques and strategies), and issues of the language considered (i.e., the basic word categories, morphological property, phrase structure, and the various types and kinds of sentences of Amharic). This understanding, in turn, has enabled the researcher to implement the features of the language that have been found appropriate to the study, and to adopt the parsing algorithm appropriately.

1.4.2 Discussion

Successive discussion with linguists and experts in the area of Amharic language at Addis Ababa University/Institute Language Studies have also been made for better understanding and analysis of Amharic complex sentences and its sub-components, particularly complex phrase structures of the language.

1.4.3 Data Collection

350 sentences were collected from books entitled "Amharic for Beginners" by Frydenlund and Svensen (1998), and **Yä Amarigna Säwasäw** "The Amharic Grammar" by Baye (1987)[2] and, both of them were prepared with the purpose of serving as references for teaching Amharic language at secondary and tertiary levels, respectively. These books were chosen as references after discussion with thesis advisors and the author of the second book, Prof. Baye Yemam. The sentences were selected randomly, though there was some judgment by the researcher such as all the sentences constitute two or more of the phrase classes and each sentence is composed of a simple noun phrase and a complex verb phrase. Also each of them contains one of the various Amharic dependent clause types (i.e. relative clause, reason clause, result clause or time clause) all discussed in chapter three. All the sentences in the sample had first been manually morphologically analyzed, tagged, and parsed by 2 experts of the language at the academic department of the Ministry of Defence, and by the researcher. It was then given to the linguistic advisor in order to get feedback on the correctness of the manual parse.

[2] The year is according to the Ethiopian calendar

1.4.4 Parsing Techniques and Prototype Development

A morphological analysis component is adopted in this study in the course of preprocessing the input Amharic text to be parsed. The part-of-speech tagger for Amharic (Mesfin, 2001) is also incorporated in the study in order to tag the string of stems that the morphological analyzer outputs. A table that contains the Amharic affixes as well as the possible word category they may be found in is also developed in order to check for any categorical changes that the tagged stems may exhibit when they are synthesized their respective affixes. The bottom up chart-parsing algorithm that was also adopted by Atelach (2002), has been customized here too with some modifications in order to generate the sentence parse. Moreover, the necessary lexical dictionary, Probabilistic Context Free Grammar (PCFG) rules, and statistical databases have been designed and implemented. The syntactic structure rules have been extracted from the sample sentences, and the development of the prototype and implementation of the parsing algorithm is carried out using an imperative language (Visual Basic programming environment) although literature indicate that NLP lends itself more to declarative languages (e.g. Prolog).

1.4.5 Testing Techniques

280 sentences (80% of the sample corpus) were picked randomly as training set while the rest 70 sentences (20%) from the corpus were used as a test set. The experiment was conducted in two phases: first on the training set and next on the test set and the results have been evaluated. The parse outputs have been crosschecked with manually parsed sentences and an iterative improvement has been carried out on the system in order to deal with the erroneous parses. This has been done until the prototype reaches the

maximum possible level which is the closest to the one that has been found in the manual parse. An application test, a test on more Amharic complex sentences taken from the same sources as the training set, has been conducted to develop confidence on the performance of the selected model and the observed results are reported. Finally, the figures obtained from the observed results have been statistically summarized and analyzed in a way that is suitable to report the attained accuracy level.

1.5 Application of Results

So far in this chapter it was mentioned that syntactic parsers are one of the core components of higher level NLP systems. That means, parsing systems would also play decisive roles in many areas of NLP for Amharic language. Hence, researchers who involve themselves in increasing the capability of computers in processing Amharic language may be benefited from the results of this study. Specifically, researchers interested in the area of phrase recognition, conceptual parsing, machine translation, spell checker, text summarization, etc are among the top beneficiaries. Besides, linguists and students in the field of Amharic language can also apply the output of this research to parse sentences in the language automatically. The output can also be used in language teaching, for recognition of phrasal categories, and to see the relationship between words in a sentence.

1.6 Scope of The Study

The scope of this study is confined to dealing with Amharic complex declarative sentences that are formed in one of the three possible ways of complex sentence

formation, i.e. from a simple noun phrase and a complex verb phrase. This is because of such resource limitations as time, cost and labor. Therefore, this prototype parses complex Amharic sentences that are composed of a simple noun phrase and a complex verb phrase (predicate).

1.7 Limitation of The Study

This study has the following limitations:

1. All kinds of Amharic sentences are not considered in this study.
2. The prototype developed used a simple morphological analysis prepared for the purpose of this study. This is due to lack of the source code of the morphological analyzer for Amharic which was previously developed.
3. The complex sentences that are included in the sample do not exhibit complex noun phrases.
4. Moreover, the size of the corpus is still very small.
5. The prototype developed supplies syntactic analyses for a given Amharic complex sentences, demarcating its constituents, labeling each, identifying the parts of speech of every word used in the sentence. However, it does not offer grammatical analysis for the affixes (e.g., subject markers, direct object markers) unless the affix plays a special role (such as a relativizer) in the formation Amharic complex sentences).

1.8 Organization of The Thesis

This study is organized in six chapters. Chapter one introduces what an NLP is and the role of a syntactic analysis for NLP. This chapter also presents the statement of the problem and the objective of the study. Different approaches and strategies for developing an automatic sent parser and issues related to sentence parsing are discussed in chapter two. The third chapter describes the Amharic language, i.e. the various word classes, phrase categories, and complex sentence formalisms in the language. Also, pertaining to complex phrases and sentence structures in the language are discussed here. Chapter four describes the design of the lexicon for Amharic complex sentence parser. The algorithm designed, the experiment conducted, and the results observed are reported in chapter five. Finally, the conclusions and recommendations made based on the findings of the study are presented in chapter six.

CHAPTER TWO

REVIEW OF RELATED LITERATURE

2.1 Introduction

In the development of parsers for the purpose of examining how the syntactic structure of a sentence can be computed, it is a standard practice to consider two things: the *grammar* and the *parsing technique*. The grammar is a formal specification of the structures allowable in the language while the parsing technique is the method of analyzing a sentence to determine its structure by using the grammar as the source of syntactic knowledge.

This chapter discusses the above two core concepts in automatic sentence parsing together with others. The first section gives some overview of parsing and its evaluation criteria. Following this, the different approaches and techniques to the task of sentence parsing are reviewed. The later sections of this chapter discuss the grammar rules and the lexicon under the title 'knowledge required by the parser,' and, finally, some previous Amharic NLP research endeavors that are incorporated (and /or adapted) into this study are summarized subsequently.

2.2 Parsing

The term *parsing* is derived from the Latin phrase *pars orationis* (part of speech) and it refers to the process of assigning a part of speech (Noun, Adjective, etc.) to each word in a sentence, and grouping the words into phrases (Andrew, 2000). Nonetheless, in NLP the task of parsing may be carried out at a word or sentence level. Word parsing is a process of tokenizing a word into its components or individual morphemes. Unlike in

stemming of words that is done for purposes such as statistical analysis, in word parsing the word should be tokenized into morphologically valid components. These tokenized components will further be analyzed for their contribution to the meaning and categorization of the whole word (Abiyot, 2000).

Sentence Parsing (also called syntactic parsing) is a procedure that explores various ways of combining grammatical rules to find a combination that generates a tree that could represent the structure of the input sentence. In other words, it is a task in NLP in which a flat input sentence is converted into a hierarchical structure that corresponds to the units of meaning in the sentence (Allen 1995; Elaine and Knight, 1991; Norvig and Russell, 1995; Molina, 2002; Fernando, 2002). More specifically, in the context of this study it means constructing a parse tree that captures the grammatical content of an input sentence.

When an input string (or tokens) is passed to the parser token by token, the parser calls the morphological analyzer for each token, which segments words into roots and affixes according to the morphological rules of Amharic. Based on the categories for each word that the part of speech tagger outputs, a parse tree is constructed. The parse tree is nothing but a diagrammatic representation of the input sentence and it records the rules of the language and how they are matched. Every node of a parse tree corresponds either to an input word or to a non-terminal in the grammar (Allen 1995; Charniak 2001). Each level in the parse tree corresponds to the application of one grammatical rule. However, the final terminal symbols are connected to the input word related to its lexical category.

In the following sentence:

nägadeu gäbäreu yäshät'älätn ∂h∂l wädä kätäma amät'a #

'The merchant brought the food that the farmer sold him to the city.'

The parsing process may then take in the sentence and produce the output given below.

S (NP (N **nägadeu**)

VP(S'(S (NP (N **gäbäreu**) VP (V **shät'älätn**)) (COMP **yä**-)

VP (NP (N **∂h∂l**) VP (PP (P **wädä**) (N **kätäma**) VP (V **amät'a**))))) #

The symbols like S, N, NP, V, VP, P, PP and COMP are used here to represent sentence, noun, noun phrase, verb, verb phrase, preposition, prepositional phrase, and relativizer, respectively. There are two ways in which the process of parsing sentences of such kind can be carried out: manual and automatic. As the volume of text to be parsed gets higher and higher, the manual process becomes tiresome, prone to error and expensive. On the other hand, automatic sentence parsing avoids such complexities and plays important roles in natural language understanding systems. Currently there exist several different approaches to the task of sentence parsing, which are broadly grouped into theoretically-based approaches and statistical approaches and this will be discussed in the following section.

2.3 Approaches To Automatic Sentence Parsing

In the current literature on NLP in general, and on automatic sentence parsing in particular, a distinction is often made among the various syntax analysis methods. Based on their reasoning mechanism, the attempts already made and the techniques discovered

are roughly classified into two categories: *rule-based* versus *stochastic* approaches (Atelach, 2002).

2.3.1 Rule-Based Approach To Automatic Sentence Parsing

The Rule-based approach to automatic sentence parsing is entirely based on the information from the knowledge base and some kind of learning technique (if any) to handle ambiguity and guess unknown words. As Mao (1997) reports, such a system learns a set of rules automatically based on a given list of tokens (strings) and then parses sentences following these rules. In this approach, the task of parsing can be carried out in two different ways, namely: Top-down and Bottom-up parsing techniques (Elaine and Knight 1991).

2.3.2 Stochastic Approach to Automatic Sentence Parsing

The stochastic approach, also called corpus-based approach, is based on the ideas of Bayes (Network) theorem, independent events and the Markov assumption in sentence parsing. The approach uses these concepts to determine the most likely lexical sequence of each word in a given sentence. Based on the type of text corpora used, the corpus-based approach can be further categorized into *supervised* and *unsupervised* approaches. Supervised approaches use annotated text corpora while unsupervised approaches use natural corpus as those found in newspaper and books.

Automatic syntactic analysis systems developed following the supervised approach are called supervised parsers, and they use probability (i.e. statistics) in analyzing the problem of parsing. There are two important information sources in a supervised parser: the lexicon, which lists each word with the entire possible lexical category for each word

with its respective estimate of lexical probabilities[3], and the list of contextual probabilities for each lexical category. The list of contextual probabilities indicates the particular lexical category that is appropriate for a particular context (See Allen, 1995). The major problems in developing supervised parsers include lack of manually or automatically parsed text (corpora) and the fact that it needs to manual parsing each time the parser is applied to a new text (Brill 1996). Manual parsing is really expensive and time consuming, but if pre-tagged corpora are easily available, the Hidden Markov Model (HMM) approach in particular and stochastic parsers in general can be adopted for new languages with little effort.

Parsers developed using unsupervised stochastic techniques do not require any pre-tagged text in the training process. Some heuristics or probabilistic information generated from the corpus is used to develop the syntactic analysis system (Kazakov and Munandhar, 2000). These parsers are similar to their supervised counterparts in that both assume the HMM assumption. HMM is a set of states (lexical categories in this case) with directed edges labelled with transition probabilities that indicate the probability of moving to the state at the end of the directed edge, given that one is now in the state at the start of the edge. The states are also labelled with a function which indicates the probabilities of outputting different symbols if in that state (while in a state, one outputs a single symbol before moving to the next state). In this case, the symbol output from a state/lexical category is a word belonging to that lexical category (Wilson, 2000). Besides, both types of parsers use a large dictionary to list all possible lexical categories for each entry, and

[3] Lexical probability is the probability of a word to have a particular label and is usually denoted by p([word]lexical category).

also use statistical, rule-based or hybrid techniques to achieve disambiguation, etc. However, the unsupervised stochastic parser has such unique features as training takes place on an unparsed or fresh text, uses the Baum-Welch algorithm (which is different from the Viterbi algorithm), and so on.

2.3.3 Parsing Strategies

Various strategies have been proposed to address such questions concerning to parsing: '*Where do we start from?*', '*In what order is the string or the Right Hand Side (RHS) of a rule looked at?*' and '*How are alternatives explored?*' Researchers in the area of NLP provided strategies like top-down, bottom-up, left-to-right, right-to-left, depth-first, breadth-first, and chart parsing as solutions. The following consecutive subsections discuss some of the major solutions provided at various times.

2.3.3.1 Top-down Vs Bottom-up Parsing

Top-down and bottom-up are rival solutions that have been proposed as alternative solutions for the strategy question regarding the direction of the parsing process. **Top-down parsing** begins with the start symbol (usually a sentence S) and applies the grammar rules forward until the symbols at the terminals of the tree correspond to the components of the sentence being parsed. For example, the parser starts in state (S), after applying the rule S → NP VP, the symbol list will be (NP VP). It then applies the rule NP → ART N and the symbol list will be (ART N VP), and so on. The parser could recursively continue in this fashion until it arrives entirely at terminal symbol states, and then it could check the input sentence to see if the classes of words in it matched with the

rewritten sequence of terminal symbols (Allen 1995). Parsers developed using this approach are called top-down parsers. This parser makes a hypothesis about what it is looking for and to decide its next move. Hence, top-down parser is characterized by a sequence of goals to determine the remaining words. Earley (1970) developed a similar parsing algorithm that often called the Earley algorithm.

On the other hand, **bottom-up parsing** begins with the sentence to be parsed and applies the grammar rules backward until a single tree whose terminals are the words of the sentence and whose top node is the start symbol (usually S, for sentence) has been produced (Elaine and Knight 1991: 388). That is, it starts from each word and assign its grammatical category until it reaches the start symbol. This operation is repeated, at each state, using the sequence of highest-level labels as the new string to operate on. The task of the parser would now appear to be that of attempting to group words into their respective categories together (e.g. take a sequence ART ADJ N and identify it as an NP) in a manner permitted by the grammar.

Top-down methods have the advantage of being highly predictive. This means that a word might be ambiguous when considered in isolation, but if some of those grammatical categories cannot be used in a legal sentence, then these categories may never even be considered (Allen, 1995).

However, this method has a reduplication of effort that becomes a serious problem, and large constituents may be rebuilt again and again as they are used in different rules. In contrast, the bottom-up parser only checks the input sentence once, and only builds each

constituent exactly once. The basic observation to make about the bottom-up parser is that it works from left to right: it does everything it can with the first item before exploring what it can do with the next two items, and so on. That is, the parser is bottom up, driven entirely by the data presented to it and building successive layers of syntactic abstraction on the basis of data provided.

Unfortunately, Allen (1995) states whether one chooses top-down or bottom-up to implement, the payback is prohibitively expensive, as the parser would tend to try the same matches again and again, duplicating much of its work unnecessarily. Hence, to avoid such reduplication problem there should be a mechanism that allows parser to store results of the matching it has done so far. Such technique is called chart-based parsing. Therefore, the combination both strategies can result in a better parser. A small variation in the bottom-up chart algorithm yields a technique that is predictive like the top-down approaches, yet avoids any reduplication of work as in the bottom-up approaches.

2.3.3.2 Left-to-right Vs Right-to-left

These are the competing solutions that have been forwarded for the question regarding the order of looking at substrings or an RHS. **Left-to-right parsing** processes the words of the sentence from left to right (i.e. from beginning to end), as opposed to right-to-left (i.e. end-to-beginning) parsing. That is, it uses the leftmost symbol first, continuing with the next to its right. Logically, it may not matter which direction parsing proceeds in, and the parser will work, eventually, in either direction (Wilson, 2000). However, right-to-left parsing is likely to be less intuitive than left-to-right. But there are situations in which using both strategies becomes useful. If the sentence is damaged, for example, by the

presence of a misspelled word, it may help to use a parsing algorithm that incorporates both left-to-right and right-to-left strategies, to allow one to parse material to the right of the error.

The top-down strategy encounters trouble with rules that exhibit *left recursion* when it is performed from left to right (Pritchett 1992; Marcus 1983). Left recursion arises when the first category on the RHS of a rule is more general than the one on the LHS (Left Hand Side). In this case, it is possible to transform a left-recursive grammar into an equivalent one with no left-recursive rules, and one that generates exactly the same set of strings (although it will not assign the same structures). However, the bottom-up never makes such thing and it only checks rules to see if there is a legitimate way of putting together the words already in hand (Allen 1995; Merlo 1996; shieber et al 1995). Such a parser encounters problem with empty grammar rules.

2.3.3.3 Depth-first Vs Breadth-first

As far as the question of exploring parse space (i.e., alternative parses) of a given input sentence is concerned, there are two contending solutions, depth-first and breadth-first. In the former case a single alternative parse is completely pursued at every choice point before trying another alternative. State of affairs at the choice points needs to be remembered and choices can be discarded after unsuccessful exploration. Breadth-first, on the other hand, pursues every alternative parse at every choice point for one step at a time. It requires massive bookkeeping since each alternative computation needs to be remembered at the same time. While depth-first search is generally incomplete, breadth-first search is guaranteed to be complete.

2.3.3.4 Chart Parsing

The chart is a data structure for representing partial results of parsing process in such a way that they can be reused later on. The chart for an n-word sentence consists of n+1 vertices and a number of edges that connects vertices. A chart parser is a variety of parsing algorithm that maintains a table of well-formed substrings found so far in the sentence being parsed. While the chart techniques can be incorporated into a range of parsing algorithms, mostly they have been used in the context of a particular bottom-up parsing algorithm (Wilson, 2000).

The main idea behind this technique is that the essence of improving parsing efficiency. As indicated by Norvig and Russell (1995), there are three points to keep in mind for efficiency consideration of chart parsing: it is advisable not to do twice what can be done once, not to do once what can be avoided altogether and not to represent distinctions if that is not the concern of the study.

Chart parsers maintain the records of all the constituents derived from the sentence so far in the parse. That is, it stores the intermediate results and maintains the record of rules that have matched but are not completed. To be more specific, for example, once it is discovered that ***leboch yägädälut yäbuna nägade askären*** 'The body of the coffee merchant that was killed by thieves' is a complex NP as it is used in the sentence ***leboch yägädälut yäbuna nägade askären wädä hospital tälakä***, 'The body of the coffee merchant that was killed by thieves was sent to hospital', it is recommended to record that results in a data structure known as a chart. Recording of intermediate results

is a form of dynamic programming that avoids duplicate work (Norvig and Russell 1995; Allen 1995).

It can be observed from the ongoing scenarios that, chart-based techniques use a combination of top-down and bottom-up processing in a way that means it never has to consider certain constituents that couldn't lead to a complete parse. (This also means it can handle grammars with both left-recursive rules and rules with empty Right Hand Sides with out going into an infinite loop). The result of the algorithm is a 'packed forest' of parse tree with its constituents labeled with their respective grammatical categories in the trees.

The basic operation of a chart-based parser involves combining an active arc (also called edge) with a completed constituent. The result is either a new completed constituent or a new arc that is an extension of the original active arc. New complete constituents are maintained on a list called ***agenda*** until they themselves are added to the chart. For example, [0, 5, S→ NP VP•] means that an NP followed by a VP combine to make an S that spans the string from 0 to 5. The numbers here indicate the indexing of the grammar rules. The symbol • in an edge separates what has been found so far from what remains to be found. The numbers before the symbol S indicates the indexing of the grammatical categories. Edges with • at the end are called complete edges. The edge [0, 2, S→ NP • VP] says that an NP spans from 0 to 2, and if the parser could find a VP to follow it, then the parser would have an S. Edges like this with the dot before the end are called active arcs.

Chart-based parsers are more efficient than those that rely on a search since the same constituent is never constructed more than once. A pure top-down or bottom-up search strategy could require up to C^n operations to parse a sentence of length n, where C is a constant that depends on a specific algorithm used. Even if C is very small, this exponential complexity rapidly makes the algorithm unusable (Allen, 1995).

However, it is reported that chart-based parser would require $K * N^2$ time and space complexity to build each constituent as lexical categories between every positions, where N is the length of the sentence and K is a constant depending on the algorithm. Thus, it reduces parsing operations substantially. In chart parsing, the process of parsing an *n-word* sentence consists of forming a chart with *n+1* vertices and adding edges to the chart one at a time, trying to produce a complete edge that spans from vertex 0 to *n* and is of category *S*. There is no backtracking; everything that is put in the chart stays there.

In general, there are two different issues that are of concern. The first involves improving the efficiency of parsing techniques by reducing the search but not changing the final outcome, and the second involves techniques for choosing between different interpretations that parser might be able to find. To accomplish this, the following technique is usually employed.

It was noted earlier that the bottom up failed to store any intermediate results. That is the key reason for its obsessively time-wasting activity in rechecking things that it has already checked and which cannot be changed. This can be considered as amnesiac! It checks each *new* category to see if it exhausts an RHS and check each *new* adjacent pair

of categories to see if they exhaust an RHS. This makes it slightly harder to use, since the implementation now has to make it impossible to keep a record of which categories a parser is in (Allen 1995; Norvig and Russell 1995; Marcus 1983).

This issue of storage of intermediate results is independent of the distinctions already discussed, even though any parser needs to store some states, simply to remember what it is currently doing; in particular, chart parser has to remember the multiple hypothesis states that are currently being entertained. Storage of intermediate results also turns out to be the key to efficient parsing. Chart-based parsers use a data structure called a chart to encode all intermediate results obtained in the course of a parse (Schildet 1987; Allen 1995; Norvig and Russell 1995; Shieber et al 1995).

One way to improve the efficiency of parsers is to use techniques that encode uncertainty, so that the parser need not make an arbitrary choice and later backtrack. Rather, the uncertainty is passed forward through the parse to the point where the input eliminates all but one of the possibilities. The efficiency of the technique described here arises from the fact that all the possibilities are considered in advance, and the information is stored in a table that controls the parser, resulting in parsing techniques that can be much faster.

It can be observed that chart parsers are more efficient than any other parsers. They encode intermediate results so that reduplication of effort is avoided. Moreover, chart parser encodes uncertainty to avoid ambiguity and predicts the word category of unknown words (those that are not in the knowledge base). Therefore, the present study

will implement chart parser suggested by Allen (1995) to avoid uncertainty and predict unknown words' category.

2.4 Knowledge Required By The Parser

In order to write efficient parsing algorithms, one needs to have a secure definition of the language to which he or she designs the parser. Much of the knowledge for NLP purpose must come from linguistic studies. This helps us understand better the organization and operation of the language. Knowledge of the grammar of the language also helps to have an idea on how the system should behave under a wide range of conditions. For this reason NLP systems will have to be based on what linguists can determine about the structure the language (Gazder and Pullum, 1985; Spark Jones, 1974).

The syntactic analysis system to be developed involves a knowledge base to guide the parser. The main components of the system to be built include a lexicon, a list of grammatical rules, a morphological analyzer, a POS tagger. While the first two of these knowledge base components of the parser are discussed here, the last three are discussed in another section of this chapter.

2.4.1 The Lexicon

The term *lexicon*, which is a linguistic term for dictionary, is a collection of information about the words of a language and the lexical categories to which they belong. It usually serves as the starting point of any NLP system and must contain information about every potential word form which the system might come across, in order to guide processing.

A lexicon is usually structured as a collection of ***lexical entries***, like ("pig" N V ADJ). "Pig" is familiar as a noun, but also occurs as a verb ("Jane pigged herself on pizza") and as an adjective as in "pig iron". In practice, a lexical entry will include additional information about the roles the word plays, such as feature information - for example, whether a verb is transitive, intransitive, bi-transitive, etc., what form the verb takes such as present participle, or past tense, etc. (Wilson, 2000).

According to Allen (1995), a grammar need not contain any lexical rules of the form, for example, N➔*flower*, so long as a lexicon is specified. The following simple context free grammar lexicon for Amharic is shown by Atelach (2002).

N ➔ **säw**
V ➔ **mättä**
Adj ➔ **t?I?k**
Adv ➔**tolo**

In this example the symbols on the left hand sides are POS categories for the words on the right hand sides.

2.4.2 The Grammar Rule

Grammar is a formal system for describing the rules and syntax allowable in the language. The most common way to represent grammars is as a set of grammar rules that capture generalizations to classify words into what are often called 'parts of speech' or grammatical categories. The following is an example of simple grammar for the sentence **Kassa wädä t?m?h?r?t bet hedä,** 'Kassa went to school.'

1. S ➔NP VP
2. VP ➔PP VP
3. NP ➔NAME
4. PP ➔P N
5. VP ➔V

5. NAME➔**Kassa**
6. P ➔ ***wädä***
7. N ➔ ***təməhərət, bet***
8. V ➔ ***hedä***

Grammar rules underlie many linguistic theories, which in turn provide the bases for many natural language understanding systems (Flickinger, 2002). The grammar of Amharic is compiled into a Left to Right (LR) table.

There are different ways of specifying grammars often called *grammatical formalisms*. The most commonly and widely used formalisms include: Context Free Grammar (Chomsky, 1957), Transitional Grammar (Chomsky, 1965; Radford, 1981), Transition Network Grammars (Woods, 1970), Unification Based Grammar (Kay, 1982) and Probabilistic Context Free Grammars (Charnaik, 1993). Hence, the grammar rules will take on different forms depending on the theoretical framework of the specific grammar.

2.4.2.1 Context Free Grammars

Grammars consisting entirely of rules with a single symbol on the left-hand side are called *context-free grammar* (CFG). A CFG is a formal system that describes a language by specifying how any legal text can be derived from a distinguished symbol called *axiom*, or *sentence symbol*. CFG is required to be monotonic and the only way a CFG rule can be non-monotonic is by having an empty right-hand side. CFGs are very important for two main reasons: the formalism is powerful enough to describe most of the

structure in natural languages, yet it is restricted enough so that efficient parsers can be built to analyze sentences (Allen, 1995).

This formalism consists of a set of *productions*, each of which states that a given symbol can be replaced by a given sequence of symbols. S ➔NP VP, for instance, is one of such productions stating that S can be replaced by the sequence of NP and VP. NP and VP, in turn, have sequences of symbols that replace each (for example, NP ➔Adj N and VP ➔V NP). Symbols that are to be replaced are called *non-terminals*, and are always represented by *identifiers,* which are sequences of letters and digits. Every non-terminal must appear before a colon in at least one production. The axiom is a non-terminal that appears before the colon in exactly one production, and does not appear between the colon and the period in any production. There must be exactly one non-terminal satisfying the conditions for the axiom. Symbols that cannot be replaced are called *terminals*, and may be represented by either identifiers or *literals* (which are a sequence of characters bounded by apostrophes).

2.4.2.2 Transition Network Grammars

The Transition Network Grammars is another formalism that is useful in a wide range of applications. It is based on the notion of a transition network consisting of nodes and labeled arcs. Simple transition networks are often called finite state machines (FSMs) and are equivalent in expressive power to regular grammars, and thus are not powerful enough to describe all languages that can be described by a CFG (Allen, 1995). The notion of recursion in the network grammar is needed in order to get the descriptive power of CFGs. Thus, the grammatical formalism that is based on such a notion is called

Recursive Transition Network Grammars (RTNG). The RTNGs consist of nodes and labeled arcs where one of the nodes is specified as the initial state, or start state.

Woods (1970) also introduced another widely used grammatical formalism called Augmented Transition Network (ATN). This formalism has become one of the most popular forms for writing natural language grammars (Grishman, 1994). A transition network is a representation of regular (or finite-state) grammar. The network is a directed graph whose arcs are labeled by terminal symbols (words or word categories). One node of the graph is designated as the start state; one or more nodes are marked as final states. The assumption is that if there is a path from the start state to some final state such that the labels on the arcs of the path match the words of the sentence, a sentence is in the language defined by the network.

2.4.2.3 Context Sensitive Grammars

Still another commonly used grammatical formalism is the Context Sensitive Grammar, which is a phrase structure grammar with context-sensitive rules. There are two equivalent formulations of the definition of a context-sensitive grammar rule:

- Rules of the form x -> y where x and y are strings of alphabet symbols, with the restriction that length (x) <= length of (y).
- Rules of the form A -> y / x__z where A is a non-terminal symbol, y is a sequence of one or more terminal and non-terminal symbols, and x and z are sequences of zero or more terminal and non-terminal symbols.

The meaning of the latter rule (or production) is that A can be rewritten as y if it appears in the context 'x__z', i.e. immediately preceded by the symbols x and immediately

followed by the symbols z (Grishman, 1994). Context-sensitive grammars are more powerful than CFGs though the former kinds of grammars are much harder to work with than the latter (Wilson, 2000).

2.4.2.4 Unification-based Grammars

The grammar formalism that makes extensive use of feature structures (such as case, gender, tense) including their values represented in the lexical entries of words is called *unification-based grammars*. These feature structures are manipulated by the operation of unification (i.e. the entire grammar can be specified as a set of constraints between feature structures). CFGs or any of the grammar formalisms discussed above can serve as the backbones for the unification-based grammars, CFGs being the most common. Joshi (NY), as cited in Atelach (2002) indicated, the main reason for the excessive power of the unification-based grammars is that recursion can be encoded in the feature structures.

2.4.2.5 Probabilistic Context Free Grammars (PCFG)

Just as Finite State Machines could be generalized to the probabilistic case, CFGs can also be generalized. This can be achieved through having some statistics on rule use. That is, simply by counting the number of times each rule is used in a corpus contatinng parsed sentences and by using this statistical information to estimate the probability of each rule being used. Given a category C and m grammar rules R_1, ..., R_m with left-hand side C, the probability of using rule R_j to derive C can be estimated by

$$\mathbf{Pr(R_j \mid C) = count(\#times\ R_j\ used) / Sum\ i=1,m\ (\#times\ R_i\ used) \ldots. (1)}$$

Such a context free grammars with associated probabilities is called Probabilistic Context Free Grammars (PCFG) formalism. A typical PCFG grammar that is based on a parsed version of a given corpus, therefore, comprises a count for LHS, a count for rule, and probability figures for each of the rules generated. A probabilistic context-free grammar (PCFG) is, therefore, commonly defined as a four-tuple $\langle W, N, S, R \rangle$, where $W = \{w^1, w^2, ..., w^u\}$ is a set of terminal symbols like words in a sentence, $N = \{N^1, N^2, ..., N^v\}$ is a set of non-terminal symbols, $S = \{N^1\}$ is a set that only has one starting symbol, and $R = \{R^1, R^2, ..., R^w\}$ is a set of grammar rules with probabilities. For a rule $R^m \in R$, it is a context-free grammar (CFG) rule with the form $R^m : N^i \rightarrow \xi^j$, and its probability $P(R^m) = P(N^i \rightarrow \xi^j)$(Yao and Lua, 1998).

PCFG implementation techniques involve making certain independence assumption about rule use. In particular, it is assumed that the probability of a constituent being derived by a rule $\mathbf{R_j}$ is independent of how the constituent is used as a sub constituent. With this assumption, a formalism can be developed based on the probability that a constituent C generates a sequence of words W_i, W_{i+1}, …, w_j, written as $w_{ij,}$ where i and j refer to the position of the word in the sentence.

Although they do not take context or lexical co-occurrence into consideration, PCFGs are especially useful for sentence parsing as it allows handling such cases as structural ambugity, ungrammatical sentence analysis, and grammar learning, even compared with CFGs. It gives a better probabilistic model for syntax analysis for statistical properties of natural languages are introduced (Yao and Lua, 1998). In this study, the PCFGs

grammars formalism is used in order to describe and represent Amharic words into their allowable grammatical and syntactic categories. The remaining section of this chapter discusses related NLP systems that can function as essential component systems to develop automatic sentence parsing

2.5 Related NLP Components Systems

2.5.1 Morphological Analyzer

The first step in any NLP task involves recognizing and identifying individual word forms from the input text (Salton, 1989). In some languages such as English, such information may be supplied by a lexicon that simply lists all word forms with their part-of-speech and inflectional information such as number and tense. Since such languages have a relatively simple inflectional system, the number of forms that must be listed in such a lexicon is manageable. But an exhaustive lexical listing is simply not feasible for many other highly inflectional languages such as Amharic that have a number of inflected forms for each noun or verb. This is because each lexical word may have literally thousands of distinct surface forms differing in inflectional properties but otherwise the same vocabulary elements (Antworth, 1994). Therefore, NLP for these languages could be practical only if there is a morphological analyzer that will use the morphological information of the language to compute the parts-of-speech (POS) and inflectional categories of words (Antworth, 1994; Kuritutnun, 1993).

Hence, a morphological analyzer is a central element needed to parse words into their morphemic components, and also to give word classes (such as noun, verb, etc.) to which

a particular word may fall before the task of parsing is carried out. It involves rules that are needed to provide information useful to treat words that are not in the lexicon of the parser. In other words, such rules are useful to make reasonable guesses as to the grammatical categories of unknown words. On top of this, a morphological analyzer would help in assisting a parts-of-speech tagger (POST), which is a major component of a syntactic parsing system. A justification for the incorporation of a morphological analyzer into this study is provided in the next section where this second essential component for a sentence parser is discussed.

In this regard, a prototype morphological analyzer for Amharic was developed by Tesfaye (2002). He used a language independent system, called Linguistica 2001, to create morphological dictionary (known as signature) by extracting prefixes, stems and suffixes from a given corpus. Regarding the experimental result he obtained, the prototype parses 87% of words of the test data (433 or 500) successfully. The assumption in this study is that the results obtained from the prototype morphological analyzer for Amharic developed by Tesfaye (2002) would play a significant role in preprocessing the input text to the parser together with the other NLP component system, the POS tagger that is to be discussed in the following section.The preceding justifications for incorporating a morphological analyzing system into an automatic sentence parser, the intention of this study was to use the morphological analyzer as a component that assists another NLP system for Amharic, a parts of speech (POS) tagger developed by Mesfin (2001).

2.5.2 Part-Of-Speech Tagger

As it was indicated in the previous section, the other major and essential component of a sentence analyzing system is a POS tagger, an NLP system that automatically assigns the possible parts-of-speech categories to a given word in a sentence. Elimination of unlikely parses (wrong analyses of a sentence) is one of the principal motivations of incorporating POST into a given automatic sentence parser since a POST involves identifying the syntactic categories of words in a text. That is, a given sentence, for example, **ǝne kassa yägäzaun näč bäg tǝlant ayahut** 'Yesterdat I saw the white sheep that Kassa bought' would become unambiguous if we can get the correct POS tag assignment as follows:

ǝne\N **kassa**\N **yägäzaun**\V **näč**\Adj **bäg**\N **tǝlant**\Adv **ayahut**\V.

Also, a single morphological form may very likely occur with various syntactic categories and some mechanism is necessary to identify the correct syntactic category of the word in the given position in a sentence. Amharic has many multi-concept words such. The word **Bäqälä** is a popular proper name in the language and, therefore, belongs to the class of nouns where as the same word **bäqälä** 'It grew' is used as a verb in the language, and , therefore, belongs to the verbs category. Hence, it is a parts-of-speech tagger that resolves the appropriate syntactic category of such a word. In this regard, Mesfin (2001) developed a prototype simple automatic part of speech tagger for Amharic. The experiment was conducted on an Amharic text composed of 261 words and the results achieved were 97% and 90% accuracies on the training and test sets, respectively. This tagger has been incorporated in this study and the modifications that were made on the contents of the original database are discussed in chapter four.

Mesfin's POS tagger identified some special like Vprep and Nprep which contain words (such as **bamakina** 'By car' **salahagar** 'About a country) in each of which the prefixed preposition is not separated from the corresponding noun. However, Baye (1987), from which the parse structures adopted in this study are taken, treats these kinds of words as prepositional phrases that are composed of a preposition and a noun. Therefore, without the involvement of a morphological analyzer (during parsing) such words would be left as phrases (i.e. as prepositional phrases) at a terminal position, which is not the desirable syntactic parsing outcome. Hence, the use of a morphological analyzer in this study is really essential.

In addition to incorporating the concepts on which the above two NLP prototype systems for Amharic are developed into it, this study used thee techniques applied to develop the automatic sentence parser for Amharic developed by Atelach (2002). The present study, therefore, can be considered as a continuation of the previous simple sentence parsing system for Amharic and also as an endeavor to add an effort towards integrating basic concepts of previous NLP works for Amharic.

2.6 Conclusion

In summary, this chapter defined automatic sentence parsing as a procedure that explores various ways of combining grammatical rules to find a combination that generates a tree that could represent the structure of the input sentence. The major approaches and strategies that were proposed at various times were also reviewed. The knowledge base of a sentence parser and the necessary components such as the lexicon, the grammar rules

and the grammatical formalisms that are need to store information that helps during the parsing process were also described in this chapter. Finally, NLP systems that could be essential components of a sentence parser were also discussed.

CHAPTER THREE

THE STRUCTURE OF AMHARIC

3.1. Introduction

This chapter discusses the structure of Amharic word classes, phrases and sentences since each of these units has its own impact on the present study. The Amharic word classes, i.e. nouns, verbs, adjectives, adverbs, prepositions and conjunctions (Mersi'hazen, 1934) are also discussed in this chapter. Pronouns are treated under the noun category, and other word categories such as interjections and numerals are also topics discussed in this chapter. The various phrase structures of the Amharic language such as noun phrases, verb phrases, adjectival phrases, adverbial phrases and prepositional phrases as classified by Getahun (1990), and sentence formalisms of the language, more importantly the features of Amharic complex sentences, are all discussed in this chapter. To the better understanding of the material in this section, the chapter begins with a brief review of the Amharic writing system and punctuation marks.

The analysis and discussions made in this chapter are based on the data extracted from Atelach (2002), Abiyot (2000), Baye (1987), Getahun (1990), Hirut (1998), Dawkins (1969), Mesfin (2001), and Tesfaye (2002). Further details on issues raised in this chapter can be obtained from these sources.

3.2. The Amharic Writing System

As cited by Lo (no year), the Blackwell Encyclopaedia of Writing Systems defines the term *writing system* as "a set of visible or tactile signs used to represent units of a

language in a systematic way". The present Amharic writing system was adopted from the Ge'ez writing system. Ge'ez, which belongs to the class of Semitic language family, was the language of literature in Ethiopia in earlier times (Bender et al., 1976). Thus, the Amharic writing system consists of a core of thirty-three characters (called '*fidel'*) each of which occurs in one basic form and in six other forms that may be described as *orders*. These seven orders (the first basic order and the other six orders) represent the different sounds of a consonant-vowel combination (a characterization known as syllabic). The non-basic forms are derived from the basic ones by somewhat regular modifications (Bender et al., 1976; Hudson, 2001). The 33 core characters then yield 231 distinct symbols. In addition to the 231 characters, there are others that contain special features usually representing labialization like *kwe, gwe,* etc. (Ibid.).

Analysis of Amharic texts reveals that different Amharic punctuation marks are used in Amharic to serve for different purposes. Beletu (1982) as cited in Abiyot (2000), indicated that there are about 17 punctuation marks of which only a few of them are commonly used and have representations in Amharic software. In this study, the Latin letters are used to represent the Amharic alphabet assuming that it is possible to develop a program that converts Amharic texts, written using the Latin letters, to the equivalent Amharic *fidel*. Detailed discussion on Amharic writing systems is found in (Abiyot, 2000).

3.3 Word Categories In Amharic

One distinguishing feature of Amharic from some languages such as English is that the basic word order of Amharic follows the syntactic criteria: S V C. the linguistic

characteristics of the language have been studied by different researchers. To start with, based of the way they categorize the Amharic words, research works in the area of Amharic word categorization can be categorized into two groups, namely, *early* and *recent* works Mesfin (2001). In the early works such as Mersi'hazen (1934), Amharic words are categorized into the following eight classes (or parts of speech). These are the *noun, Verb, Adjective, Adverb, Preposition, Pronoun, Conjunction* and *Interjection* classes. In recent works like Baye (1987), however, the early classification of Amharic words is reduced into five putting *pronouns* and *conjunctions* under the *noun* and *preposition* categories, respectively. Here *interjections***,** which are words with out syntactic functions, are not considered as parts of speech at all.

Following the recent NLP works for Amharic(e.g. Atelach, 2002; Mesfin, 2001), this study adopts the classification by the early scholars but treating nouns and pronouns in the same part of speech class as suggested by Baye (1987). While justifying the adoption of such a classification into his work, Mesfin (2001) indicated that the early categorization is more exhaustive and could allow his POS tagger to tag words exhaustively and easily.

3.3.1 The Amharic Noun Class

Amharic nouns are words used to name or identify of a class of things, people, places, ideas, etc. or any subset (member) of these classes. For the purpose of this study, this class comprises proper nouns, nouns that take the suffix /*-oč*/ as a plural marker and /-n/ as accusative case marker, and pronouns like **ɔne** 'I', **antä** 'you', and so on. Amharic

nouns can be either primitive or derived, where nouns or words in general are said to be in their primitive forms if they exist in their original form (e.g **gwadäña** 'a friend) whereas they are referred to as derived if they originated (derived) into their present state from a different, and possibly completely different categories (e.g. **gwadäñ∂nät** 'friendliness').

3.3.2 The Amharic verb class

The Amharic Verbs are those words that usually come at the end of a complete grammatical declarative sentence. They also take suffixes (i.e. subject markers) like /-hu/ 'I', /-h/ 'You', /-č/ 'She' and so on, to agree with the subject of the sentence.

3.3.3 The Adjective Class

These are words that usually come before nouns and serve as modifiers of nouns they precede. In the sentence

∂su gobäz l∂j näw

'He clever boy is'

'He is a very clever boy.'

for instance, the adjective gobäz 'clever' is an adjective, and it modifies the noun *l∂j* 'boy'. Like nouns Amharic adjectives can be either primitive or derived (for the definition of these terms, see section 3.3.1).

3.3.4 Prepositions in Amharic

The term 'preposition' refers to those words that will have meanings only when they are attached or used together with other words such as nouns, verbs, pronouns and adjectives. The prepositions category in Amharic comprises affixes such as (as they are also roughly

glossed in English) **/lä/** 'for', **/kä/** 'from', **/bä/** 'with', **/yä/**'that' or 'of', **/slä/** 'for' and so on. They are used to form adverbial phrases appearing before nouns.

Amharic prepositions can take varied forms. First, in the phrase ***səlä təmhərt*** 'About education', the preposition ***səlä*** appears as a simple preposition that stands alone as a separate word. Secondly, a preposition can appear as one that is prefixed to other words (e.g. nouns and verbs). For example, in the phrase **bä-mäkina** 'By car', the preposition **bä** is prefixed to the noun **mäkina** 'car'. When they are prefixed to verbs (e.g. **səläwädäqä** 'Since he fell down', **yä-säräqäu** 'that stole'), they form different types of subordinate clauses that are typical components of complex sentences. The roles that prepositions play in Amharic complex sentence formation will be discussed in section 3.6.

Thirdly, prepositions can appear as compound prepositions consisting of two parts: prepositional prefixes and post positions with a noun at the middle. For instance, in the phrase **Kä-wändimu gar** 'with his brother', **Kä** 'from'is used as a prepositional prefix whereas **gar** 'with' is an independent preposition that usually follows nouns. Detailed discussion about compound preposition is found in Baye, (1986).

3.3.5 The Adverb Class in Amharic

In Amharic, adverbs that are found in their primitive form are very few and the most common include: **gäna** 'yet', **tolo** 'quickly', **kəfuňa** 'severely', and **mənəňa** 'foolish' (Getahun, 1990). Thus, phrases that are combinations of prepositions and some other words like nouns (e.g. **bä-hayəl** 'By force') or prepositions and adjectives (e.g. **bä-dähna** 'well') perform the functions of adverbs in the language. Adverbs refer to places,

time, circumstances etc. of the action mentioned by the verb. Besides, there are adverbs of position, also called to as *noun adverbs* that function either as a *noun* or as an *adverb* depending on the context they are used in. for example, in **wädä wəsť gäbä** 'He entered inside', the word **wəsť** is used as a *noun* while in **bä-wəsť yäsäral '**He works in' '**wəsť** is used as an *adverb* (Atelach, 2002).

3.3.6 Conjunctions in Amharic

This class of words includes both coordinating and subordinating conjunctions. Conjunctions are used to synchronize words, phrases, clauses and sentences. **ənа** 'And', and **wäynəm** 'Or' are examples of Amharic conjunctions and a list of Amharic conjunctions followed by a detailed discussion on them is found in (Dawkins, 1969).

As it has been pointed so far at the beginning of this chapter, this study adopts the early method of Amharic POS classification in which *conjunctions* and *prepositions* are treated as independent POS categories. However, one problem with this trend of classification is the fact that sometimes the same forms (e.g. **səlä**, **bä** and **kä**) forms are used both as prepositions and conjunctions. This categorical ambiguity of such words can easily be solved by making careful analysis of the context in which the words are used.

3.3.7 Numerals

This class of words represents numbers, i.e. both **cardinal** and **ordinal** numbers. A list of the Amharic cardinal numbers is found in (Dawkins, 1969; Leslau, 1973). In Amharic, ordinal numbers are formed from their cardinal number equivalents and the suffix /-**äňa/.**

Example:	Cardinal	Gloss	Ordinal	Gloss
	hulät	two	**hulätäňa**	second

Like English, compound Amharic numerals are put separately. The following are examples to illustrate this.

Example: **hulät mäto sälasa and** 'two hundred thirty one'

hulät mäto sälasa andäňa 'two hundred thirty first'

There are also numerals to indicate distribution and also partitions of something whole.

Example of distributive numerals**:** **hulätt hulatt** 'two two'

Examples of partition numerals: **gəmaš** 'half'; **siso** 'one third'; **rub** 'quarter'

3.3.8 Interjections

Like English, Amharic has many words or phrases that are used to express emotions like sudden surprise, pleasure, annoyance and so on. Words that are used for such purposes are called *interjections*. For example, words like **goš !** 'Good job!' **wəy** 'ouch' and so on are common interjections in the language. They can stand alone or can appear any where in a sentence. A long list of Amharic interjections is found in (Dawkins, 1969).

Generally speaking, Amharic words may be classified under one of the aforementioned word classes. Nevertheless, knowledge of the different word classes is not enough for the task of categorizing words of a given sentence into their proper classes. Moreover, some Amharic words (like **abbäbä** 'a proper name' or 'blossomed', **bäkälä** 'a proper name or 'it grew', and so on) can be categorized in more than one word category (e.g. as noun and as verbs) depending on the context they are used. Recall also the case of some of the prefixes like **səlä**, **bä**, **kä** etc. that can be categorized as both conjunctions and

prepositions (see section 3.3.6). Hence, there should be a mechanism to disambiguate lexical classes of words. According to Williams, (1981), there are two common methods to identify lexical classes of a word. The first method is morphological, i.e. based on the type of suffixes a particular word takes. For example, Amharic nouns cannot take /-**äč**/ as their suffix but perfective verbs do take. The second method is syntactic, i.e. based on the position of a word in a sentence. For example, in Amharic a verb cannot come at the very beginning of a sentence and a noun cannot come at the end of a sentence, provided that the sentence is a declarative sentence (Baye, 1994). The remaining sections of this chapter discuss issues related to the structural aspects of Amharic phrases.

3.4 Phrasal Categories

A phrase is a structure in a language that is constructed from one or more words in the language. In Amharic phrases are categorized into five categories, namely noun phrase (NP), verb phrase (VP), adjectival phrase (AdjP), adverbial phrase (AdvP) and prepositional phrase (PP) (Baye, 1987). Each phrase type can be categorized into 'simple' (where only one word class is represented) and 'complex' (where more than one word class are represented).

3.4.1 Amharic Noun Phrases

A noun phrase (NP) is a syntactic unit in which the head (H) is a noun or a pronoun. An NP can be simple or complex. The simplest NP consists of a single noun (e.g. Kassa) or pronoun such as **ə́su** (he), **ə́sua** (she), **ə́näsu** (they), etc. A complex NP can consists of a noun (called *head*) and other constituents (like complements, specifiers, adverbial and adjectival modifiers) that modify the head from different aspects (Baye, 1987). For

example**,** in the NP **ya yätəmanəntu tələk yäsar bet** 'That yesterday's big thatched house', **ya** 'that' is a specifier, **yätəmanəntu** 'yesterday's' is an adverbial modifier, **tələk** 'big' is an adjectival modifier, and **yäsar** 'thatched' is also an adjectival modifier specifying the material from which the object named by the **bet** (head) is made of. **yäsar** is also called a complement of the head and, therefore, is not treated equally with the other modifiers.

The grammar rule for the above example NP can be formulated as NP →Spec AdvP Adj NP where the NP on the right side of the rule stands for **yäsar bet** 'thatched house' and can further be rewritten as NP →NP N. There are many combinations for the Amharic NP constituents (See chapter four) below.

3.4.2 Amharic Verb Phrases

A verb phrase (VP) is composed of a verb as a head and other constituents such as complements, modifiers and specifiers. For example, in the VP **Bä-mäkina wädä təməhərt bet hedä** '(He) went to school by car', both **Bä-mäkina** 'By car' and **wädä təməhərt bet** 'To school' are prepositional phrases (PPs) modifying the verb **hedä** 'went' from manner and place points of view, respectively. Therefore, structure rule for this example VP is VP → PP PP V. (more discussion of Amharic VPs and additional grammatical rules will be shown at the end of this chapter and in the next chapter).

3.4.3 Amharic Adjectival Phrases

The construction of Amharic adjectival phrase (AdjP) is similar to that of a NP and a VP. It can be composed of an adjective (head), and other constituents such as complements,

modifiers and specifiers. For example, in the AdjP **yač bät'am qonjo** 'That very beautiful (girl)', **yač** 'that' is a specifier, **bät'am** 'very' is a modifier modifying the the head of the AdjP (**qonjo** 'beautiful'). The structural rule governing this phrase is: AdjP → Spec Adv Adj.

3.4.4 Prepositional Phrases

Amharic prepositional phrase (PP) is constructed from a preposition (P) head and other constituents such as nouns, noun phrases, verbs, verb phrases, etc. In the PP ∂ndä säw bä-m∂d∂r 'like a man on earth', for instance, ∂ndä 'like' and bä 'on' are prepositions that are combined with the nouns säw 'a man' and m∂d∂r 'earth', respectively to form their PPs. The two PPs, in turn, combine to result in the bigger PP that is provided in the example. This PP is formed by the structural rule: PP → PP PP and each of the PPs on the right hand side can further be analyzed as: PP → P N. Sometimes one may find a PP (for example, zaf lay 'on a tree') that is rewritten as by the rule PP → N P.

3.4.5 Adverbial Phrases

An Adverbial phrase (AdvP) is constructed from one or more adverbs in the language. In the example, k∂fuňa tägačču '(They) crashed severely', k∂fuňa 'severely' (head) is the only adverb that formed the AdvP and is governed by the rule: AdvP → Adv. However, in tolo tolo tärämädä.'He walked briskly', the phrase is composed of two adverbs tolo tolo 'briskly' that make up the AdvP having the later tolo as its head and the structural rule is: AdvP → Adv Adv.

3.5 Sentence Formalisms In Amharic

As Getahun (1990) indicated, a sentence is a big phrase that expresses a complete idea. The sentence **hulät təlaləq ləjoch bämäkina wädä Gojam təlant hedu** 'Yesterday two big children went to Gojam by car', for instance, transmits a complete idea as it fully answers such questions as 'who?' 'What?' 'Where?' 'How?' and 'When?' When viewed from structural point of view, a sentence is a result of the combination of two phrases: an **NP** and a **VP** asits immediate constituents. As far as their ordering in a sentence is concerned, an NP always precedes a VP. All the remaining phrase types are subcomponents of a sentence and are found within one of the above main components of a sentence. Thus, in observing the structure of the above example, one can understand that it is composed of the NP (**hulät təlaləq ləjoch**) [Two big children] and the VP (**bämäkina wädä Gojam təlant hedu**) [went to Gojam by a car yesterday] which, in its turn, consists of VP (**təlant hedu**) and PP (**bämäkina wädä Gojam**), and so on. Based on the number of verbs they contain sentences can be classified into two: simple and complex sentences.

3.5.1. Amharic Simple Sentences

A simple Amharic sentence consists of an NP, which is the subject, followed by a verb phrase that comprises the predicate.

Səgaw ččoma näw.
The meat white is
The meat is white.

The morphemes like /-**u**/ that is attached to a verb (e.g. **hed-u**) refers to definiteness and number of the subject, and objects of the sentences. The morpheme that indicates the subject always precedes the morpheme that indicates the object. In the sentence, for instance, **Asəter bərččəqown säbäräču** 'Aster broke the glass,' /-**äč**/ indicates the subject Asəter, and /-**u**/ refers to the object **bərččəqowən** 'glass'. A given simple sentence may describe the state of being of the subject or an action that takes place in the sentence.

Example: **Kassa tämari näw** 'Kassa is a student'

tämarioču däbdabbe şafu 'The students wrote a letter'

The first example describes the present state of being of Kassa while the second describes a past action done by the students. Sentences of the latter type, in turn, are subcategorized as follows:

- those that do not involve an object of the verb (i.e. intransitives) like **Asəter wädä gäbäya hedäč** 'Aster went to market';
- those that involve transitive verbs with only one object like **tämarioču däbdabe TSafu** 'The students wrote a letter'; and
- Those that involve transitive verbs with two objects like **Asəter läKassa mäTSəhaf sät'äču** 'Asəter gave Kassa a book'.

Baye (1987) classifies simple sentences into four, namely: declarative sentences, interrogative sentences, negative sentences and imperative sentences. Declarative sentences are used to convey ideas and feelings that the speaker has about things, happenings, feelings, etc, that could be physical, mental, real or imaginary.

Example: **Aster astämari honäč**. 'Aster became a teacher'

A sentence that questions about the subject, the complement, or the action the verb specifies, is called an interrogative sentence.

Example**:** **Kassa mäče mät'a?** 'When did Kassa come?'

In order to construct interrogative sentences, Amharic sentences usually involve such interrogative pronouns as **man** 'who', **mənn** 'what', **yät** 'where', **sənt** 'how many', and **mäče** 'when'. These interrogatives can then be combined with prepositions to produce some more interrogative prepositional phrases like **käman** 'from whom'**, lämənn** 'why', etc. One interesting feature of Amharic interrogative sentences is the fact that the interrogative pronouns usually appear at the same position of the sentence where the thing being enquired would also appear.

Example:

- Q: **man tənnant hulät bägoč gäza?** 'Who bought two sheep yesterday?'
- A: **Kassa tənnant hulät bägoč gäza** 'Kassa bought two sheep yesterday'
- Q: **Kassa mäche hulät bägoč gäza?** 'When did Kassa bought two sheep?'
- A: **Kassa tənnant hulät bägoč gäza**. 'Kassa bought two sheep yesterday.'

Negative sentences simply negate a declarative statement made about something.

Example: **Kassa mənsa-wənn bäla**. 'Kassa ate his lunch'

Kassa mənsa-wənn al-bäla-m. 'Kassa did not eat his lunch'

In the above example, the verb in both sentences is bäla "ate". It is negated by the negation prefix /al-/ and the suffix /-m/ indicating that the subject of the sentence is masculine, third person singular. Simple imperative sentences convey instructions and mostly their subject is a second person pronoun that is usually but implied by the suffix on the verb.

Example: mäṣhaf am∂t'a! 'Bring a book! (Masculin)'

3.5.2. Amharic Complex Sentences

Complex sentences in Amharic are those sentences that are composed of complex phrases such as NP, VP, or AdjP. The pattern of combination could take the form of a complex NP and a simple VP, a simple NP and a complex VP, or both complex NP and VP. It is worth doing to see the structure of complex phrases before analysing how they combine with each other to construct complex sentences.

A complex NP is one that contains an embedded sentence with in it. The phrase, for instance, **Kassa yägäzaw yäsuf kot** 'The wool coat that Kassa bought' is a complex NP whose head is **kot** 'coat'. This head has been combined with the complement **yäsuf** 'wool' in order to produce the simple NP **yäsuf kot** 'a wool coat'. This simple NP, in turn, has combined with the dependent clause **Kassa yägäzaw** 'that Kassa bought' to produce the above complex NP. The presence of the relativizer, **yä** 'that' in it indicates that the clause is a subordinate clause and it cannot stand alone. The structure of this complex NP can be depicted by a parse structure tree as follows.

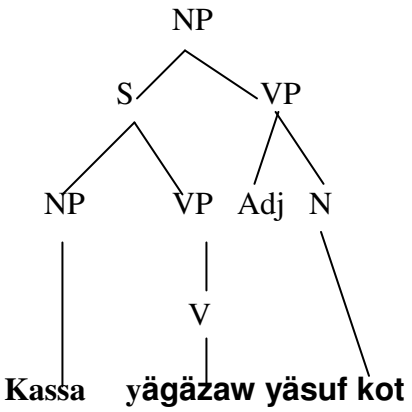

Figure 3.1: The Structure of a Complex Noun Phrase

One can understand from this tree diagram that the relative clause **Kassa yägäzaw** 'that Kassa bought' modifies the NP **yäsuf kot** 'wool coat'. The object of the verb **yägäzaw** 'that (he) bought' is only implied by the suffix **-aw** (3rd person singular masculine) and its person level is exactly the same as that of **yäsuf kot**. Hence, **Kassa yägäzaw** 'that Kassa bought' is referred to as a relative clause (Getahun, 1990). Likewise, a VP is complex if it contains more than one verb or a sentence within it. That is, like a complex NP, a complex VP also contains an embedded sentence that plays the role of a compliment or a modifier.

Asəter [[Kassa wädä Gojam səlähedä] aläqäsäch]

'Asəter wept for the reason that Kassa went to Gojam.'

The dependent clause here is **Kassa wädä Gojam səlähedä** 'for the reason that Kassa went to **Gojam'** and **səlä** is the part that made the clause dependent. As this clause indicates the reason for 'Asəter's weeping, it is used as an adverbial clause of reason. The structure of this VP can be depicted by a tree diagram as follows:

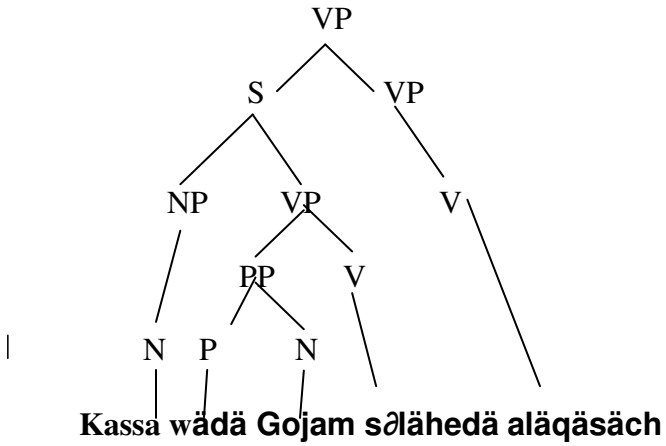

Figure 3.2: The Structure of a Complex Verb Phrase

Like NP and VP, adjective phrases (AdjP) can also be complex involving a clause in it like **käAsɔ̈ter yäbälät'äch qonjo**. Here, **qonjo** 'beautiful' is the head and **käAsɔ̈ter yäbälät'äch** 'that is better than **Asɔ̈ter** is the dependent clause. The structural feature of this phrase is represented by the following tree-diagram.

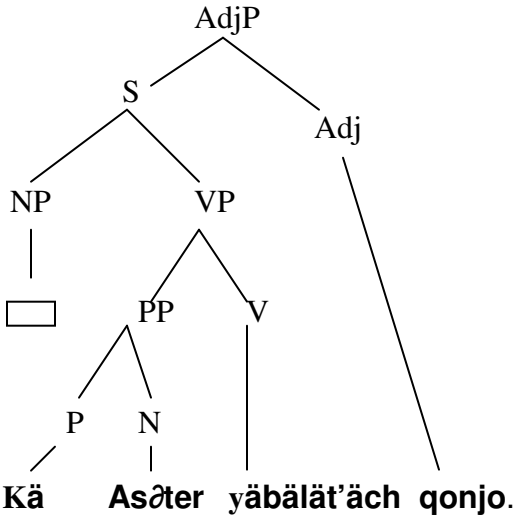

Figure 3.3: The Structure of a Complex Adjectival Phrase

As one can understand from the above tree-diagram, the embedded sentence involves the prefix **/yä**-/ attached on the verb making the clause dependent on the main clause. The suffix /-**äch**/ on the verb indicates that the implied subject in the dependent clause is 3sf, which is expected to appear in the bigger complex sentence where this Adjp is used as a sub-component. In the sentence, for instance, **Alɔmaz käAsɔter yäbälät'äch bät'am qonjo lɔj näch** 'Almaz is a girl who is more beautiful than Aster', the complex Adjp is a subcomponent of the complex VP **käAsɔter yäbälät'äch bät'am qonjo lɔj näch** and the suffix /-äch/ (3sf) on the verb is similar to the subject of the largest complex sentence, i.e. Alɔmaz. A complex sentence can also consist of a complex NP and a complex VP. The sentence, for example, **ɔnate käGojam yäamät'achat lɔj Kassa ɔndäwädädat bädänɔb awɔqaläch** 'The girl who came from Gojam knew well that Kassa loved her' is a typical example. The parse tree of this sentence would look like the following.

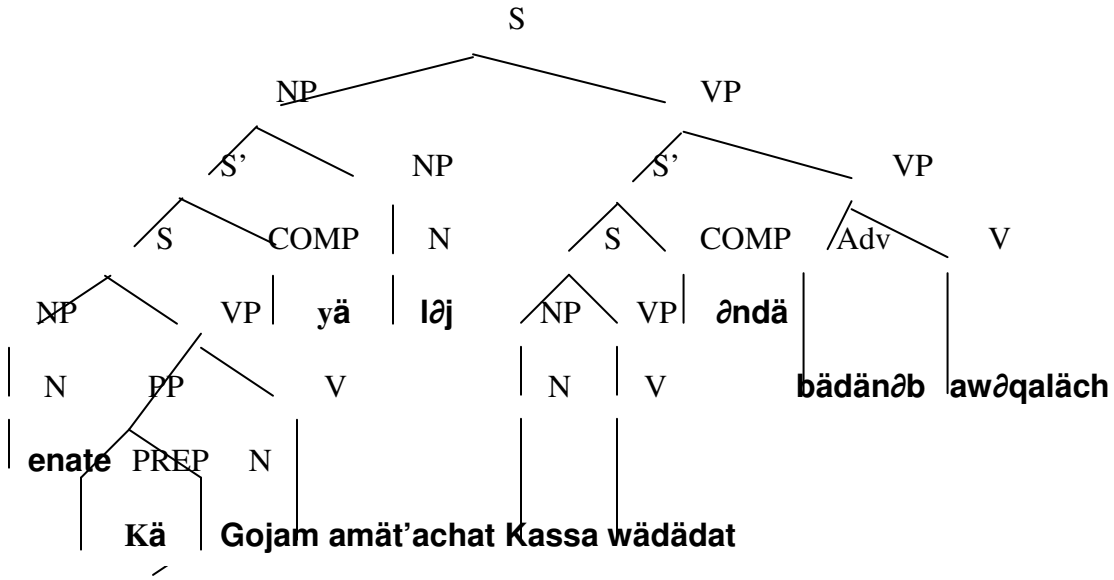

Figure 3.4: The Structure of a Complex Sentence

As it can be observed from the above complex sentence tree diagram, the position of the dependent clause is immediately before the part of the independent sentence it relates to. In order to avoid complexity, the study considered complex sentences that contain dependent clauses of which the nouns (subject of the verbs in the dependent clause) are explicitly indicated in the structure. Consider the following two examples of complex sentences:

a. **Asəter sat'ən wəst' yäalläun shurab lädäha sät'äĉ**.

'Aster gave the poor the clothes that is in the box.

b**. Asəter ənatua sat'ən yäasqämät'utn shurab lädäha sät'äĉ**.

'Aster gave the poor the clothes that her mother reserved.

To see the structural features of each of the dependent clauses in the above two example sentences, i.e. **sat'ən wəst' yäalläun** and **ənatua sat'ən wəst' yäasqämät'utn**

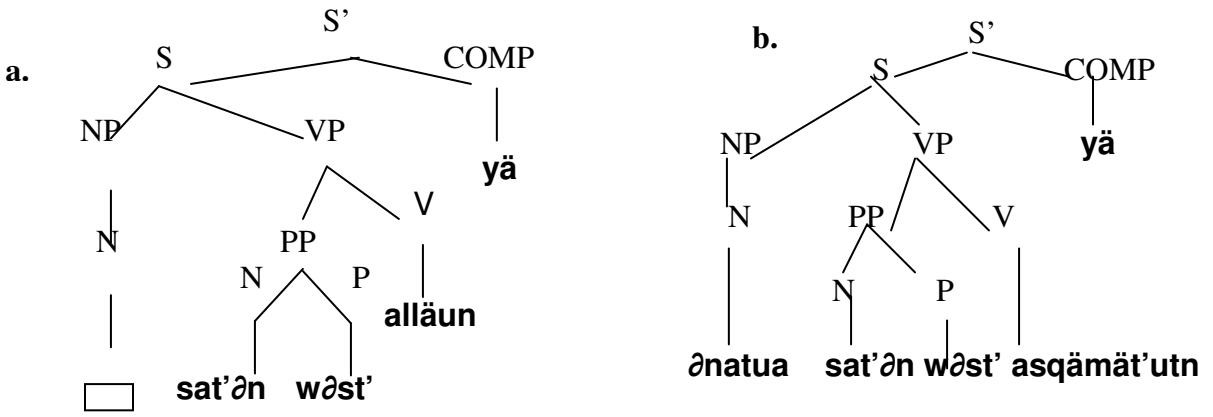

Figure 3.5: The Structure of a Dependent Clauses in Complex Structures

From the above tree diagrams one can understand that the subject of the dependent clause in 'a' is missing or is represented by a blank box, i.e. it is only found in the bigger structure that contains the whole complex sentence. However, in 'b' the subject of the dependent clause is explicitly mentioned, i.e. **ənatua** 'Her mother'. It is clauses of the

second type, i.e. where the subjects are explicitly mentioned that this study is concerned about.

To sum up, simple sentences are composed of simple noun phrase and simple verb phrases while complex sentences can consist of a complex NP and a simple VP, a simple NP and a complex VP, or a complex NP and a complex VP. Only complex sentence that are composed of a simple NP and a complex VP are considered in this study due to time and resource constraints. The next section discusses Amharic verb prefixes that are important for the formation of various types of clauses that are typical features of complex sentences.

3.6 Prefixes To Amharic Verbs In Clause Formation

There are different affixes that are attached to either end of Amharic verbs. Inflectional suffixes of Amharic verb stems indicate grammatical categories associated with nouns such as person, number, gender, tense and aspect. They are mostly subject and object markers. Based on the function they perform, Amharic verbal affixes can be categorized as clause markers, negative marker, subject markers of the imperfect stem and derived verb prefixes. Only clause markers of the Amharic verbal affixes are included in this study for the same reason mentioned for the type of complex feature focused at. As indicated in Abiyot (2000), clause markers in Amharic include those prefixes that are used as follows:

- **Indirect Object Indicator [lä-]**

Example: **halafiw čəgərun läat'änut säwoč bəzu gänzäb sät'ačäw** 'The head gave much money to the people who studied the problem'. Here the indirect object **säwoč** 'people' is expressed by the prefix lä.

- **Relative Clause Marker [yä-]**

Example: **ene Kassa yägäzawn näcc bäg təlant ayähut** 'Yesterday I saw the white sheep that Kassa bought'. /**yä**-/ is prefixed to the verb **gäza-w-n** 'bought' in order to make the clause **Kassa yägäzawn** 'that Kassa bought' a relative clause.

- **Reason Clause [səlä-]**

Example: **Läma säw səlägädälä bäpolis wädä t'abya täwäsädä** 'Lema was taken to the police station for killing a person'. The reason clause **säw səlägädälä** 'for he killed a person' is expressed by using the prefix /**səlä**-/.

- **Time Clause [əskä-]**

Example: **ləju ənatu bet əskädäräsächbät säat dəräs aläqäsä** 'The child cried until the time his mother reached home'. The particle **əskä** in **əskädäräsächəbät säat dəräs** 'until the time (she) reached home' expresses that this clause is a time clause.

- **Conditional Clause [kä-]**

Example: **Kassa enatu kämät'u wädä bet tolo yəmät'al** 'Kassa will come home soon if his mother comes'. As it is shown in the example, a conditional clause can be expressed using the prefix **kä** as in **enatu kämät'u** if she (polite) comes.

3.7 Conclusion

In summary, this chapter discussed issues in Amharic that are related to this study based on recent and early works. Although researchers in the area classify Amharic words into five, seven, or eight parts of speech categories, for the purpose of this study, they were categorized into seven parts of speech, namely *nouns*, *verbs*, *conjunctions,* adverbs, *adjective, prepositions*, and *interjections.* Five phrasal categories and various aspects of Amharic sentences (such as type and complexity) were identified and discussed. Amharic complex phrases of all sorts, features of verbal affixes that are important in Amharic clause formation as well as complex sentence construction, were all discussed. The focus of the study, which is parsing of complex sentences that are formed by a simple NP and a complex VP, was indicated.

The next chapter discusses the PCFG extraction and the incorporation and architecture of the Amharic NLP systems that may be essential for the development of the parser

CHAPTER FOUR

DATA PREPARATION AND PCFG EXTRACTION

4.1 Introduction

This chapter may be considered as the core of this study. Based on the assumptions and approaches discussed in chapter two and the syntactic property of Amharic reviewed in chapter three, this chapter discusses the PCFG that has been employed by the automatic complex sentence parser for Amharic developed in this study. The chapter begins by discussing the approaches taken to design the parser, and then moves to the second section where the sample text used in the study is analyzed. The third section discusses a simple morphological analyzer developed for the purpose of this study and, following that, the part of speech tagger developed by Mesfin (2001) and how it was embedded in the design and development of the parser is described. The fifth section deals with the extraction of a probabilistic context free grammar rule from the tagged corpus. The conversion of the rules to the Chomsky Normal Form (CNF) is presented in section six.

4.2. The Design Approach of The Parser

The use of statistical methods has greatly furthered progress in many areas of language processing. Disambiguation, database classification of documents, speech recognition and grammar learning are just some of the areas that statistics has helped. Statistics is the ideal tool for analysing certain regularities of a language frequently occurring phenomena (Cahil, Aoife, no year). Currently, efforts are being put to make large Amharic corpora

electronically available, and this will give researchers in the area of NLP the opportunity to extract the statistical information that can help them understand language.

As mentioned in the second chapter, the approach considered in the design of Amharic complex sentence parsing system is the PCFG bottom up chart parsing. As it was mentioned in chapter two, CFGs are powerful enough to describe most of the structure in natural languages, and they are important in that they are restricted enough so that efficient parsers can be built to analyze sentences (Allen, 1995). PCFGs, which define a language as a probability over strings, are the probabilistic version of CFGs and they have been used in a variety of applications (Cahill, Aoife, no year; Stocked, 1993). Compared to the CFGs, PCFGs are appear to be more useful for sentence parsing as they allow handling frequent parsing problems like structural ambiguity especially as grammar gets large, predicting the parse space, and ungrammatical sentence analysis.

Assuming that a sentence $w_{1,n}$ (where $w_{1,n}$ is a sequence of words w_{11}, … , w_{1n}) has T (n) possible parse trees (possible structures), and t_i is the ith parse tree that $0 < i \leq \mathrm{T(n)}$, then the probability of the i^{th} parse tree of the sentence $w_{1,n}$ is the product of the probabilities of all rules used in the parsing and is given by (Yao and Lua, 1998)

$$P(t_i) = \prod_{j=1}^{n} P(R^j) \qquad (2)$$

And the probability of the sentence $w_{1,n}$ is the sum of the probabilities of all possible parse trees that

$$P(w_{1,n}) = \sum_{t_i = 1}^{T(n)} P(t_i) \quad (3)$$

In statistical meaning, $P(t_i)$ shows the possibility of the ith parse tree of all possible parses, while $P(w_{1,n})$ shows how grammatical a sentence $w_{1,n}$ could be in the language. The larger the $P(t_i)$ is, the more reasonable the parse is. What this means is that to find the best parse of all possible parses of a sentence is nothing but to find $\max_{0<t_i \le T(n)} \{P(t_i)\}$. Therefore, $P(w_{1,n})$ and $\max_{0<t_i \le T(n)} \{P(t_i)\}$ are two important facts in syntax analysis that the first one gives the reasonability of a sentence in PCFGs, while the second one gives the most possible parse of all. In this study, the basic assumption made is, all sentences are considered to grammatical in the language. Accordingly, the focus is on the $P(t_i)$ values, to get the most probable parse structure.

The following section describes the sample corpus on which this algorithm was applied and the data pre-processing tasks that were done on it.

4.3 The Sample Corpus

For the purpose of this study, 350 Amharic complex sentences that were collected from two widely used grammar books of the language were considered. Due to lack of annotated text for the purpose of grammar induction and training, it was really much time that was spent during manual morphological analysis of each word, hand tagging and parsing process of each sentence in the sample corpus. However, the researcher admits

that the sample size considered is still very small as compared to the 4310 sentences that were used by Yao and Lua (1998).

The sentences were selected from books entitled "Ya Amarigna Sewasew" by Baye (1987) and "Amharic for Beginners" by Frydenlund and Svensen (1998), both of them were prepared with the purpose of serving as references for teaching Amharic language at tertiary and secondary levels, respectively. These books were chosen as references after discussion with linguistic advisors and the author of the former book, Prof. Baye Yemam. The sentences were selected randomly, though there was judgment by the researcher such as all the sentences constitute two or more of the phrase classes and embed one of the various clause types (i.e. relative clause, reason clause, result clause and time clause) all discussed in chapter three.

These sample sentences were transcribed according to the International Phonetic Alphabet (IPA) standard (See Appendix 3). The sentences were then hand tagged for later comparison purpose of the improvement made on the tagger. As the tagger accepts only .txt files, the transcribed sentences that were saved using this format exhibit some changes such as **ə** is changed to **?**. This is because of the fact that characters like ə are not supported in a .txt file. Similarly, **ĉ** and **ĉĉ** change to **c** and **cc**. The latter character **ĉĉ** stands for a more or less unique character of Amharic representing the velar explosive sound**.** Again since **ň** changes into **n**, which is already used for representing the usual nasal sound 'n', the former **ň** is re-represented by **nn** in the word code table for convenience. These kinds of changes were made for the sake of convenience and the

special tags identified by Mesfin (2001) and the tagged version of the sample sentences are found in Appendix 4 and 5, respectively.

Some of the parses of the sample sentences were given in Baye (1987) while the remaining sentences were hand parsed by the researcher and a high school Amharic teacher according to the phrase structure rules of the Amharic language. Comments and suggestions were then taken from the author of the book, an Amharic language expert and the linguistic advisor for the thesis. The probability calculations for the words in the sentences, grammar rule induction, and probability assignment to the grammar rules, were conducted using the 320 sentences (20% of the sample corpus). These sentences were picked randomly as training set while the rest 80 sentences (20%) from the corpus were used as a test set (See Appendix 3).

4.4 The Morphological Pre-processing

Although in simple examples and small systems one can list all the words allowed by the system, representing the lexicon for sentence parsers that entertain large vocabulary would face a serious problem. Not only there are a large number of words, but also each word may combine with affixes to produce additional related words. One way to address this problem is to preprocess the input sentence into a sequence of morphemes (Allen, 1995). In Amharic, for instance, a word may consist of a single morpheme (e.g., **bäg** 'A sheep'), but, as it was mentioned in chapter three, Amharic is one of the highly inflectional languages and, therefore, most of the words in the language consist of a stem plus an affix(s) (e.g., **bägoĉ** 'Sheep' Pl.; **bägoĉu** 'The sheep'; **bägoĉe** 'My sheep' etc.). With out any preprocessing, a lexicon would have to list all the forms of **bäg**,

including **bägoĉ**, **bägoĉu**, and **bägoĉe**. However, with preprocessing, there would be one morpheme **bäg** that may combine with suffixes such as **–oĉ**, **-oĉu**, -**oĉe**. Thus, in this case, the lexicon would only need to store one entrie (i.e., **bäg**).

As it was pointed out in chapter two, such a preprocessor (i.e., a morphological analyzer) is one of the most essential NLP systems in the development of a sentence parsing system. The morphological analyzer component that was intended to be incorporated in this study was the prototype Amharic morphological analyzer developed using Linguistica 2001 by Tesfaye (2002). The system analyzes words applying a series of heuristics. The first heuristic segments words into promising parts studying word patterns in a given corpus. He also constructed a morphological dictionary (called a **signature**), and used a **Minimum Description Length (MDL)** test in order to accept or reject a given signature as part of the morphology of Amharic. The test result shows that 403 words (86%) of the 500 words in the test corpus were analyzed correctly (are in good category). But the system gave spurious analysis for 20 words and failed to analyze 46 words and wrongly analyze one word.

Unfortunately, by the time this study was conducted, the researcher of this study failed to get and test the present corpus on Tesfaye's Linguistica 2001 morphological analyzer for Amharic. The main reason for this was the fact that Tesfaye's study focused more on Amharic Stems Morphological Analyzer (ASMA), a different system that handles the task of morphological analysis internal to Amharic stems (and which is available in his thesis but not the concern of the present study), than on Linguistica 2001. Thus assuming that Linguistica 2001 would handle the preprocessing tasks for an automatic sentence

parser, the correct outputs of Linguistica 2001 were analyzed, and a simple morphological analyzing component is used in this study. This preprocessing mechanism was built following what the first heuristic in Linguistica 2001 would correctly output, i.e. segmenting words into a free morpheme and a bound morpheme (if any). Then, it passes only the stem to the part-of-speech tagger, which is the other component system of this system.

Therefore, if the input sentence is the one given below:

ɔne kassa yägäzauɔn näĉ bäg tɔlant ayähut #

'I saw the white sheep that Kassa bought yesterday.'

The preprocessing heuristic takes in each word in the sentence at a time and passes the following to the parts-of-speech tagger:

ɔne kassa gäz näĉ bäg tɔlant ay #

Note that the words **ɔne**, **kassa**, **näĉ**, **bäg**, and **tɔlant**, are all single morphemes and the preprocessing mechansm outputs them as they are. However, both the words **yägäzauɔn** and **ayähut** are composed of a stem and affix (es). Hence, the function preprocessed each into a sequence of morphemes like **yä-gäz-auɔn** and **ay-ähut** and then passed only their stems (**gäz** and **ay**) to the parts-of-speech tagger.

4.5 The Part Of Speech Tagger

As it was mentioned in chapter two, Mesfin (2001) has developed a prototype simple automatic parts of speech tagger (POS tagger) for Amharic language. In his study, he used the Viterbi algorithm without any modification. A module for sentence splitter was also developed in order to facilitate the preparation of texts in a file to be tagged with

appropriate parts of speech. He identified 24 tags, which generally fall under the word classes discussed in chapter three. POS tags were assigned on the basis of the review made regarding the linguistic properties of the Amharic word classes. The study adopted the Stochastic Hidden Markov Model approach to develop the prototype. The results achieved by the tagger based on the small sample taken (a page long Amharic text), were high, 97% on the training set and approximately 90% on the test set.

In this study, the tagger receives a string of stems (i.e. a sentence containing only stems of the words it was built from) that the morphological analyzer outputs. This is primarily hoped by the researcher to considerably reduce the size of the database (particularly, the size of the word code table) that the tagger uses. For instance, the words **bäg** 'a sheep' and **bägoch** 'sheep (pl) that were given different entries in the word code table are now represented by a common stem word **bäg.**

The other potential advantage of inputting a string of stems into the POS tagger is making a considerable reduction on the occurrence of (the chance of) getting words whose category is not known by the tagger. The POS tagger, for instance, tags the word **gäza** 'he bought' as a verb V but may also tag **yägäzaun** 'that he bought', which is just an inflected form of the former word, as unknown category UNC just because **yägäzaun** but **gäza** is not found in the word code table.

Mesfin's tagger had made use of four tables in a database, namely word code table, category code table, lexical probability table and transitional probability table. Since the type of the data that the tagger originally used to accept differs from what the morphological preprocessor in this study passes to it, the whole content of the database

the tagger used was replaced by new data. The probability values of the lexical and transitional tables were also recalculated using the new data entered in the word code table. The changes that were made to each of these tables and the functions that are performed by them are discussed below.

Word Code Table:

This table is used to store words from the sample text and a corresponding word code is given sequentially for each word. The 261 words (taken from 23 sentences) that this table previously contained were replaced by 894 stems (taken from 320 sentences).

Category Code Table:

This table is used to store the 25 word categories identified with a corresponding category code. The total number of the total number of categories identified is 24, the 24^{th} tag being for all punctuations and an additional 1 for all unidentified words. Some minor changes such as the replacement of 'J' by 'Adj', which both represent the adjectives class in the POS tagger and the parser, respectively, have taken place.

Lexical Probabilities Table:

This table stores the probabilities of words given categories (tags) for each word in a given corpora. This is written as p (word\tag) or p (word\category) or shortly as p (w_i\C_i). For instance, p (**bäkälä**\N) denotes the (lexical) probability of **bäkälä** "it grew", which is a common name in the language, to be a **noun,** and p (**bäkälä** \V) denotes the (lexical) probability of **bäkälä** to be a **verb** in a given pre-tagged corpus. The lexical generation probability is estimated simply by counting the number of occurrence of each word by a category. Mathematically this is given by

$$P(W_i \backslash C_i) = \frac{\text{number of times } W_i \text{ appears in category } C_i}{\text{total number words with category } C_i} \quad (4)$$

Such probability values of the lexical tables were recalculated using the new data entered in the word code table.

Transition Probabilities Table:

Transition probabilities are denoted by $p(C_i \backslash C_{i-1} \ldots C_n)$ and refer to the probability of a tag given one or more previous tags. For instance, the probability of a noun to be preceded by an adjective is given by $p(C_i = \text{Noun} \backslash C_{i-1} = \text{Adjective})$. Depending on the value of n (which tells us the maximum number of categories being considered), we can have bigram (n = 2), trigram (n = 3) or in general an n-gram transitional probabilities. If (n = 2), the model is a bigram model and it is denoted by $p(C_i \backslash C_{i-1})$. If (n = 3), the model is a trigram model and it is denoted by $p(C_i \backslash C_{i-1} C_{i-2})$. These models assume that the probability of the occurrence a particular category depends solely on the one or more categories immediately preceding it. In practice, given a database of texts tagged with part of speech, the bi-gram or transitional probabilities can be estimated simply by counting the number of times each pair of categories occurs compared to individual category counts. Mathematically this is written as:

$$P(C_i = \propto / C_{i-1} = \beta) = \frac{\text{count of the number of times } \propto \text{ and } \beta \text{ occur together in the corpus}}{\text{Number of times } \beta \text{ occurs in the corpus}} \quad (5)$$

Where ∝ and β are parts of speech codes. Such probability values of the transitional probability table were recalculated using the categorical sequence of the newly entered data into the word code table.

4.6 Extraction of A Probabilistic Context Free Grammar

In the process of extracting the PCFGs, the sentences that are used as a training set were first hand parsed and represented in the following manner.

S (NP (N **?ne**)

VP (S' (S (NP (N **kassa**) VP (V **gäzaun**)) (COMP **yä**))

VP (NP (Adj **näcĉ**) (N **bäg**) VP (Adv **t?lant**) (V **ayähut**))))

Following this, the CFG rules were extracted from these manual parses of the training set. Then, in order to gather statistical information about the grammar rules observed, counting the number of occurrences of each rule, which was found to be the simplest way for the purpose, was carried out in the manually parsed training sentences. Later, these statistical figures were used to estimate the probability of each rule being used (See also Allen, 1995). Finally, the probability values were assigned to each of the extracted rule using the formula:

$$P(Rj/C) = \frac{\text{Count of the number of times Rj occurs in the PCFG table}}{\text{Count of the total occurrence of the category C on the LHS in the PCFG}}$$

4.7 Chomsky Normal From (CNF) Representation

The representation of the PCFG in the Chomsky Normal Form (CNF) is carried out for the purpose of simplicity.

After the probabilistic context-free grammar was extracted, the next thing that needed to be done was to convert it into Chomsky Normal Form (CNF). This conversion was done for the purpose of simplicity. Any CFG rules can be easily re-written in the CNF, and this restriction will not result in the loss of any real expressive power. It means each rule $R^i \in R$ is in one of the following two forms,

$$R^i : N^i \rightarrow w^j \qquad \text{or} \qquad R^i : N^i \rightarrow N^j N^k$$

where $w^j \in W$ is a terminal symbol and $N^i, N^j, N^k \in N$ are non-terminal symbols. A grammar in CNF would also be easier to manipulate because of the binary structure it attains.

Rules of the form A --> BCDE (p), where p is the probability, are replaced by a set of rules (Abney, 1996):

A → BC' (p), C'→ CD' (1) and D'→ DE (1)

The PCFGs extracted for the Amharic language are converted to CNF manually. This conversion was in line with the phrase structure rules of the language and in actual sense it related to representing it in the exact structure more than converting it to the CNF. That is, since all sentences considered were 4-word sentences, most of the rules were already in the CNF. Examples are given below.

The rules:

VP → Adv Adv V (0.14)

VP → Adv N V (0.17)

Table 4.1 below depicts an example of the extracted CNF rules with their respective probability values.

LHS	RHS1	RHS2	Probability
S'	S	COMP	1
NP	Adj	R1	0.006
R1	Adj	N	1
NP	AdjP	NP	0.005
NP	N	E	0.647
NP	N	N	0.004
NP	N	P	0.028

Table 4.1 Probabilistic Context Free Grammars in CNF

4.8 Conclusion

In this chapter, the PCFG parsing approach implemented and the sample text used in this study were described. Also, the simple morphological analyzer that preprocesses the input to the embedded POS tagger, the POS tagger itself and the changes that are made in its database are discussed. Finally, the extraction of a probabilistic context free grammar rule from the tagged corpus as well as some of the conversion of the rules to the Chomsky Normal Form (CNF) was presented.

CHAPTER FIVE

PARSING ALGORITHM AND EXPERIMENTATION

5.1 Introduction

This chapter discusses the parsing algorithms used to develop the prototype Amharic complex sentence parser, the experiments conducted, and the analysis and discussions made based on the findings of the experiments. Besides, the design of the parser, which comprises the input output interface, the probabilistic rule base and the chart parsing module, is presented in this chapter.

The next section presents a discussion on the Inside-Outside algorithm based on which this parser was designed. The third section of this chapter introduces and discusses PCFG parsing as it was implemented by the Inside-Outside algorithm. The fourth section describes the design of the parser while the fifth section presents a report on the experimentation, the results observed and the solutions given.

5.2 The Parsing Algorithm

Ambiguity resolution is a crucial problem facing computational models of natural languages. A given sentence, for instance, can have many possible syntactic parses that can be obtained, and are collectively known as its parse space. The probabilities of specific parse trees of a given sentence can be found using a chart parsing algorithm, where the probability of each constituent is computed from the probability of its sub-constituents and the probability of the rules used. Thus, the probability of an entry E (also called node $N_{(I,j)}$ in this thesis) of category C using a rule I with n sub-constituents corresponding to entries E_1, …, E_n,

$$P(E) = P(Ri/C) * P(E_1)\ldots * P(E_n) \quad (6)$$

Together with the chart parsing algorithm, a step that computes the probability of each entry when it is added to the chart can be implemented. The standard algorithm for computing this probability is called the Inside-Outside algorithm. For computing $P(w_{1,n})$ and $\max_{0<t_i \le T(n)} \{P(t_i)\}$, either of the two probability calculations are required. That is, the inside probability or the outside probability, for each node in the parse structure. For a word sequence $w_{k,l}$ derived directly or indirectly from a non-terminal node $N^j_{k,l} \in N$, the inside probability is:

$$\beta_j(k,l) = P(w_{k,l} | N^j_{k,l}) \quad (7)$$

And the outside probability is:

$$\alpha_j(k,l) = P(w_{1,k-1}, N^j_{k,l}, w_{l+1,n}) \quad (8)$$

Where $\beta_j(k,l)$ denotes the probability of the word sequence $w_{k,l}$ in which all the words are inside the node $N^j_{k,l}$, and $\alpha_j(k,l)$ denotes the probability of the words outside the node $N^j_{k,l}$ (Yao and Lua, 1998).

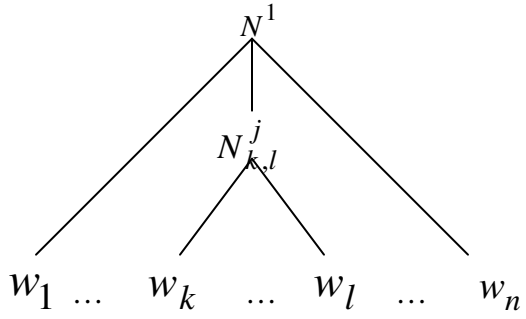

Figure 5.1 The words outside and inside $N^j_{k,l}$.

The above figure shows the words, $w_1, \ldots, w_{k-1}$ and $w_{l+1} \ldots w_n$ are outside $N^j_{k,l}$ while the words $w_k \ldots w_l$ are inside $N^j_{k,l}$. The probability of the best parse tree of a sentence which is $\max_{0<t_i \leq T(n)} \{P(t_i)\}$ can, therefore, be obtained by calculating the inside probability or the outside probability. The parser developed in this study is based on this algorithm with a modified parse chart proposed by Yao and Lua (1998) to assist the parsing. The algorithm used for finding the best (the most probable) parse structure for a given sentence is given in section 5.4.

5.3 PCFG Parsing

In this study, a parser based on the Inside-Outside algorithm was developed to implement PCFG parsing on Amharic complex sentences. Its parse mechanism, parse chart, and configuration are introduced in this section.

5.3.1 Parse Tree

Parsing a sentence involves finding a possible legal structure (or structures) for the sentence and this usually results in a tree, often referred to as a *parse tree*. Since the PCFG rules in this study are restricted to the CNF, each possible parse tree of a sentence is a binary tree. Generally, in a given parse tree, level 1 is always a set of terminal nodes, which are words, and level 2 is a set of non-terminal nodes, which are the POS tags of the corresponding words. In CNF, this is written in the form $N^i \rightarrow w^j$, and each node in level 2 only has a descendant in level 1.

For a non-terminal node N^i, if there is a rule $N^i \rightarrow w^j$, then N^i is supported by the grammar rule $N^i \rightarrow w^j$; and if there is a rule $N^i \rightarrow N^j N^k$, then N^i is supported by the grammar rule $N^i \rightarrow N^j N^k$. Assuming that each word in an *n*-word sentence $w_{1,n}$ has one POS, tag, *T* (*n*) denotes the maximum number of structure trees that $w_{1,n}$ probably have, and each non-terminal node of the trees is supported by a grammar rule, then T(n) can calculated as follows:

$$T(n) = \begin{cases} 1, & n = 1 \\ \sum_{i=1}^{n-1} T(i)T(n-i), & n > 1 \end{cases} \quad (9)$$

This indicates that the value of *T* (*n*) increases exponentially as the number of words in a sentence increases though practically there are trees in the parse tree space that involve nodes that are not supported by grammars rules and, therefore, the number of all parse trees of a sentence is less than *T* (*n*).

5.3.2 Parse Chart

As it was indicated earlier, this study adopted the parse chart that was designed by Yao and Lua (1998) to implement the Inside-Outside algorithm. This parse chart was based on equation (9) in section 5.3.1 and is a matrix but only top-right half would be used since it is a symmetrical matrix. The size of this chart depends on the number of words in the sentence being parsed. It means that for an *n*-word sentence the chart was an *n*×*n* matrix.

Note that the process of analyzing Amharic complex sentences starts prior to passing the input sentence into the parse chart. Thus a 5-word long input sentence would be tagged into 6 POS tags, a 6-word long sentence into 7 POS tags, a 7-word long sentence into 8,

and so on. This was only because of the heuristics used to deal with movements of verbal affixes during structural representation of Amharic complex sentences. This is illustrated by the example **qonjoua lə̀j Kassa ə̀ndäwädädat bädänə̀b awäqäch** # 'The beautiful girl knew that Kassa loved her,' and the following figure.

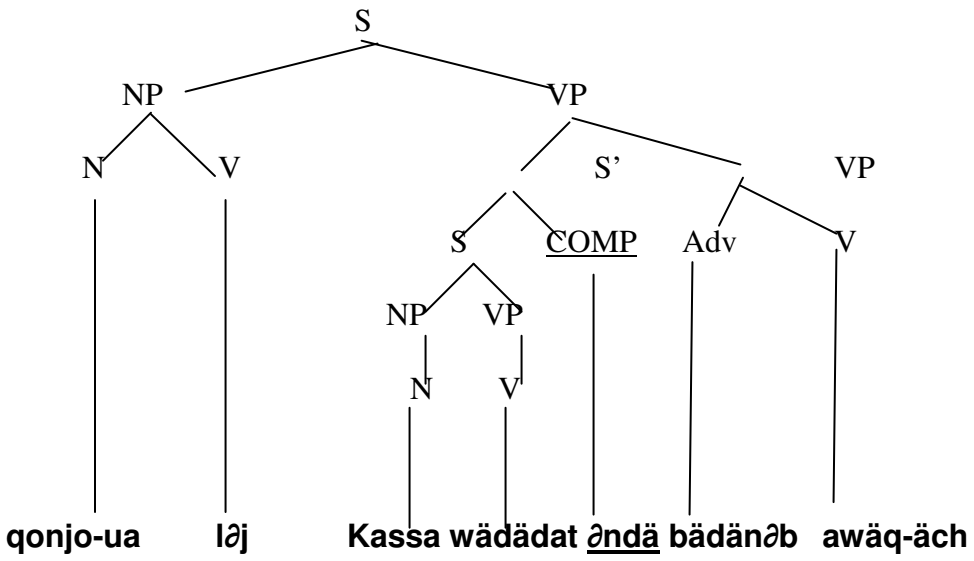

Figure 5.2: Tree structure diagram of the Amharic complex sentence:

qonjoua lə̀j Kassa ə̀ndäwädädat bädänə̀b awäqäch.

As it is illustrated by the above figure, the underlined verbal affix **ə̀ndä** 'that', detached itself from **ə̀ndäwädädat** 'that he loved her', and took the position after the verb to which it was first prefixed and got a new category COMP. This was handled before the tagged input was passed to the parser. All the Amharic complex sentences that were considered in this study were formed by embedding clauses that involve such relativizers, which were then analyzed into verbs and verbal affixes (called ***Complements*** COMP). The element $N_{(i,j)}$ $(i, j \in [1, 8])$ in the figure denotes a non-terminal node and the element $N_{(1,8)}$ is the starting symbol if each $N_{(i,j)}$ is supported by a grammar rule. The non-

terminal node $N_{(i,i)}$ ($i \in [1, 8]$) in the diagonal supported by the rule $N_{(i,i)} \rightarrow w_i$ is the POS of the word w_i. Figure 5.3 below shows an example of 8-level-chart, which can parse 7-word sentences.

j

i							
$N_{(1,1)}$	$N_{(1,2)}$	$N_{(1,3)}$	$N_{(1,4)}$	$N_{(1,5)}$	$N_{(1,6)}$	$N_{(1,7)}$	$N_{(1,8)}$
	$N_{(2,2)}$	$N_{(2,3)}$	$N_{(2,4)}$	$N_{(2,5)}$	$N_{(2,6)}$	$N_{(2,7)}$	$N_{(2,8)}$
		$N_{(3,3)}$	$N_{(3,4)}$	$N_{(3,5)}$	$N_{(3,6)}$	$N_{(3,7)}$	$N_{(3,8)}$
			$N_{(4,4)}$	$N_{(4,5)}$	$N_{(4,6)}$	$N_{(4,7)}$	$N_{(4,8)}$
				$N_{(5,5)}$	$N_{(5,6)}$	$N_{(5,7)}$	$N_{(5,8)}$
					$N_{(6,6)}$	$N_{(6,7)}$	$N_{(6,8)}$
						$N_{(7,7)}$	$N_{(7,8)}$
							$N_{(8,8)}$

Figure 5.3: An 8-level-chart

Therefore, in this study, for an **n**-word sentence $w_{1,n}$**,** if each non-terminal node is supported by a grammar rule, there are **n+1** terminal nodes in level 1 which is the diagonal, ***n*** non-terminal nodes in level 2, **n-1** non-terminal nodes in level 3,…, and **1** non-terminal node (the starting symbol) in level **n**. in the above chart, let $P(N_{(i,j)})$denote the probability of node $N_{(i,j)}$, then each node $N_{(i,j)}$ **(*i=j*, and *i*, *j*∈[1, 8])** in level 1 is determined by $P(N_{(i,j)}) = P(N_{(i,j)} \rightarrow w_k)$, which is completely handed by the tagger, and each node $N_{(i,j)}$ in the non diagonal levels is determined by the equation

$$P(N_{(i,j)}) = \prod_{k=1}^{j-1} P(N_{(i,j)} \rightarrow N_{(i,k)} N_{(k+1,j)}) P(N_{(i,k)}) P(N_{(k+1,j)}) \qquad (10)$$

The calculations of all these non non-diagonal nodes (e.g. **$N_{(1,2)}$, $N_{(1,3)}$,** and **$N_{(1,5)}$**)is indicated by charts **5(b), 5(c),** and **5(d)** on figure 5.

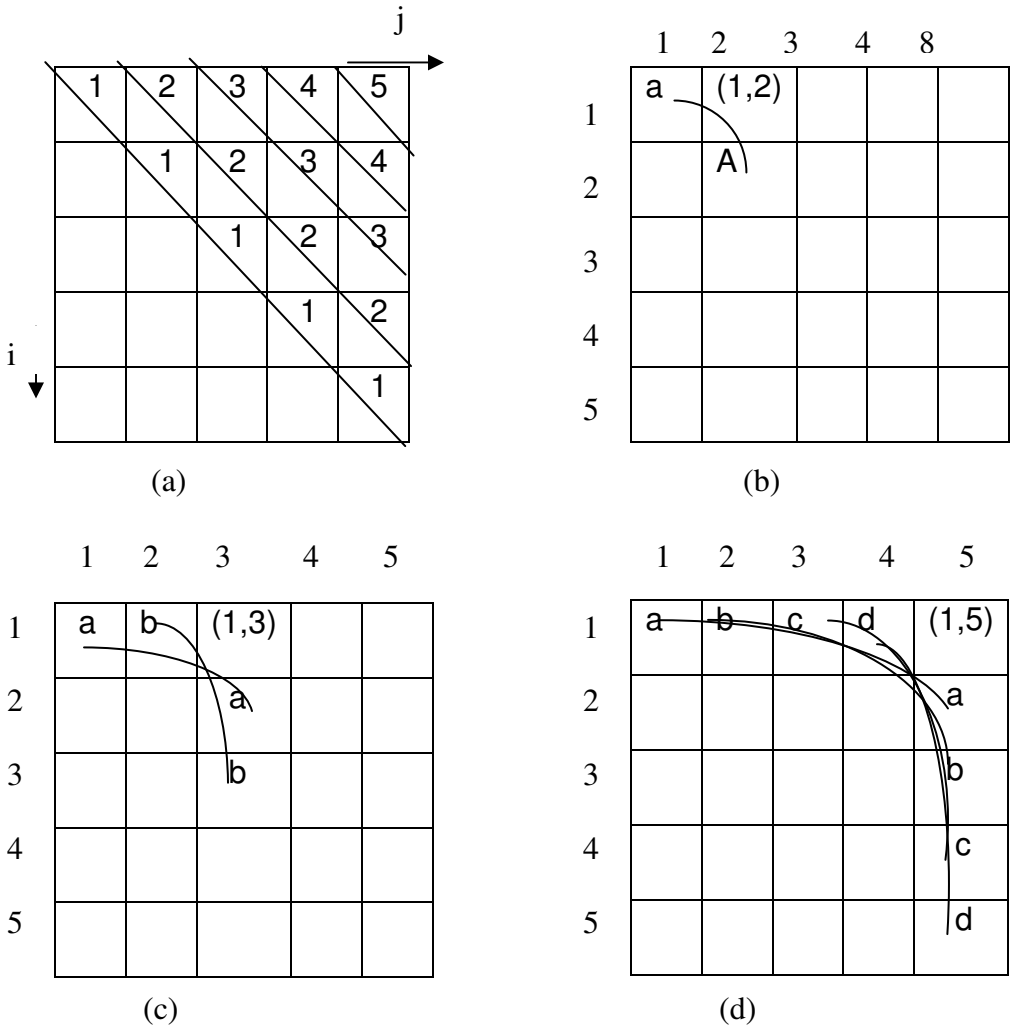

Figure 5.4: The calculation of probabilities of non-terminal nodes

In the above charts, (a) shows a 5 level chart, (b) indicates that a node in level 2 depends on "aa" in level 1, (c) shows that a node in level 3 depends on "aa" and/or "bb" in level 1 and 2, and (d) shows that a node in level 5 depends on "aa", "bb", "cc", and/or "dd". This parse chart gives the parse tree space for a given input sentence, and calculating the value of $N_{(i,j)}$ based on the chart would result in the calculation of the inside probability of

($N_{(i,j)}$). The tree with the maximum probability $\max_{0<t_i \le T(n)} \{P(t_i)\}$ can then be obtained from the structure that consisted of rules with the maximum probabilities. This would also result in the calculation of $P(w_{1,n})$, if there is an interest to determine how grammatical a sentence is in the language.

5.4 The Design of the Parser

- **Preprocessing the Parser's Input**

A sentence is input from a file for parsing. For example, if we take the sentence:

?ne kassa yägäzaun näĉĉ bäg t?lant ayähut #

'I saw the white sheep that Kassa bought yesterday.'

Here the # symbol indicates the end of the sentence. When this sentence passes through the morphological analysis process, each word is analyzed into a stem and affix(s), and outputs the following format that the tagger would use as an input:

?ne kassa gäz näĉĉ bäg t?lant ay #

The tagger tags each stem with the appropriate POS and outputs the following format

?ne\N **kassa**\N **gäz**\V **näĉĉ**\Adj **bäg**\N **t?lant**\Adv **ay**\V **#**\PUNC

Each tagged stem will then be re-synthesized with its respective affix using a hyphen (-) as a separator between a stem and its affix(s) as shown below.

?ne\N **kassa**\N **yä-gäz-aun**\V **näĉĉ**\Adj **bäg**\N **t?lant**\Adv **ay-ähut**\V **#**\PUNC

Following this, each word will go through a post tagger morphological process to see if the category of each tagged stem changes when reunited with the affix(s). This is a very important stage in the development of the parser not only for the reason just mentioned, but also it simplifies the complex process of parsing Amharic complex sentences. In other words, Amharic verbs that take verbal affixes (such as **s?lä, ?ndä, yä**), and, therefore, often called *Relativizers*, are identified and handled at this stage. Below is a sample of the table that performs this task (See Appendix 2 for the complete table).

Stem Cat.	Suffix	Prefix	New Cat.
N	E	E	N
V	**aun**	E	Rel
V	**u**	**s?la**	Rel
V	**ä**	**?ndä**	Rel
Adv	E	E	Adv
V	**ähut**	E	V

Table5.1: A Table For Checking Categorical Changes of Inflected Words

In the table, E stands for empty affix position. For each word composed from the tagged stem and the corresponding affix, if the stem category value and the affixes information of a given row matches the stem category and affix information of the word, the word's category will be updated by the corresponding new category, which is the resulting category of the stem and the affix(s). Also, whenever a verb that is prefixed by one of the

verbal affixes such as **yä** is found (e.g. **yägäzaun**), it will be divided and tagged as a verb (**gäz-aun**\V) and a complement (**yä**\COMP). Infixes are not incorporated in this table for the following two reasons. First, they involve detail consonantal and vocalic features which are out of the scope of this study. Secondly, it is the other types of affixes, particularly the prefixes that play a major role in the formation of the Amharic complex sentences that are considered in this study. Finally, the input sentence would attain the following format, and will be passed to the sentence parsing module:

?ne\N **Kassa**\N **gäz-aun**\V **yä**\COMP **näcĉ**\Adj **bäg**\N **?lant**\Adv **ay-ähut**\V #\PUNC

The input preprocessor module algorithm is provided below.

For each sentence in the file
Get one sentence at a time
For each word in the sentence
Call the Morphological Analysis Function
Get the stem of each word
Call the HMM POS tagger
Get the tagged stems of the sentence
Call the Morphological Synthesizing Function to update the category output of the tagger
Pass the final tagged string of words to the parser

Figure 5.5: An algorithm for preprocessing an input sentence to the parser

- **The Input & Output Interface**

A sample interface was designed for the sentence parser application program. When the Amharic complex sentence parser application program runs, the Amharic complex

sentence parser window is displayed. In fact, this is the main interface that will be displayed when the application program runs for the first time and it persists until the program is exited. This window has four main buttons, just below the menu bar. Below is a brief description of each of these buttons.

The Morph Analysis button

Clicking this button displays the stems of all the words in the input sentence. This information would enable a user view all the possible stems of the input sentence.

The POS Tagger button

Clicking this button displays the tagged stems of the input words.

The Tagged Sentence button

Clicking this button displays the final tagged input sentence isolating and tagging the complement independently among the other words.

The Parse button

This button allows a user to parse sentences from a saved file one by one.

Therefore, the prototype developed performs its function as follows. Phonetically transcribed Amharic sentences are input from a file for parsing.

mäng?st hagäritu yädäräsäbatn käf?tänna yäekonomi kisara amänä #

'The government admitted the economic crisis that the country faced.'

The morphological analysis component takes in the input sentence and outputs a string of stems as given below:

mäng?st hagär däräs käf?tänna ekonomi kisara amän #

The tagger then takes in the above string of stems as an input, tags each stem with the appropriate POS tag and outputs the following format that the parser would use as an input.

mäng?st\N **hagär**\N **däräs**\V **käf?tänna**\Adj **ekonomi**\N **kisara**\N **amän**\V **#**\PUNC

However, before the above output of the tagger is passed to the parser, each tagged stem word would be synthesized with its affixes (if any) and the resulting word's category will be searched from a table that updates the categories of inflected stems. Thus in passing through this process the above Amharic complex sentence, for example, would have the following format.

mäng?st\N **hagär-itu**\N **däräs-äbatn**\V **yä**\COMP **käf?tänna**\Adj **yä-ekonomi**\Adj **kisara**\N **amän-ä**\V **#**\PUNC

In the above example, those words in the input sentence that are affected by the above processes are indicated by underline. This was the actual output of the POS tagger assisted by the morphological analysis. The parser then extracts each word and each POS tag from the tagged sentence and stores them in one dimensional array variables. The output of the parser gives the probability of the selected parse structure, the parse result, and the grammar rules used in parsing. For the example sentence shown so far, the outputs include:

The probability of the parse structure: 0.000034129851158584

The parse result:

S (NP (N **mäng?st**)

VP(S'(S (NP (N **hagär-itu**) VP (V **däräs-äbatn**)) (COMP **yä**))

VP (NP (Adj **näcĉ**) (Adj **yä-ekonomi**) (N **kisara**) VP (V **amän-ä**))))

The rules applied in parsing the above sentence include:

S ➔ NP VP	S' ➔ S COMP	COMP ➔COMP' E
NP ➔N E	VP ➔ V E	NP ➔Adj R1
VP ➔S' VP	VP ➔NP VP	R1 ➔Adj N

The rules used in parsing are the grammar rules used and here those rules containing terminal nodes (for example, N → **mäng?st**) are not displayed since such lexical rules are not included in the PCFG table. This is because the parser inputs pre-tagged sentences and lexical information is handled by the component systems that preprocess sentences before the parser.

In the above example, it is also important to note that the word **yäekonomi** 'economical' got its base form **ekonomi** 'economy' and was tagged as noun N by the tagger. However, when its affix was brought back to its former place (i.e., when the stem was synthesized with its prefix **/yä-/**), the word exhibited a categorical change, i.e., from (N **ekonomi**) to (Adj **yäekonomi**). This was the main contribution of the morphological analysis in this study.

- **The Probabilistic Rule Base**

The PCFG rules, which are in CNF, were extracted form the sample corpus and are represented in the same database as the lexical probability table and others in the following format.

PCFG-CNF			
LHS	**RHS1**	**RHS2**	**Probability**
S	NP	VP	1
NP	Adj	R1	0.006
R1	Adj	N	1
NP	AdjP	NP	0.005
S’	S	COMP	1
COMP	COMP	E	1
VP	Adv	V	0.114
VP'	S'	VP	0.262
VP	V	E	0.332
NP	Adj	N	0.185

Table 5.2: PCFG Representation

In the above table, E is an empty production. The field LHS stores the left hand side of the rules while RHS1 and RHS2 represent the first and the second right hand side rules, respectively. The probability field represents the associated probability values for each of the grammar rules. (For a complete list of the PCFG rules extracted from the corpus see Appendix 5). Word information needed in the parsing is stored in the Word Code and Lexical Probabilities tables that were implemented by the tagger and stored in the same database as the PCFG table.

- **The chart parsing module**

This is the PCFG Inside-Outside algorithm assisted by the parse chart that Yao and Lua (1998) designed based on equation 9 in order to implement the Inside-Outside algorithm.

For each non terminal node N(i,j) in the parse chart where i is different from j and i, j = [1,n], its value is determined using the formula given in equation 10.

The following is the Inside-Outside algorithm that is used in this study to implement PCFG parsing.

For each POS tag in the diagonal
Check if a POS tag occurs alone or together with the next one at the right hand side of a rule:

If it occurs in both cases , get the highest probability of these rules and store its left hand side on the POS matrix, store the rule in the chartentry table and store the probability value in the Probability array

Else store its left hand side on the POS matrix, store the rule in the chartentry table and store the probability value in the Probability array

Else, store 0 in the Probability array and leave the chart matrix value empty
Set the lexical probabilities of each of the words to their corresponding categories.
Get the maximum of $P(t_i)$ and assign it to be the best parse.

Figure 5.6: The Parse Chart Procedure to Implement the Inside-Outside Algorithm

This algorithm was also coded in the development of the prototype and was used to calculate the inside probabilities of each parse in the parse tree space. In calculating the probabilities, the algorithm used equation 12 that adds a step to construct the parse bottom up.

During parsing, the categories of words in a sentence are fed one by one into the diagonal (i.e., the first level) of the chart. Here, the parse tree space is constructed bottom to top and the probability of each parse is calculated as well.

For each sentence to be parsed

Extract the words and tags from the output of the tagger.

Insert the POS tags into the diagonal of the chart matrix

Call the parse chart procedure that implements the Inside-Outside algorithm to calculate the probability of each node in the chart matrix.

Get the highest probability from the Probability array.

Get the rules that produced the highest probability from the chartentry table.

Generate the parse structure and display the output.

Figure 5.7: The Over all Implementation of The Parsing Algorithm

Generally speaking, the first thing that the program would do is the extraction of the words and POS tags from the output of the preprocessing modules. Consider the following example:

?ne\N **kassa**\N **gäz-aun**\V **yä**\COMP **näcĉ**\Adj **bäg**\N **t?lant**\Adv **ayähut**\V **#**\PUNC

While the words are stored in an array for later parse construction, the POS tags are inserted into the diagonal of the chart matrix, and this step would produce the following chart format:

j

i

N	$n_{(1,2)}$	$n_{(1,3)}$	$n_{(1,4)}$	$n_{(1,5)}$	$n_{(1,6)}$	NP	S
	N	$n_{(2,3)}$	$n_{(2,4)}$	NP	S	S'	VP
		V	$n_{(3,4)}$	$n_{(3,5)}$	VP	Comp	VP
			Comp	$n_{(4,5)}$	$n_{(4,6)}$	$n_{(4,7)}$	$n_{(4,8)}$
				Adj	$n_{(5,6)}$	$n_{(5,7)}$	$n_{(5,8)}$
					N	$n_{(6,7)}$	$n_{(6,8)}$
						Adv	$n_{(7,8)}$
							V

Figure 5.8: Chart Structure During Parsing

In the above chart, N stands for noun while $n_{(i,j)}$ represents a node that is found at the i^{th} row and j^{th} column of the chart matrix. The nodes S, NP, and VP always appear at the same position as in the figure since all the sentences in the corpus share these constructs. Also, S', Comp, and VP are common features of the complex sentences considered in this study, therefore, take the same position as in the chart. Nodes like $N_{(1,2)}$, $N_{(1,3)}$, $N_{(2,3)}$, etc are to be replaced by the non-terminals that are retrieved from the PCFG table in the database. To do this, the records in the table are searched for rules that produce, for instance, each node on the diagonal with an empty production E or a given POS tag with the next one, like N and N, N and V, V and COMP, etc. at the right hand side. These rules could be one or more and are stored in a chart entry table that is designed to hold all the rules that were expanded so far, and the respective probability values are stored in a probability array. Then the entries are replaced by the left hand side of the highest probability rules that generated the pair of POS tags in the diagonal. In the above matrix, for instance, the node $N_{(1,2)}$ is replaced by NP, since the highest probability constituent to produce the corresponding sequence N N at the right hand side of a rule is the VP (noun

phrase). All the remaining entries are replaced in the same manner, and finally, the matrix would have the form:

j ———

i								
	N	NP	NP	E	E	E	NP	S
		N	E	NP	NP	S	S’	VP
			V	E	VP	VP	Comp	VP
				Comp	E	E	Comp	
					Adj	NP	E	VP
						N	E	E
							Adv	VP’
								V

Figure 5.9 A complete POS Matrix

The chart entry table now holds all rules that were expanded so far. The above complete chart matrix shows that the Inside probability calculation for each possibility of a 7 word sentence (which is equal to 8 POS tags in the case of complex sentences) is done at this point.

Once the highest probability structure is identified, the rules that produced the probability are retrieved from the chart entry table, and the corresponding parse structure is constructed using these rules, the words array and opening and closing brackets.

The parser is represented diagrammatically in the figure below.

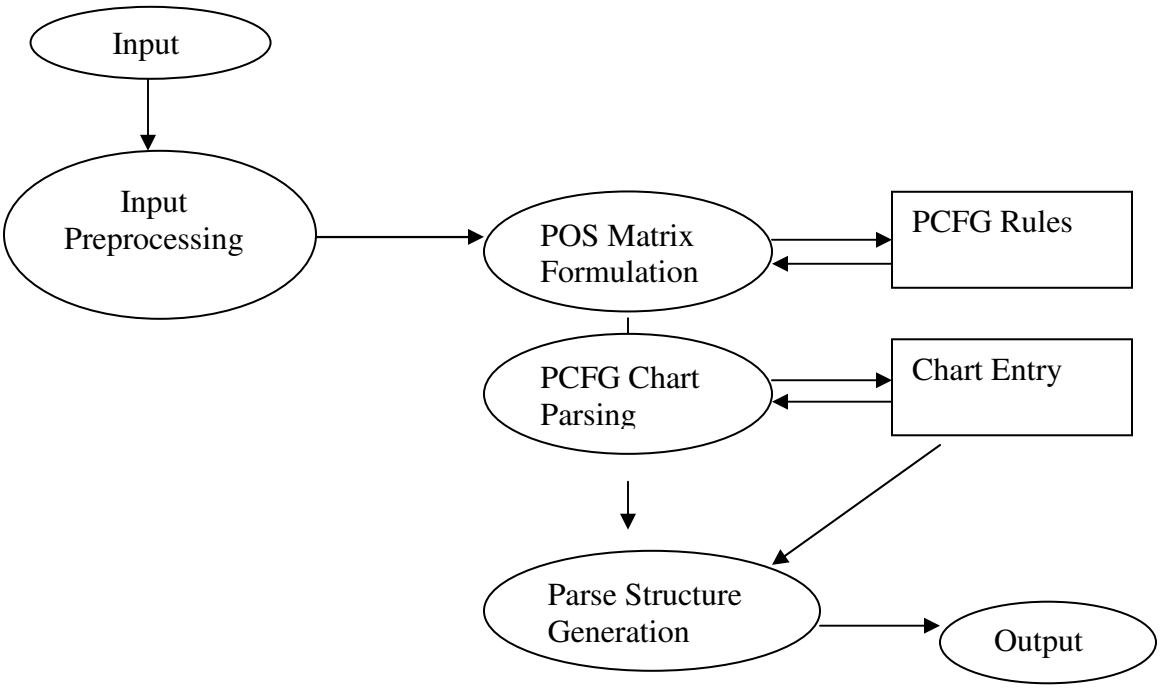

Figure 5.10 Diagrammatic Representation of the Parser

5.5 The Experiment

The sample text selected, which was discussed in chapter four, was used for experimentation. Each word in the corpora had been morphologically analyzed manually, each sentence were also hand tagged and hand parsed by two Amharic language instructors from the academic department of the Ministry of Defense and by the researcher himself, with comments and suggestions from the linguistic advisor and other experts of the language at Addis Ababa University/Institute of Language Studies and some other experts of the language. The 350 sample sentences were selected randomly based on the way they attain their complex features (i.e., being composed of a simple NP and a complex VP) and the distribution of the different phrase structures, using judgment sampling technique.

As it was made clear from the very beginning of this study, the main objective of this study was to parse Amharic complex sentences automatically using the PCFG bottom up chart parsing approach and the Inside Outside algorithm that gets its POS inputs fro ma POS tagger in which a morphological analysis is embedded. Hence, the experimentation began with checking the boost in accuracy (if any) that the POS tagger exhibited due the embedded morphological analysis.

To that end, when the tagger was first trained**,** and then run on the same data, the accuracy obtained was 86.3%**.** The errors observed were more of human made errors (errors made during the manual morphological analysis and tagging, and the preparing the lexical and transitional probability tables. After reviewing the the manually performed activities and the lexical and transitional probability calculations, and making corrections where necessary, the tagger attained 98.7% accuracy on the training set. This was higher than what the tagger attained on its original training set, i.e., 97%.

Tested with the Test Set, the tagger originally had 90% accuracy and this grew up into 94% in this study. One source of error observed was the accidental conflicting category that the morphological analysis and the bi-gram propose for a given word. This source of error was left unsolved due to lack of time. Also, errors in this phase of testing the tagger could be attributed to the small size of the sample corpus considered.

Generally speaking, the improvement on the POS tagging module (or on the input preprocessing module in general) could mainly be attributed to the use of the morphological preprocessing analysis prior to the application of the HMM tagging, the

category checking mechanism after the tagged stems were synthesized with their affixes, the statistical category guessing mechanism that fully relies on the transitional probabilities, and the slight increment in the corpus used (from 261 words in the original tagger to 883 stems in this study).

5.5.1. Experiment on The Training Set

All the manually carried out tasks, i.e., the morphological analysis, tagging and parsing as well as the probability calculations for the words in the sentences, the grammar rule induction, and probability assignment to the grammar rules, which were all discussed in chapter 4, were conducted using 280 sentences randomly picked from the sample corpus and saved as Training Set. The first experiment was conducted on these 280 sentences. One major source of error was the existence of two relatively more probable (than the other rules) rules that have the same RHS. These rules were **S➔NP VP** with a probability of 1.0 and **VP➔NP VP** with a probability of 0.524. Since the former rule has more probability than the latter, it dominates and **S** frequently appeared at a node where **VP** should have come. This was corrected by treating these two rules as rules with different RHS by labeling the two **VP**s on the RHS differently, i.e. by putting an apostrophe on the second rule's **VP** as **VP➔NP VP**'. The other errors discovered at this phase of the experiment were more of human than the parser itself. These errors were made during the manual morphological analysis, hand tagging, parsing, extraction of PCFG rules and Lexicon development, and they were reviewed making corrections where necessary.

The parser was then retrained and the test redone again on the Training Set. The final results obtained before and after such corrections were made were then reported under results of the experiment sub-section 5.5.3.1.

5.5.2 Experiment on The Test Set

The second experiment was done on the remaining 70 sentences from the original corpus and saved as Test Set. See the output of the prototype Amharic complex sentence parser for these sets in Appendix 7. The findings on this unseen part of the corpus were reported under section 5.5.3.2.

5.5.3. Results of The Experiment

The following sections discuss the results on the obtained on the experiments carried on the Training Set and Test Set. It also reports the result (before making no correction and after making corrections to the lexicon, grammar and the algorithm) on the training set.

5.5.3.1. Result on The Training Set

The result obtained when the Parser was trained and run on the same data, Training Set, is shown in the following table.

Data/set	No of sentences	No of erroneously parsed sentences	Accuracy
TrainingSet	280	163	56.1%

Table 5.3 Parsing result on Training Set before making no error correction

As one would expect the accuracy achieved should have been higher than 56.1% as the parser was trained and tested on the same data, i.e. on the Training Set. But, most of the errors at this time were as mentioned earlier emanated from the rule confusion of S➔NP VP and VP ➔ NP and VP, and this was solved considering the two rules as ones that use different RHS. The other reason for the low accuracy experiment was due to the human made errors the accuracy was not as high as it was expected. The final accuracy obtained on this set after human made errors were identified and corrected is shown in the table below.

Data/set	**No of sentences**	**No of erroneously parsed sentences**	**Accuracy**
Training Set	280	29	89.6%

Table 5.4 Parsing result on Training Set after making some error correction

5.5.3.2 Result on The Test Set

The test on the unseen part of the training corpus provided the result shown in the following table.

Data/ set	**No of sentences**	**No of erroneously parsed sentences**	**Accuracy**
Test Set	70	13	81.6%

Table 5.5 Parsing result on Test Set

The result obtained when testing the parser on the test set was approximately 82%. This experiment indicated that the accuracy level achieved using the small amount of training

set was rather acceptable. Thus, the parser developed seemed acceptable with such accuracy assuming that the manually parsed text as the point of reference.

Another test was attempted to be conducted using some 15 of the 100 simple sentences that were used by the previous study, i.e. parsing Amharic simple sentence. Although the PCFG tables of both the previous and the present studies share most of the rules, the probability values associated a given rule differs from table to table. For instance, in the case of PCFG rules extracted from the simple sentence corpus, the probability of the rule VP➔N V(0.199) is higher than VP➔V E(0.066) whereas in the PCFG rules table that was extracted from complex sentences, the probability of VP➔V E (.332) is higher than VP➔N V(0.007), which is totally a reverse case. Hence, in this test almost every sentence considered were erroneously parsed.

A third test was conducted just by embedding the morphological analysis module into the previous simple sentence parsing for Amharic to assess the relative importance of the morphological analysis module. The test was conducted on 15 sentences which included sentences that were erroneously parsed by the simple sentence parser. In addition to those sentences that were already correctly parsed by the simple sentence parser, three of the five simple sentences that were left unparsed in the previous studies were successfully parsed. One of such sentences is given below:

Simple Sentence: **b?rqyeocu yädur arawitoc täsädädu #**

Erroneous Parse In The Previous Study: S (NP (Adj **b?rqyeocu**) (N **yädur**) VP (N **arawotoc**) (V **täsädädu**)

The Correct Parse After Embedding The Morphological Analysis:

S (NP (Adj **b?rq-yeocu**) (N **yä-dur**) (N **arawit-oc**) VP(V **täsädäd-u**)).

All the 21 simple sentences that were parsed wrongly by the previous simple sentence parser were all composed of noun phrases with more than one word, as in the one given above. Hence, if the embedded morphological analysis helped the parser to correctly parse three of the five such sentences, the embedment of such a module could boost the parser's accuracy from an average of 85% to above 90%.

5.5.4 Solution to Identified Problems

Errors that were encountered during the various experimentation phases of this study start from such human errors as wrong manual morphological analysis, tagging and parsing of the data used the sample corpus. These and the other sources of errors such as miscalculation of the lexical, transitional and PCFG rule probabilities in the database were corrected through iterative approach. Besides, errors were encountered when words that did not have their stem in the database or any of their inflectional forms in the sample corpus used were mistagged or left untagged by the tagger. To deal with this untagged words, a module that was implemented by former studies to guess the word categories for untagged words, which the tagger usuaaly tags as UNC, was adopted. In this study, therefore, new words or stems were disambiguated by using bi gram lexical co occurrence probabilities. This, coupled with the morphological analysis module, solved the problem to some extent. However, since there was no absolute way of handling mis-

tagged words, the accuracy report on the Test Set was very low. The major problem of mis-parsing was the dominance of **S➔NP VP** over another rule with the same RHS but relatively lower probability, **VP➔NP VP.** This problem was alleviated by putting an apostrophe on the **VP** at the RHS of the second one and, thereby considering the two confusing rules as ones with different RHS. The inadequacy of rules at the rules library, i.e. at the PCFG table, often referred to as *under generation* and errors during the extraction of rules and their probabilities were additional sources of nuisance during the experiments, which were both minimized through iterative approach

CHAPTER SIX

CONCLUSION AND RECOMMENDATION

6.1. Conclusion

The thesis has tried to describe a way of integrating the ideas and outputs of already studied Amharic NLP systems in order to solve a bit further problem in the area – the development of an automatic complex sentence parser for Amharic. To this end, a morphological analysis, a POS tagger and a simple sentence parser for Amharic were used as the springboard. The Inside-Outside algorithm together with a chart parse module that was originally proposed by Yao and Lua (1998) in order to parse Chinese sentences, were also used to implement the Probabilistic Context Free Grammars PCFGs parsing, and efforts have also been made to develop a prototype.

Parsing is the process of discovering analyses of sentences, that is, consistent sets of relationships between constituents that are judged to hold in a given sentence, and, concurrently, what the constituents are, since constituents are typically defined inductively in terms of the relationships that hold between their parts. There are two ways in which the process of parsing sentences of such kind can be carried out: manual and automatic. The manual process is tiresome, prone to error, expensive, and obviously, the problem would worsen as the volume of information gets huge and complex. Thus, the second way of parsing sentences, automatic sentence parsing, avoids such complexities and plays important roles in Natural Language Understanding systems.

As the end goal this study was to develop an automatic complex sentence parser for Amharic, important concepts and terms in relation to parsing were reviewed and areas where the outputs of sentence parser are useful were illustrated. Besides, the rule based and the stochastic approaches, which are the major approaches to automatic sentence parsing in particular and to NLP in general, and other strategies, were briefly described and reviewed. The knowledge base of a sentence parser and the necessary components such as the lexicon and the grammatical formalisms that are needed to store information that would assist the parsing process were also discussed in some detail.

Following this, literature in the area of the Amharic writing system, lexical and grammatical categories was reviewed and discussed. This was because the knowledge of the grammar of the language is the core component in designing the parser. Thus, features of the language that were considered in designing the various components of the parser were made clear. Almost all lexical and phrasal categories, sentence formalisms, typical characteristics of complex sentences, and features of the language that were considered in designing the various components of the parser were also discussed.

Next, the developed sample corpus that was used in this study and some of the major problems the researcher faced in the process of getting the necessary sample and the steps taken to deal with the problems were presented. That is, since there is no corpus developed so far for studies on Amharic NLP, 350 complex sentences were collected from two widely used Amharic grammar books. Then, all these sentences were phonetically transcribed (due to the current lack of a standard Amharic font) in line with the IPA system, each of the words in the corpus were morphologically analyzed, tagged

and parsed manually. The lexical and transitional probabilities of each of the words in the ***Training Set***, which was a subset of the sample corpus used as a training set, were calculated. The grammar rules were extracted, the probability of each rule in the ***Training Set*** were calculated, the resulting PCFG rules were converted into Chomsky Normal Form (CNF) for simplicity, and represented in a table called PCFG-CNF. In a nut shell, a database with 7 tables was designed and/or modified to hold all the necessary lexical and grammatical information that were used in the parsing process.

The thesis then presented the algorithms and modules required by the parser to access the knowledge base and parse input sentences with appropriate lexical categories. For this purpose, an interface, which would allow a user communicate with the system was created and a prototype was developed using Visual Basic.

Experiments were conducted in two phases, one on the Training Set and the other on the Test ***Set***. Only one parameter, the percentage of correctly tagged and parsed words and sentences in the sampled text, was used to measure the performance boost of the original purely statistical part-of-speech tagger, and a simple sentence parser, and a newly developed complex sentence Amharic parser in this study. On the experiments conducted using the Training Set and Test Set to see the POS tagger's performance when the morphological analysis module was embedded, 98.7% and 94% accuracies were observed on the two sample corpus subsets, respectively. The results achieved based on the small sample were high, 89.6% on the training set and approximately 81.5% on the test set. Before achieving such accuracy, the experiment was repeatedly done on both the Training Set and the Test Set identifying errors and making corrections. Most or the

identified errors were human made errors during the preprocessing of the parser's input, conflicting PCFG rules, low probability and absence of some rules. Finally, a discussion on the possible causes of errors and their solutions was made before closing up the thesis.

Although the accuracy of the parser developed in this study was somewhat acceptable, it may not have an immediate practical application for the parser was not trained on large quantities of data that endorsed all features of Amharic (complex) sentences. To sum up, this thesis was an attempt to integrate previous NLP studies on Amharic and to use the sum total of the ideas and outputs of these works to solve a bit larger problem. It was in the mind of the researcher that Ethiopian students and researchers would strengthen such a practice that would lead towards the ultimate materialization of higher levels and more demanding research endeavors such as conceptual parsing and machine translations, which all are tasks of NLP.

6.2 Recommendations

There are many shortcomings in this research. These limitations are pointed out below and are active research areas, which should be addressed by interested individuals in the area. The efforts of those researchers might enable efforts of coming up with an efficient sentence parser for Amharic language. Thus, the following could be recommended as possible research areas.

1. In this study, the bi gram lexical co-occurrence assisted by a simple morphological analyzer, which was developed by assuming the input and output formats of a previously conducted study, was used to guess for unknown words. This technique works pretty well for the various inflections of a given stem in the database. Although this was one step forward to earlier studies, some words that were absolutely new to the database (i.e., those none of their inflectional forms exist in the database) were still tagged wrongly. Thus future researches could explore for a better result combining the tri gram lexical co-occurrence and the integrated system of the morphological analysis systems studied.

2. Future researchers can prepare corpus in which both simple and complex Amharic sentences have proportional representation, extract the PCFG rules and then test the performance of the approach and techniques used in this study in parsing both simple and complex Amharic sentences.

3. Replicate this work using a large data and incorporating all types of sentences with all attributes like case, number, gender, person, tense, and definiteness

and so on to increase the coverage of the current system in parsing various sentence types. Based on this, the performance of PCFG on the NLP for Amharic could be explored.

4. Conduct similar researches in other local languages such as Oromo language, Tigrigna, and so on by adopting the procedures followed in this study.

5. Noun Phrase recognition, conceptual parsing, word sense disambiguation and machine translations are other possible future research areas worth conducting as a continuation of a full-fledged Amharic sentence parser.

6. It could commence the need to develop processed Amharic corpus (i.e., morphologically analyzed, hand tagged, parsed, etc data) for purposes of experimentation and researches on statistical natural language processing.

7. The grammar rules extracted and the corresponding probabilities are still static values. Both the grammar induction and the corresponding probability computations could be made dynamic, i.e. to recalculate the probability values and add new grammar rules, during the parsing of sentences.

8. Integration of works that are already conducted and are being conducted in the area of NLP of Amharic is necessary in order to ensure the continuity of works and to come up with an end result. Future researchers could, therefore, focus on relatively higher levels and more demanding NLP research endeavors such as conceptual parsing and machine translations for Amharic.

BIBLIOGRAPHY

Abiyot Bayou. 2000. *Developing Automatic Word Parser for Amharic Verbs and Their Derivation, M*aster Thesis at School of Information Studies for Africa, Addis Ababa.

Allen, J. 1995. *Natural Language Understanding. **2nd Ed***. The Benjamin/ Cummings Publishing Company, Inc., California.

Atelach Alemu. 2002. *Automatic Sentence Parsing for Amharic Text*: An Experiment Using Probabilistic Context Free Grammars, Master Thesis at School of Information Studies for Africa, Addis Ababa.

Baye Yimam. 1987 (E.c). *Yamariñña Sáwasáw* (Amharic Grammar). Addis Ababa. EMPDA.

Bender, M. L., Sydney, W. H. and Roger Cowley. 1976 The Ethiopian Writing System, In Bender et al (Eds.) Language in Ethiopia. London: Oxford University Press.

Berwick, R. C. and A. S. Weinberg. 1989. *The Grammatical Basis of Linguistics Performance: Language Use and Acquisition*: London. MIT press.

Berwick, R.C. and A. S. Weinberg. 1984. The Grammatical Basis of Linguistic Performance: Language Use and Acquisition: London. The MIT Press

Biber, D., S. Conrad and R. Reppen. 1998. *Corpus Linguistics: Investigating Language Use*: New York. Cambridge University Press

Brill, E. 1993 Transformation-Based Error-Driven Parsing: Spoken Language Systems Group Laboratory for Computer Science. MIT

Briscoe, T. and Waegner, N. Robust. 1992. Stochastic Parsing Using the Inside-Outside Algorithm. Proceedings of AAAI: Workshop on Probabilistically Based Natural Language Processing Techniques. San Jose.

Cardie, C. 1993. A Case Based Approach to Knowledge Acquisition for Domain Specific Sentence Analysis. In: *Proceeding of the Eleventh National Conference on Artificial Intelligence*. AAAI press/ MIT press. Pp. 798-803.

Charniak, 2001. *Immediate head parsing for Language Models*. At: http://citeseer.nj.nec.com/384171.html

Chomsky, Naom. 1957. *Syntactic Structures*. Netherlands: Mouton & Co.

Chomsky, Naom. 1965. Aspects of the Theory of Syntax. MIT Press. Cambridge, Massachusetts,.

Dawkins, C.H. 1969 *The Fundamentals of Amharic*. A.A Sudan interior mission.

Dostert, B. H and F.B. Thompson. 1976. Syntactic Analysis in REL English. In: Papp, F. and G. Szépe. 1976. *Papers in Computational Linguistics*: Budapest. Akadémiai Kiadó

Feldman, S. 1999. *NLP meets the Jabberwocky: NLP in Information Retrieval.* At:- http:// www.onlineinc.com/onlinemag/OL1999/fieldmans.htm/

Fernando, P. 2002. *Sentence Modeling and Parsing*. At: http://clsu.cso.ogi. Edu/HLTSurvey/HLTSurvey.html

Flickinger, D. *et al*. *Natural Language Engineering—Efficient Processing with HPSG: Methods, Systems, Evaluation*. At: http://www.coli.uni-sb.de/nlesi/ Accessed at 4/15/2003

Frydenlund, M. and Sevensen, K.1998. Amharic For Beginners. Norwegian Lutheran Mission Language School. Addis Ababa.

Gazdar, Gerald; Mellish, Chris. 1996. *Natural Language Processing in Prolog.* http://www.cogs.susx.ac.uk/local/books/nlp-in-prolog/ch01/chapter-01-sh-1.6.html#sh-1.6..

Getahun Amare. 1998. Zamanawi yaamarEna Sawasaw baqalal aqaararab. Commercial Printing Press: Addis Ababa

Grishman, Ralph. 1984. Natural Language Processing. *Journal of the American Society for Information.* Vol. 35 (5): 291-296.

Grishman, Ralph. 1986. *Natural Language Processing: An Introduction*: Cambridge. University Press.

Harris, James. 1992. *Natural Language Understanding*. Reston, Virginia: Reston Publishing.

Kay, M. Functional grammar. 1979. In *Proceedings of the Fifth Annual Meeting of the Berkeley Linguistic Society*,

Lenat, Douglas B.; Guha, Ramanathan V; Pitmann, Karen; Pratt, dexter; Shepherd, Mary. 1990. CYC: Towards Programs with Common Sense. *Communication of the ACM*. V 33 (8).

Levine, R.D. and G.M. Green. (eds). 1999. *Studies in Contemporary Phrase Structure Grammar*: Cambridge. University Press

Mao, Y. .1997. *"Natural Language Processing Module (Part of Speech Tagging and Sentence Parsing) Laboratory Manual*" at http://www.csic.cornell.edu/201/natural_language/, Internet

Merlo, P. 1996. *Parsing with Principle and Classes Information*: Boston. Kluwer Academic Publishers.

Mersehazen woldeqirqos (1934) yäamarðň säwäsäw, Birhanena Selam Printing Press, Addis Ababa.

Mesfin Getachew. 2001. *Automatic Part of Speech Tagging for Amharic Language*: An Experiment Using Stochastic Hidden Markov (HMM) Approach, Master Thesis at School of Information Studies for Africa, Addis Ababa.

Molina, A., *et al.* 2002. *Incremental Partial Parser of Unrestricted Natural Language Sentences*. http://www.dsic.upv.es/~fpla/ARTICLES/snrfai99.pdf.

Prichett, B.L. 1992. *Grammatical Competence and Parsing Performance*: Chicago. The University of Chicago.

Rauch-Hindin, Wendy B. 1986. *Artificial Intelligence In Business, Science & Industry: Volume I: Fundamentals.* New Jersey: Prentice-Hall,

Reyle, U and C. Rohrer. 1988. *Natural Language Parsing and Linguistic Theories:* Boston. Reidel Publishing Company.

Rich, E. and K. Knight. 1991. *Artificial Intelligence*: New York. McGraw-Hill, Inc.

Salton, G and Michael J. McGill. 1983. Natural Language Processing. In *Introduction to Modern Information Retrieval*. New York: McGraw-Hill

Salton, G.1989. *Automatic Text Processing*: The transformation, analysis, and retrieval of Information by Computer; Wadley: Massachusetts.

Samarin, W. J. 1967. *Field Linguistics: A Guide to Linguistic Field Work*: New York. Holt, Rinehart and Winston, Inc.

Schutzer, Daniel. 1987 *Artificial Intelligence: An Applications-Oriented Approach*. New York: Van Nostrand Reinhold Company.

Shieber, S. M., Y. Schabes, and F.C.N. Pereira. 1995. Principles and Implementation of Deductive Parsing. In: *The Journal of Logic Programming*. Vol. 12. Elsevier Science Publishing Co, Inc., Pp. 1-37.

Singh, R.A. 1991. *An Introduction to Lexicography:* Mysore. Central Institute of Indian Languages.

Spark jones, & Bonnie Lynn Webber 1986. (eds.). *Readings in Natural Language Processing*. Los Altos, USA: Morgan Kaufnann.

Stolcke, Andreas. 1993. *An Efficient Probabilistic Context-Free Parsing Algorithm that Computes Prefix Probabilities* Berkley, CA.

Stuart, J. R. and P. Norvig. 1995. *Artificial Intelligence*: A Modern Approach: New Jersey. Prentice Hall

Tesfaye Bayou. 2002. Automatic Morphological Analyzer: An Experiment Using Unsupervised and Autosegmental Approach. Masters Thesis. Addis Ababa University.

Thompson, C. *et al*. 1999. Active Learning for Natural Language Parsing and Information Extraction. In: *Proceeding of the Sixth International Machine Learning Conference*. Pp. 106-114.

Warner, A.J. 1987. Natural Language Processing. In: Williams, M. E. (Ed). *Annual Review of Information Science and Technolog*y. Vol.22. Elsevier Science Publishers B.V. Pp. 79-107.

Winston, P. and P. Henry. 1984. *Artificial Intelligence*. 2nd ed: London. Addison-Wesley Publishing, Inc

Woods, William A.(1970). Transition Network Grammars of natural Language Analysis. *Communication of the ACM*. Reprinted in Barbara J. Grosz, Karen

Yao, Y. and Lua, T. 1998. A Probabilistic Context-Free Grammar Parser for Chinese. Department of Information Systems and Computer Science. National University of Singapore. http:// w.w.w.comp

APPENDICES

Appendix 1. The Amharic Alphabet, extracted from Leslau (1976)

ሀ	ሁ	ሂ	ሐ	ሄ	ህ	ሆ
hä	hu	hi	ha	he	h	ho
ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ
Lä	lu	li	la	le	l	lo
ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ
hä	hu	hi	ha	he	h	ho
መ	ሙ	ሚ	ማ	ሜ	ም	ሞ
mä	mu	mi	ma	me	m	mo
ሠ	ሡ	ሢ	ሣ	ሤ	ሥ	ሦ
sä	su	si	sa	se	s	so
ረ	ሩ	ሪ	ራ	ሬ	ር	ሮ
rä	ru	ri	ra	re	r	ro
ሰ	ሱ	ሲ	ሳ	ሴ	ስ	ሶ
sä	su	si	sa	se	s	so
ሸ	ሹ	ሺ	ሻ	ሼ	ሽ	ሾ
šä	šu	ši	ša	še	š	šo
ቀ	ቁ	ቂ	ቃ	ቄ	ቅ	ቆ
qä	qu	qi	qa	qe	q	qo
በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ
bä	bu	bi	ba	be	b	bo
ተ	ቱ	ቲ	ታ	ቴ	ት	ቶ
tä	tu	ti	ta	te	t	to
ቸ	ቹ	ቺ	ቻ	ቼ	ች	ቾ
čä	**ču**	**či**	**ča**	**če**	**č**	**čo**
ኀ	ኁ	ኂ	ኃ	ኄ	ኅ	ኆ
hä	hu	hi	ha	he	h	ho
ነ	ኑ	ኒ	ና	ኔ	ን	ኖ
nä	nu	ni	no	ne	n	no
ኘ	ኙ	ኚ	ኛ	ኜ	ኝ	ኞ
ňä	ňu	ňi	ňa	ňe	ň	ňo
አ	ኡ	ኢ	ኣ	ኤ	እ	ኦ
a	u	i	a	e	ə	o
ከ	ኩ	ኪ	ካ	ኬ	ክ	ኮ
kä	ku	ki	ka	ke	k	ko
ኸ	ኹ	ኺ	ኻ	ኼ	ኽ	ኾ
hä	hu	hi	ha	he	h	ho
ወ	ዉ	ዊ	ዋ	ዌ	ው	ዎ
wä	wu	wi	wa	we	w	wo
ዐ	ዑ	ዒ	ዓ	ዔ	ዕ	ዖ

a	u	i	a	e	ə	o
ዘ	ዙ	ዚ	ዛ	ዜ	ዝ	ዞ
zä	zu	zi	za	ze	z	zo
ዠ	ዡ	ዢ	ዣ	ዤ	ዥ	ዦ
žä	žu	ži	ža	že	ž	žo
የ	ዩ	ዪ	ያ	ዬ	ይ	ዮ
yä	yu	yi	ya	ye	y	yo
ደ	ዱ	ዲ	ዳ	ዴ	ድ	ዶ
dä	du	di	da	de	d	do
ጀ	ጁ	ጂ	ጃ	ጄ	ጅ	ጆ
ğä	ğu	ği	ğa	ğe	ğ	ğo
ገ	ጉ	ጊ	ጋ	ጌ	ግ	ጎ
gä	gu	gi	ga	ge	g	gop
ጠ	ጡ	ጢ	ጣ	ጤ	ጥ	ጦ
ṭä	ṭu	ṭe	ṭa	ṭe	ṭ	ṭo
ጨ	ጩ	ጪ	ጫ	ጬ	ጭ	ጮ
ćä	ću	ći	ća	će	ć	ćo
ጰ	ጱ	ጲ	ጳ	ጴ	ጵ	ጶ
ρä	ρu	ρi	ρa	ρe	ρ	ρo
ጸ	ጹ	ጺ	ጻ	ጼ	ጽ	ጾ
şä	şu	şi	şa	şe	ş	şo
ፀ	ፁ	ፂ	ፃ	ፄ	ፅ	ፆ
şä	şu	şi	şa	şe	ş	şo
ፈ	ፉ	ፊ	ፋ	ፌ	ፍ	ፎ
fä	fu	fi	fa	fe	f	fo
ፐ	ፑ	ፒ	ፓ	ፔ	ፕ	ፖ
pä	pu	pi	pa	pe	p	po

Appendix 2. List of Punctuation Marks In Amharic

No	Punctuation mark	Symbol	Purpose
1	The four dots (or double colon)	**::**	Marks end of a word and at the same time the end of a sentence
2	Colon	(**:**)	Separate individual words in a sentence, the
3	White space		Separate individual words in a sentence, the current practice
4	Question mark	**?**	Marks the end of an interrogative sentence
5	Exclamation	**!**	Used at the end of such sentences or interjections that express such emotions as…
6	Semi-colon		Serves roughly the same function as comma
7	Semi-colon		It separates related meanings. That is, serves roughly the same function as semicolon
8	Semi colon		It separates unrelated meanings. That is, it serves roughly the same functions as comma
9	Three dots	**…**	Marks deliberate omission of words, phrases or sentence
10	Quotation marks	“ “ ‘ ’	Used at the beginning and end of words that are being quoted
11	Parenthesis	**( )**	Encloses elaboration of Amharic meanings
12	Stroke	|	Separates date, month, year on official let’ers of an organization
13	Vertical stroke	‘	Indicates stress or gemination in Amharic.
14	Mocking mark		Placed at the end of a mocking sentence

Appendix 3. Special Tags Identified by Mesfin (2001)

No.	TAG	DESCRIPTION
1	N	Nouns including all pronouns, invariant for number, gender and case except for verbal nouns and such nouns formed using the prefix balä(e.g. l*ə*j “chiled”, l*ə*j-očč “ Children”, *ə*ssu “He” ,däg*ə*nät “kindness”)
2	NV	Verbal nouns (e.g. mäblat “Eating”, mäťäťät “ Drinking.”
3	NB	Noun formed by prefixing the prefix balä to nouns (e.g. balä-bäg “The Šheep owner”, balä-suq “Šhop keeper”, balä-l*ə*bs “owner of the clothing”)
4	NPrep	A word with a preposition not separated from a noun (e.g. b**ä**m**ä**kina “By car”, s*ə*lähägär “a bout a country”)
5	NC	A noun suffixed with a conjunction, i.e. a word with noun not separated from a conjunction (e.g. lomina b*ə*rtukan “Lemon and orange”, zäyts “how about oil”)
6	V	Verb in any form except auxiliary verbs, compound verbs and other forms of the auxiliary and compound verbs (e.g. gäddälä “He killed”, gäddlo “after he killed”, gäddäläč “še killed”)
7	AUX	Auxiliary verbs and all their other forms. This does not include compounds of allä, adärrägä, assäňä and all their other forms (e.g. alä “He, It present” ,näbbär “He, It was”, näw “It is”, nä**č “**Še is”,na**č**äw “They are”)
8	VCO	Compound verbs, i.e. compounds of allä, adärrägä, assäňä, and all their other forms (e.g b*ə*q allä “He appears”, z*ə*m assäňto “In a way of making them silent”, b*ə*d*ə*g adärrägä “He take it up.”)
9	VPreP	Any verb (main or auxiliary) headed by a preposition. The preposition not separated from the verb (e.g. s*ə*lämäťťa “Since he came”, kähedä “If he went”)
10	VC	Any verb suffixed or prefixed (i.e. headed) by a conjunction. That is, a word with the conjunction not separated from the verb (e.g. mäťťana “He come and” , s*ə*mäťťa “When he comes”, s*ə*zänb “when it rains”, *ə*skt*ə*ċärs “Until you finiš”)
11	J	An adjective which is preceded by neither prepositions nor conjunctions (e.g. däg “Kind”, k*ə*fuňa “Dangerously”, t*ə*ll*ə*k “Big”)
12	JC	An adjective not separated from a conjunction (e.g. däg*ə*na yäwah “Kind and Innocent”, ť*ə*qurna näčč*ə* “Black and white”)

13	JNU	A numeral that function as an adjective (e.g hulät b*ə*rr*ə*ċ*ə*qqo " two glasses", amm*ə*st guräno "five Šeds")
14	JPN	A preposition not separated from a noun but that function as an adjective (e.g. yäťäla b*ə*rr*ə*ċ*ə*qqo "A glass for "tella" ", yäčayna sah*ə*n "A china made plate")
15	JP	A word with a preposition not separated from the adjective. That is, the adjective is headed by a preposition (e.g. bädäh*ə*na "In a fine way")
16	PREP	A preposition that appear being not at'ached with other words (e.g. kä "From", lä "To", s*ə*lä "For Sb/Sth.", *ə*ndä "Like")
17	ADV	An adverb (e.g. tolo "In a hurry", t*ə*lant*ə*nna "Yesterday", zare "Today", hulg*ə*ze "Always")
18	ADVC	An adverb which has a conjunction suffixed to it (e.g. ahun*ə*m "Even now")
19	C	Coordinating conjunctions that appear being not at'ached with other words (e.g. nägär g*ə*n "However", wäy*ə*ss "Or")
20	REL	A word which is a relative clause (e.g. yätäsäräqäbät "one who is stolen", yäqomut "those that stand")
21	ITJ	Interjections (e.g. goš! "Wonderful", wa! "Take care! Be careful! Watch out!")
22	ORD	Ordinal number (e.g. amm*ə*stäňa "The fifth" ,ass*ə*räňa "Tenth")
23	CRD	Cardinal number (e.g. amm*ə*st "Five", ass*ə*r "Ten")
24	PUNC	Punctuation (e.g. **÷, :, !,*ə*)**
25	UNC	Unrecognized word, i.e. a word not found in the lexicon of the tagger

Appendix 4. Stem-Affix Synthesizing Table

STEMCATEGORY	SUFFIX	PREFIX	NEWCATEGORY
N	E	E	N
Adj	un	E	Adj
V	lätn	yä	comp
Adj	E	Bä	Adv
V	E	E	V
N	u	E	N
N	acäun	E	N
V	utn	yä	comp
N	ocu	E	N
Adv	E	E	Adv
V	u	E	V
V	ayn	yä	comp
V	hut	E	V
N	acäun	E	N
N	oc	E	N
N	E	bä	Adv
V	uacäu	E	V
N	itu	E	N
V	abatn	yä	comp
N	E	yä	Adj
V	ä	E	V
V	äun	yä	comp
N	n	E	N
V	äch	E	V
N	w	E	N
V	ut	E	V
N	achäu	E	N
N	yan	E	N
Adj	nät	E	N
V	alachäwn	yä	comp
V	a	E	V
N	u	bä	
V	ätn	yä	comp
N	at	E	N
V	achwn	yä	comp
V	achat	E	V
N	tua	E	N
V	at	?ndä	comp
V	achnn	E	V
N	yo	E	N
N	un	E	N
V	a	?ndä	comp

*E stands for an empty affix position.

Appendix 5. Sample Transcribed Training Set

nägadeu gäbäreu yäshät'älätn ∂h∂l wädä kätäma amät' # minist∂ru wätadärocu hagäracäun käwärarioc s∂lätadägu bät'am amäsägänu # ∂ne kassa yägäzaun näĉ bäg t∂lant ayähu # azažu wätadärocu wädä g∂biacäu ∂ndägäbu bäm∂sgana täqäbäluacäu # mäng∂st hagäritu yädäräsäbatn käf∂täňa yäekonomi kisara amänä # Abära wänd∂mu yägabäzaun gobäz yäkolleg tamari täwawäqä # n∂gusu fashistoc yätärbädäbädulätn jäg∂na säw säqälu # ∂thiopia fänt'at'a yätäwägädäbätn ∂lät t∂lant akäbäräc # prezidantu artistu bäfilm yägäläşäun b∂l∂h n∂gus adänäqu # dur∂yeoc ∂natu yälakäcun t∂n∂shj l∂j k∂fuňa däbädäbut # säratäňaocu d∂r∂j∂tu yäshälämäun kokäb säratäňa liqämänbär adärägut # anbabiyan därasiu yäsaläun ĉĉäkaň gäşäbah∂ri tä'amaninät tät'arat'ru # gazet'äňau n∂gusu bäAmerika yätädäräglacäun aqäbabäl bädänb zägäbu # mäng∂st h∂gäwät' nägadeoc s∂lätäwäsädäbacäu ∂rm∂ja mabrarya sät'ä # yä∂thiopia t'älatoc hagäritu bäd∂r∂j∂tu w∂st' yätäsät'atn bota täqawämu # g∂rmawinätacäu awrajau yägänäbun adadis täqwamat gobäňu # birou yän∂gd fäqad yätäsäräzäbacäun nägadeoc z∂rz∂r awät'a # wänbädeoc bägofäqadänoc

yägänäbutn täqwam kät’ɘqɘm wɘĉĉ adärägu # nɘfasu zälanoc ashäwa lay yäsärutn betoc ĉĉärɘso afäräsä # Asɘter ɘhɘtua bät’am sɘläwäfäräc kämät’än bälay tänadädäc # lɘjɘtu tɘllɘq wändɘmua ɘndäast’änat kɘfɘl wɘst’ nägäräcn # hakimu lɘjua asĉägari yähonäun setyo bäqɘdmya astänagädä # lɘju abatu bäqut’a sɘlätänagärut käbet wät’a # säwyäw Bälät’ä wädä Gojjam ɘndähedä ahun nägärun #

Transcribed Test Set

lɘju ɘne betacäu tɘlant yämät’ahut säw mähonen tänagärä # ɘne yäkɘfɘl gwadäňaye yätäňacbätn alga kägoräbet täwasku # säratäňaua zenabu yäaräťäbäun lɘbs bet wɘst’ asät’ac # Wolde Kassa yäyazäun addis borsa adänäqä # astämariocu käshay bet ɘskätämäläsu dɘräs kɘfɘl qoyän # gäbäyatänocu eňa yäqomɘnbätn awlala meda wädyau mollut # wäfocu Bäqälä yäafäsäsäun sɘnde bäĉɘkola läqämu # arogitua lebaoc amna yägädälutn bɘĉäňa lɘjacäun astawäsu # şalotänaua setyo yäşälot mäşhfacäu sɘlät’äfa bät’am azänu # mistyäua balyäu wädä bet bägize sɘlägäba tädäsätäc # mängɘst dɘrq yäafänaqälacäun däha gäbäreoc qälläb akäfafälä # täqwamu yäwɘĉĉ mängist yäastämaräun mämɘhɘr bäwɘd qät’äräu # geläsäbu yäsɘkuar ĉĉärätau sɘlätät’änaqäqä mɘrtun bämäkina ĉĉanä # zäbäňau asäriu sɘläfäqädälät yämata tɘmɘhɘrt tämäzägäbä # shɘmagɘleu t’ɘrsacäu sɘläalläqä bɘzu t’ɘre mäblat aqomu # qongit särgua sɘlääräsä käguadäňaua gar shɘrgud aläc # ɘne lɘbse zare sɘlätat’bä aroge lɘbs läbäsku # polisu fätashu adänzaz ɘş yäagäňäbätn gɘläsäb wäsädä # säfäru mäbrat zare sɘlät’äfa bät’am ĉĉälämä # abatyäu lɘju andäňa sɘläwät’a yäwärq sä’at shälämäu # borsau bɘzu ɘqa bäwɘst’ sɘläyazä bät’am käbädän # tamariu guadäňau sɘläzägäyä wädä kɘfɘl bɘĉaun gäba # bɘzu säwoc Japan yäsäracun ɘqa mägzat fällägu # ɘňa astämariu yäsät’änɘn yäbet sɘra kɘfɘl säran # Seble ɘne university ɘskämät’ahubät

sa’at dəräs aĉĉawätäcn # Tolosa t’əmbirau təlant əsközorä dəräs aräqe t’ät’a # Mesfin abatu t’änkəräu əndäastämarut läaddisua fiqräñau nägärat # abat käsəra mät’ wändəm manbäb kəfəl gäb #

Appendix 6. The Tagged Stem Output

1. nägade\N gäbäre\N shät'\V ∂h∂l\N wädä\P kätäma\N amät'\V #
2. minist∂r\N wätadär\N hagär\N wärari\N tadäg\V bät'am\Adv amäsägän\V #
3. ∂ne\N kassa\N gäz\V näĉ\Adj bäg\N t∂lant\Adv ay\V #
4. azažu\N wätadärocu\N wädä\P g∂bi\N gäb\V m∂sgana\N täqäbäl\V #
5. mäng∂st\N hagär\N däräs\V käf∂täňa\Adj ekonomi\Adj kisara\N amän\V #
6. Abära\N wänd∂m\N gabäz\V gobäz\Adj kolleg\Adj tamari\N täwawäq\V #
7. n∂gus\N fashist\N tärbädäbäd\V jäg∂na\Adj säw\N säqäl\V #
8. Ethiopia\N fänt'at'a\N ĉär∂so\Adv täwägäd\V ∂lät\N t∂lant\Adv akäbär\V #
9. prezidant\N artist\N bäfilm\NPrep gäläş\V b∂l∂h\Adj n∂gus\N adänäq\V #
10. dur∂ye\N ∂nat\N lak\V t∂n∂sh\Adj l∂j\N k∂fuňa\Adv däbädäb\V #
11. säratäňa\N d∂r∂j∂t\N shäläm\V kokäb\Adj säratäňa\N liqämänbär\N adäräg\V #
12. anbabi\N därasi\N sal\V ĉĉäkaň\Adj gäşäbah∂ri\N tä'amani\N tät'arat'r\V #
13. gazet'äňa\N n∂gus\N bäAmerika\Nprep tädäräg\V aqäbabäl\N bädänb\Adv zägäb\V #
14. mäng∂st\N h∂gäwät'\Adj nägade\N täwäsäd\V ∂rm∂ja\N mabrarya\N sät'ä\V #
15. Ethiopia\Adj t'älat\N hagär\N bäd∂r∂j∂t\NPrep w∂st'\P täsät'\V bota\N täqawäm\V #

Appendix 7. PCFG generated using the 280 sentences

PCFGCNF3			
LHS	**RHS1**	**RHS2**	**Probabilities**
PP	PREP	N	0.407000005245209
PP	N	PREP	0.562999994754791
AdjP	ADV	Adj	1
NP	N	E	0.590999989509583
NP	Adj	N	0.223000004887581
NP	N	N	0.136999994516373
NP	N	NP	2.19999998807907E-02
NP	NP	N	3.59999984502792E-02
NP	Adj	NP	7.00000021606684E-03
NP	N	PREP	1.40000004321337E-02
VP	PP	V	0.199000000953674
VP	N	VP	0.125
VP	N	V	0.199000000953674
VP	NP	V	0.146999999880791
VP	ADV	V	0.109999999403954
VP	AdjP	V	3.70000004768372E-02
VP	ADV	VP	6.59999996423721E-02
VP	V	V	4.39999997615814E-02
VP	V	E	6.59999996423721E-02
VP	Adj	V	7.00000021606684E-03
S	NP	VP	1
VP	S:	VP	0.199090909835673
S:	S	COMP	1
VP	NP	VP:	0.514200095952352
VP:	PP	V	0.199126589304567
VP:	ADV	V	0.11
COMP	COMP	E	0.813

Appendix 8. Sample Parse Output

1. S (NP (N nägade-u)

VP(S'(S (NP (N gäbäre-u) VP (V shät'älätn)) (COMP yä))

VP(NP(N ∂h∂l) VP(PP(P wädä) (N kätäma) VP(V amät'a)))))

2. S (NP (N minist∂r-u)

VP(S'(S (NP (N wätadär-ocu) VP (NP (N hagär-acäun) VP (N käwärari-oc)

(V tadäg-u) (COMP s?lä))

VP (Adv bät'am) (V amäsägän-uacäu)))

3. S (NP (N ∂ne)

VP(S'(S (NP (N kassa) VP (V gäz-aun)) (COMP yä))

VP (NP (Adj näĉ) (N bäg) VP (Adv t∂lant) (V ayähu)))))

4. S (NP (N azaž-u)

VP (S'(S (NP (N wätadär-ocu) PP(P wädä) (N g∂bi-acäu) VP(V gäb-u))

(COMP ?ndä)

VP (Adv bäm∂sgana) (V täqäbäl-uacäu))))

5. S (NP (N mäng∂st)

VP(S'(S (NP (N hagär-itu) VP (V däräs-äbatn)) (COMP yä))

VP (NP(Adj käf∂täňa) NP(Adj yä-ekonomi) (N kisara) VP(V amänä))))

6. S (NP (N Abära)

VP(S'(S(NP(N wänd∂m-u) VP(V gabäz-ä)) (COMP yä))

VP(NP(Adj gobäz) NP(Adj yä-kolleg) (N tamari) VP(V täwawäq-ä))))

7. S (NP (N n∂gus-u)

VP(S'(S(NP(N fashist-oc) VP(V tärbädäbäd-ulätn)) (COMP yä))

VP(NP(Adj jägəna) (N säw) VP(V säqäl-u))))

8. S(NP(N Ethiopia

VP(S'(S(NP(N fänt'at'a) VP(Adv ĉärəso) (V täwägäd-äbätn)) (COMP yä)

VP(NP(N əlät)VP(Adv təlant) (V akäbär-äc)

9. S (NP (N prezidant-u)

VP(NP(N artist-u) VP(Adv bäfilm) (V gäläş-utn) (COMP yä))

VP(NP (Adj bələh) (N nəgus) VP (V adänäq-u))))

10. S (NP (N durəye-oc)

VP(NP(N ənat-u) VP(V lak-äc) (COMP yä)

VP(NP (Adj təntəsh) (N ləj) VP(Adv kəfuňa) (V däbädäb-ut)))))