# ADDIS ABABA UNIVERSITY

## SCHOOL OF GRADUATE STUDIES

## SCHOOL OF INFORMATION STUDIES FOR AFRICA

**AN AUTOMATIC SENTENCE PARSER FOR OROMO LANGUAGE USING SUPERVISED LEARNING TECHNIQUE**



**BY**

**DIRIBA MEGERSA**

**A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER OF SCIENCE IN INFORMATION SCIENCE**



**JUNE 2002**

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION STUDIES FOR AFRICA

# AN AUTOMATIC SENTENCE PARSER FOR OROMO LANGUAGE

By

DIRIBA MEGERSA

Name and Signature of Members of the Examining Board

Ato Getachew Jemaneh, Chairman, Examining Board _____

Ato Mesfin Getachew, Advisor _____

Ato Million Meshesha, Advisor _____

Dr. Haile Eyesus Engidashet, Advisor _____

Dr. Fiaz Hussein, External Examiner _____

# Acknowledgments

My indebtedness goes to my advisors Ato Mesfin Getachew and Million Meshesha without whom such work would have been impossible. They have been providing me constructive and invaluable comments which have brought life to this paper. Without the presence of their constructive, critical and friendly suggestions, this project would have been more demanding.

I would also like to thank my Linguistic Advisor, Dr. Haile Eyesus Engdashet, for his Commitment, constructive suggestion, constant support and encouragement. His appreciation and input to my work helped me to make my research more live.

My parents have been my backbone in terms of providing me moral and financial supports during my stay in the University.

I would like to express my deepest thanks to my friend Mr. Ersin Karabillioglu who has been on my side at every time whenever I am in need of him since 1997. He has been there to me. He has been acting as my provider, counselor and brother in those times. I never forget him. I would express my gratitude to my friend Abebaw Woubishet who technically assisted me during the coding of the algorithm.

Finally, the acknowledgement would be incomplete without extending my deepest gratitude and respect to those people who were not mentioned here but their presence and efforts have been inspiring. I thank especially my friend, Daniel Gochel and his mother.

# Abstract

*The goal of Information Retrieval has been to reduce human language complexities and as a result serve users in the most efficient way. The decisive tool in achieving such end is the Natural language Processing (NLP). NLP has many components in serving such purpose. Parsing is one of such components in NLP in improving precision and recall which is the goal of Information Retrieval Systems. Moreover, parsing is also used in the effort towards machine translation which is one of the heart of Natural Language Processing.*

*Today, different kinds of parsers have been developed for languages, which have relatively wider use nationally and/or internationally since the 1960s. Unfortunately Oromo has not captured the advantage of such system being the working language of the State Government of Oromiya, and one of the major languages in Ethiopia and Africa (Abebe 2002) for there are no systems (parsers of any sort) that parse written texts in this language. This study is, therefore, an attempt to develop a simple automatic sentence parser for Oromo language.*

*In the study, the chart algorithm was used with some modification. A module for morphological analyzer, which splits words into root form and their corresponding morpheme, was also developed in order to facilitate the preparation of texts in a file to be parsed with appropriate lexical categories. In addition, the unsupervised learning algorithm was designed to guide the parser in predicting unknown and ambiguous words in a sentence. Grammar rules, lexicon, morphological rules and contextual information were also designed on the basis of the review made on the linguistic properties of Oromo grammatical categories. This system, in fact, is the first in its kind for this language.*

*The study adopts an intelligent (Rule-Based+ learning module) approach to develop a prototype, which is a simple Oromo parser for the language.*

*The thesis, in short, describes processes of automated sentence parsing of Free Texts. That is, it is aimed at developing a prototype and conducting an experiment with it. The result obtained (95% on the training test and 88.5% on the test set) using the small manually parsed sentences encourage further research to be launched, especially with the aim of developing a full-fledged Oromo sentence parser.*

# Dedication

**This paper is dedicated to my mother, Shenkore Desta, who was unable to reap the fruit of her own.**

# Table of Contents

## Chapter One

# LIST OF TABLES AND FIGURES

## TABLES



## Figures

# Abbreviations and Symbols Used

## Symbols

{ } What is inside is an affix

[ ] inside is a phrase

" " The Translation for Oromo into English

## Abbreviations

| | |
|---|---|
| Neg. | Negative |
| D. | Determiners |
| 3rd. | 3rd person |
| Sg. | Singular |
| Pl. | Plural |
| M. | Masculine |
| F | Feminine |
| N | Noun in all forms |
| NP | a noun phrase |
| V | Verb in all forms except auxiliary, compound and all forms of auxiliary and compound verbs |
| CONJ | conjunction in any form |
| AUX | Auxiliary verbs and all their other forms |
| VP | a verb phrase |
| ADJ | An adjective |
| ADJP | an Adjective Phrase |
| P | A adposition |
| PP | a adpositional phrase |
| ADV | An adverb |
| ADVP | an adverb phrase |
| OromoText | The original Oromo text |
| ManuParsedText | Manually parsed text |
| TrainingSet | Training set |
| TestSet | Test set |
| Parsedb | Parse database |
| NUM | Number |
| GEN | Gender |
| TEN | Tense |

# Chapter One

## Introduction

### 1.1 Background of the Study

The purpose of an information retrieval system is to find the most relevant materials for the user while it reduces the least relevant ones from the available collection. The ability of such system to accomplish this achievement can be evaluated using precision and its recall (Salton and Michael 1983; Salton 1989; Brants 1997). It is usually viewed as a proposition.

Any automatic information retrieval system has uncertain blocks (i.e. it does not know what to come in from users queries), particularly since they must deal with the complexities of human language. Human languages pose all kinds of complexities that are hard to resolve without resorting to the actual meaning of a word. Ambiguity is ubiquitous unless the system can bring not only the knowledge gained from the context of the text, but also the knowledge of the real world that people carry with them (Corpus 2001).

Moreover, users are looking for an accurate answer, or a good answer, but not for all the possible right or good answers. Most researches in information retrieval report that information retrieval systems rarely retrieve more than 20% of the possibly relevant materials in the database (Feldman 1999; Mao 1997). Therefore, one can extremely be interested in any new technologies that can bring precision and recall to a logical level at the same time. This is where Natural Language Processing research comes into being. Such research strives to find how best user can exploit information retrieval system to get

information they requested from the system. Moreover, such research may reduce human language complexities in information retrieval systems.

As Feldman (1999) says Natural Language Processing (NLP) research pursues the elusive question of how the meaning of a sentence or a document can best be understood. Words like nouns, verbs, adjectives and adverbs- are the building blocks of meaning of a sentence. Thus, the understanding of these words and capturing their relationship to each other within the structure of a sentence, within a document, and within the context of what one already knows about the world is crucial. It is because these words convey the true meaning of a text.

Furthermore, NLP assumes that if one can define those patterns (the patterns of those building blocks in the structure of a sentence) and describe them to a computer, then it may be possible to teach a machine something that the computer can communicate to each other. Much of this work is based on research in Linguistics, Computational Linguistics, Computer Science and Cognitive Science (Dostert and Thompson 1976; Allen 1995; Merlo 1996; Feldman 1999). For example, as Volk (1988) states, an important goal of Computational Linguistics has been to use linguistic theory to direct the implementation of efficient NLP systems.

However, one of the greatest challenges for NLP systems is to distill a sentence down to an absolutely precise, unambiguous representation of its contents. If the problem of these formal representations of NLP can be understood and represented in a computer in

procedural terms, computer systems have the ability to generate and interpret natural language. These formal representations must be unambiguous and contain not only the words and their meaning in a sentence, but the structure of that sentence as well.

If the structure of a sentence is very important in NLP, an appropriate representation of this component of the language is of paramount. Winston and Henry (1984:293) argue that good representation is the key to success in language understanding. Furthering the argument, they say that the most commonly used representation in language understanding is the syntactic processing (parsing). As Rich and knight (1991) state, the formal representation of Natural language processing consists of three components-source language (in this paper Oromo), target representation (the output of the system developed), together with a mapping (Parser) process between elements of these components.

There are a number of works carried out on this area in different languages including English. The main purpose of all these works are to enable computers understand human languages (Warner 1987). However, as far as the knowledge of this paper is concerned, there is no research work done so far on any one of the Ethiopian languages, including Oromo, the concern of the present paper. Thus, this paper has taken the opportunity (of carrying out a research as partial fulfillment of the program) to shade light on syntactic processing.

## 1.2 Syntactic processing (Parsing)

Syntactic processing ***(Parsing,*** now on wards***)*** is the step in which a flat input sentence is converted into a hierarchical structure that corresponds to the units of meaning in a sentence. Volk (1988:6) explains syntactic parsing as "partial functions that map features to atomic features values or to syntactic categories". Broadly speaking, parsing is the delinearization of linguistic input, i.e. the use of syntax to determine the functions of words in the input sentence in order to generate a data structure that can be used to get at the meaning of the sentence (Winston and Henry 1984; Harris 1992).

The syntactic structure of a sentence indicates the way that words in the sentence are related to each other. The major objective of this phase is to transform the potentially ambiguous input phrase into unambiguous forms.

Parsing plays an important role in many Natural Language (NL) understanding systems for the following reasons: Sentence parsing (e.g. syntactic or semantic) is a task of NLP. It is used in solving basic problems (such as language comprehension and production work) that most NLP applications face (Mao, 1997). The first use of parsing is that semantic processing operates on sentence constituents. If there is no syntactic parsing step, then the semantics system must decide on its own constituents. If parsing is done, however, it contains the number of constituents that semantic can consider. It is reported that syntactic parsing is computationally less expensive than is semantic processing which may require substantial inferences (Allen 1995; Elaine 1983; Elaine and Knight 1991; Grishman 1984; 1986). Thus, syntactic parsing can play a significant role in reducing overall system complexity.

Second it is not always possible to extract the meaning of a sentence without using grammatical facts. Grammatical facts are the cornerstone in understanding syntactic processing which will later be developed into semantic processing. Thus, parsing fills this gap and makes latter semantic representation easier. Furthermore, parsing regularizes the syntactic structure that subsequent processing (i.e. semantic analysis) can be simplified if one maps the large number of possible input structures into a smaller number of structures (Stuart and Norvig, 1995; Grishman, 1984; 1986).

In the development of parsers for syntactic analysis, it is a standard practice to posit two working levels: the grammars and the algorithm, which produce the analysis of the sentence by using the grammar as the source of syntactic knowledge. Parsers that can use grammars directly are more likely to have wide coverage, and be valid to many languages, and they also constitute the most economical model of the human ability to put knowledge of language to use (Berwick and Weinberg 1984; 1989). Therefore, the postulation of a direct correspondence between the parser and theories of grammar is usually assumed as a starting point of investigation (Dostert and Thompson 1976; Merlo 1996).

By making the assumption that there exists a relation between grammar and parser, the study of Natural Language Processing (NLP) can pursue engineering and cognitive goals at the same time, since the human processor is a very efficient parser, and hence a good, and possibly enlightening model. On the other hand, engineering endeavors to solve the parsing problem as efficiently as possible. This will cast light on the study of cognition in the computational paradigm (Merlo 1996). As many researchers report, parsing system is also

useful in works related to Machine translation (Mao, 1997; Rich and Knight, 1991; Allen, 1995). Especially, the increasing use of Internet in this information age makes machine translation a pressing need. This is mainly attributed to the fact that majority of people, especially those in Ethiopia having their own native language at each region, cannot make use of the huge information available on the Internet unless translated to local languages for they are unaware and/or far from foreign languages.

Today there are a lot of parsing systems developed for various languages of the world starting from the early 1960s, including English (Dostert and Thompson 1976). The first of this kind is Earley's parser. This parser is considered as the first efficient chart parser for English language. From then on wards, there have been a lot of attempts to develop such Automatic Sentence Parsing (ASP) to the languages in the world. As Oromo is one of such languages—that should have an automatic sentence parser, as far as the knowledge of the current paper is concerned, there is no such kind of system developed so far for the language. Thus the development of such an automatic system is of some importance. Hence, this paper addresses this problem and tries to develop an automatic sentence parser for the language and shades lights for further works in this area.

Some of the research works that were considered so far are stemming by Wakshum (2000) and Text to Speech by Morka (2001) for Oromo, but no one considered sentence parsing as far as my knowledge is concerned. Hence, parsing the language's text using an automatic parsing is of paramount to the language. As far as other Ethiopian languages are concerned, Abiyot (2000) and Mesfin (2001) have worked on Amharic word parser and part of speech

tagging (POST) respectively. Considering their importance, these works are reviewed in this research work.

## 1.3 Statement of the Problem and Its Justification

As indicated earlier, Parsing systems are useful in many areas of NLP for Oromo. Moreover, parsing is the step toward the process of semantic representation. Thus, the development of automatic parsing technique would help the later development of such systems as semantic parsers, spell checker, machine translation, text extraction, and other important components in the language, ***Oromo***. The absence of syntactic parser in Oromo limits, if not completely hinders, later efforts of making computer understand Oromo. Hence, this paper will try to fill such gap in the language.

In addition, the development of automatic sentence parser for Oromo will avoid the large amount of time wasted to manually process sentences in the language to show its syntactic structure. The current Parsing system developed for this language is also used as component in many other applications. These include phrase recognition, grammatical function assignment, recognition in message extraction systems, and generation of intonation in speech production systems and many others (Mao, 1997). As a result, Oromo will benefit a lot from the development of automatic parser.

The output of this research may also be used in language teaching of the sentences and in identifying phrase categories of the language, the member of each category and the relationships that hold between categories. In this regard, the linguistic structure of sentences

in the language can be discovered easily if a parser is developed for the language. The lexical categories information is obtained using the automatic parser as a component which facilitates higher levels of analysis such as noun phrase recognition, semantic and others. Thus, the current system, not only facilitates teaching but covers the shortage of human power in the area of Oromo language, if customized and expanded.

The parser is also useful for extracting meaning from a sentence and checking the well formed-ness of a sentence, which is useful in a number of applications (such as language teaching). All these benefits of the system necessitate the development of automatic parser for Oromo to reap the fruit.

## 1.4 Objective of the Study

The general objective of this research is to explore techniques and design an automatic sentence parser for Oromo Language.

In line with the general objectives, the research is aimed at dealing with the following specific objectives.

- Review the basic word categories, phrase structure (sentence as special Phrase structure) of Oromo language with the aim of investigating patterns that allow computer representation,
- Study the type of lexicon required for Oromo Parser and designs the appropriate Knowledge base for the parser,
- Review techniques of parsing adopted for other languages,

- Study the grammar of the language and represent it in a computer,
- Select and customize a parsing algorithm which best fit for Oromo language and develop a prototype for the language, and
- Test the prototype for the parser to measure its performance.

## 1.5 Methodology

The following sections reviews the methodologies employed in designing the current parser. These methodologies include literature review in the area of both grammar and algorithms used, personal discussion made with people in the area.

### 1.5.1 Literature Review

Developing an automatic parser requires one to investigate and identify the property of Oromo words and sentences that are useful for the parser by making a research on the language in general. But, this approach might be costly and can take also long time. Thus, an alternative approach was used in investigating and identifying the properties of the language. To this end, output of other research done in the area of lexical, morphological and other properties of the Oromo language was reviewed and used. Besides this, literature in the area of parsing in particular and computational linguistics in general (e.g. algorithms and data structures used) was reviewed to better understand how a language works. Thus, the an Intelligent (hybrid of Rule-based and supervised learning) System approach to parsing was selected for the reasons indicated in chapter two, with the integration of supervised learning

algorithm to develop an intelligent system that learns and predicts when new facts are encountered.

### 1.5.2 Discussion with Experts

Discussion with my linguistic advisor, especially with linguists, was made to refine the lexicon knowledge base and the parser developed with the aim to achieve the most efficient parser. The researcher also consulted these linguists for the well formed-ness of sentences in Oromo and about the correctness of the parser. These linguists include both native and non-native Oromo speakers.

### 1.5.3 Development of the Parser

Based on the analysis of the Oromo language, a parsing algorithm was developed. The developed system was crosschecked with the linguists and an iterative improvement was carried out on the system. In designing the lexicon, consideration was given to the linguistic elements, the size and quality of the lexicon. The lexicon knowledge base was developed based on the rule of the language. Moreover, the entire lexicon included in the knowledge base was selected based on the six criteria suggested in (Samarin 1967; Lehmann 1973 and Biber et al 1998)[1] from the book entitled "Galmee Jechootaa Oromo" (1996).

Visual Basic programming environment was used to develop the prototype. This programming language is selected for the various features it provides. One of the reasons is that the language can easily be manipulated to code the algorithm.

[1] See Chapter five for detailed dicusssion of these criteria.

### 1.5.4 Testing Technique

Two types of data sets were used during the experimentation: training set and test set. The training set was used to train the system. The prototype developed was tested using selected Oromo sentences from a handout of the course in Oromo Syntax (ELOm 302). A scholar then went through the output and correction was made to any erroneously parsed words. Based on this, the algorithm was refined and tested iteratively (twenty one times). Several iterations of this (i.e. tests and modification) were conducted until the final algorithm is found to be effective. The iterative improvement was ceased when the system was seen to be no more improved.

In the testing process attempt was made to challenge the parser by creating some unusual yet grammatically acceptable sentences with the aim to find the causes of the problem and making corrections as explained earlier. Finally, a test set was used to measure the performance of the system. The overall system efficiency was tested by cross checking against human (linguists) parser to see how much of the nodes in a sentence are correctly identified. Test results were also reported in chapter five.

## 1.6 Application of Results and Beneficiaries

Automatic sentence parser is useful in many NLP systems. Researchers in the area of NLP of Oromo Language could get some advantage out of the output of this paper. Specifically, researchers interested in area of phrase recognition, conceptual parsing, machine translation, spell checker, text summarization, etc are among the top beneficiaries.

Linguists and students in the area of Oromo language could also apply the output of this research to parse Oromo sentences 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. Moreover, those who are interested in generating the syntactic structure can use it.

## 1.7 Scope of the study

The scope of this is confined to dealing with simple sentences of Oromo Texts because of resource limitations in terms of time, cost and labor. Thus the current parser parses sentences of simple types. It does not deal with complex sentences which consist of clauses as phrases in the sentences. Moreover, the paper does not deal with complete features of the selected type of sentences.

## 1.8 Limitations of the study

The parser should have been trained and tested several times (at least ten folds) on different texts of Oromo sentences to see its performance but because of time and strenuous nature of grammatical data this was not possible. Moreover, shortage of enough resource in terms of linguistic information and financial constraints on Oromo language was an acute problem during this work.

The parser is developed purely based on an Intelligent (hybrid of Rule-based and supervised learning) System approach and tagger was not included which could have been used as a preprocessor to the parser. Had there been a tagger which could be used as the input of the parser, the performance this parser might have been better than this.

In this study, the parser developed was not designed for all types of sentences for a number of reasons. The reasons of such inability could be attributed to the following points.

1. Lack of enough information on the grammar of the language tagged with its lexical category.
2. Manually parsing and generating such quantities of data is very laborious, expensive and time consuming.
3. Financial constraint to carry out such huge research
4. Scope of the thesis, which is limited to demonstrate the potential of An Intelligent (hybrid of Rule-based and supervised learning) System approach to develop an automatic sentence parser for Oromo sentences.

Apart from some limitations indicated, the research has indicated the possibility to develop an automatic sentence parser for Oromo language. That is, it appears to be feasible to develop an efficient parser for Oromo language provided that enough training data is available.

## 1.9 Organization of the Thesis

The first chapter describes statement of the problem, objective and methodology used in the paper. Chapter two presents what Automatic Sentence Parser (ASP) is, approaches to automatic Sentence Parser, issues such as Grammar rules, lexicon and a learning module for the parser and steps involved in ASP.

The third chapter discusses in detail grammatical categories in Oromo language. The two sub categories of grammatical categories: phrasal and lexical categories with most of their features are included here.

Chapter four presents the process of data preparation for the system to be developed and the manual parsing of Oromo language. It presents the lexicon and grammar rules preparation in the database based on the analysis of grammatical categories reviewed in Chapter three.

The lexicon designed, the algorithms used to develop the automatic sentence parser and the rules designed to get the Lexicon and grammar rules were treated in chapter five. The database designed to hold the knowledge base, the evaluation procedures, and the experiments made and the results achieved were also discussed in this chapter.

Chapter six concludes the thesis presenting conclusions, recommendations and by indicating some directions to future works. Appendices for this thesis are found separately in the Bibliographic Lab of the School of Information Studies for Africa [SISA]. The appendices are not compiled with this thesis due to the requirement that the thesis is limited all in all to 150 pages. These appendices provide additional information on some of the topics discussed in the different parts of this thesis and are thus compiled in the form of booklet to be used together with this thesis.

# CHAPTER TWO

# Automatic Sentence Parsing (ASP)

## 2.1 *Introduction*

Parsing, as defined by many people, 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 the step 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). The input string (or tokens) is passed to the parser token by token. For each token the parser calls the morphological analyzer, which segments words into roots and affixes according to the morphological rules of Oromo. Roots and affixes are stored in a lexicon, which consists of a set of records relating various types of linguistic information. The simplest structure, which can be built, is a parse tree, which simply records the rules 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:

Tulluun saree ajjeese. [Tullu killed a dog]

the process that takes this sentence as input and produces the output of the following kind is called parsing process.

[Tulluu]N {-n}NOM-MRK[[saree]NP [ajjeese]]VP

The symbols like N, NOM-MRK, NP and VP are used to represent nouns Subject marker, noun phrase and verb phrase respectively. 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 and expensive, as the volume of information gets huge and complex. Thus, automatic sentence parsing avoids such complexities and plays important roles in natural language understanding systems.

Parsing is a complex process because it deals with the investigation of computational and psychological issues at the same time. Viewing parsing as putting a mental grammar to use has often imposed stringent requirements on the forms of the grammars used, mostly requiring that the grammar proposed by the linguists be used directly. Were the study of grammars not viewed as the study of a mental system, there would be no need to have a precise and motivated mapping between the grammar and the parser (Merlo 1996; Reyle and Rohrer 1988).

As long as the parser recovers the same structural descriptions that are assigned by the grammar to the strings in the language, no other stricter relation would be needed. The parser is subject to time/space constraints, which are irrelevant to the grammar, and these are the only pressuring constraints to shape the architecture of the parser and the techniques (Dostert and Thompson 1976; Norvig and Russell 1995; Merlo 1996; Mao 1997).

## 2.2 Approaches to Automatic sentence parsing

Sentence parsing requires exploring a way in which that sentence could have been generated from the start symbol (usually sentence written as 'S') (Doskocs 1986; Norvig and Russell 1995; Allen 195; Elaine and Knight 1991; Merlo 1996). A lot of investigations have been made so far and still under way to explore such techniques that efficiently parse a sentence. But the attempts made so far and the techniques discovered so far to respond to this problem can be categorized, in general, in two ways: Stochastic versus Rule-based.

### 2.2.1 Stochastic Approach to Automatic Sentence Parser (ASP)

Stochastic-based parsers use probability (i.e. statistics) in analyzing the problem of parsing. The stochastic approach is based on the ideas of Bayes (Network) theorem, independent events and the Markov assumption in sentence parsing. Thus, the approach uses these ideas to determine the most likely lexical sequence of each word in a given sentence. There are two sub approaches in stochastic approach: supervised and unsupervised parsers based on the type of data they use to parse a given sentence.

There are two important information in a supervised stochastic parser. A lexicon[2] is the core component that lists each word with the entire possible lexical category for each word with its respective estimate of lexical probabilities[3]. The dictionary and the lexical generation

[2] A lexicon is a linguists term for dictionary

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

probabilities can be stored in either the same or different databases. The presence of such dictionary with/and lexical probabilities provide information on the possible lexical category of each word.

Another important information for supervised stochastic parser is a list of contextual probabilities for each lexical category. Such list of contextual probabilities, like lexical probabilities, is a table of statistics that provides contextual information. It indicates the particular lexical category that is appropriate for a particular context. For detailed treatment of this topic reader can consult Allen (1995).

One problem in developing supervised parsers is the lack of manually or automatically parsed text (corpora). This is mainly because of the fact that manual parsing is expensive and time consuming. Another problem associated with supervised parsers is that there is a need to manually parse each time the parser is applied to a new text (Brill 1996). But if pre–tagged corpora are easily available, the Hidden Markov Model (HMM) approach in particular and stochastic parsers in general can be adopted in new languages with little effort.

On the other hand, parsers developed using unsupervised stochastic technique does not require any pre-tagged[4] text in the training process. The supervised and unsupervised approaches in stochastic parsers share some similarities. These are:

[4] See Mesfin (2001) for the discussion of tagging

- Both of them assume the same underlying assumption called Hidden Markov Model (HMM)
- They use a large dictionary (that lists all possible lexical category for each entry) and inflectional information to determine the possible lexical category for words in a corpora
- Both of them involve calculation of parsers accuracy to improve performance and get better results.
- In both cases disambiguation can be achieved using statistical, hybrid or rule based approaches.

However, the unsupervised stochastic parser has some unique features that make it differ from the supervised one.

- Pre-processed corpus is not necessary during the learning processes. That is training is on an unparsed or fresh text.
- Training is more difficult for the set of state transitions used to generate the training corpus are not visible
- They use different algorithms called Baum-Welch algorithm while the supervised ones use the Viterbi algorithm.

### 2.2.2 Rule-Based approach to ASP

The Rule-based approach doesn't make any use of probabilities (or statistics) to parse a sentence. It 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, Rule-Based system learns a set of rules automatically based on a given tokens (strings) and then parses sentences following these rules. Rule-Based parser consists of different components in the process of parsing a sentence.

The major component of any parser is the grammar rules (sometimes called Rewrite rules, production rules). These are the rules that the parser consults every time it starts parsing a sentence. The grammar rules can be considered as a friendly guider to the system on what action should be and should not be taken. They are important for the parser to assign appropriate grammatical categories for each constituent[5] in a sentence. Therefore, the inclusion of such rules in the knowledge base of the parser to be developed is inevitable.

A lexicon (or dictionary) is another major component of rule-based parser. A parser requires a lexicon, which are lists of all the grammatical categories of words and phrases used in parsing process. It provides distinct coding for all classes of words having distinct grammatical behavior. The lexicon is important because as soon as the parser receives the input tokens (strings), it refers to this dictionary to parse the sentence into a syntactic tree structure. The lexicon contains a list of all possible lexical categories that the word can be assigned. For example, the word can in English has the following possible lexical categories.

Can N V MD Where N= Noun V= Verb and MD=Modal

In the sentence can/Modal we/Pronoun can/Verb the/Determiner can/Noun, for instance, the word Can has all the three lexical categories mentioned. These are indicated in the sentence

[5] Constituents are elements or member of a sentence that make a sentence, or a phrase or parts of it for that matter.

by attaching a front slash after each appearance of the word can (to serve as a separator) and following it by the grammatical categories that can have at the different positions in the sentence. Thus, the word can in the lexicon must be stored in such a way that the dictionary provides all such lexical information.

Morphological rules are also useful components in rule-based approach. The morphological rules 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. For instance, assuming the English word worked is not in the lexicon, this word will morphologically be analyzed, i.e. using the lexical rules, as work + ed and it will be parsed as VVD (an arbitrary symbol for past tense of the lexical verb for this work). In line with these morphological rules, there is also morphological analyzer which is used to strip of some affixes from a word to get the base form. The base form with the syntactic feature is then passed to the syntactic parser to assign its grammatical categories. Thus, it is possible to guess the lexical categories of unknown words based on the morphological information stored in the Knowledge base. In treating words that are unknown during the parsing process, some systems include rules pertaining to capitalization and punctuation. These rules of capitalization and punctuation are additional rules incorporated in such systems besides contextual and lexical rules. But, such information on capitalization and punctuation may or may not be useful in the parsing process depending on the language being parsed. In German, for instance, information about capitalization proves to be extremely useful in the parsing of unknown nouns. In the Rule based approach there

are two ways in which parsing can be done. These are: Top-down and Bottom-up parsing techniques (Elaine and Knight 1991).

### 2.2.3 Top-down parsing

The top-down parsing begins with the start symbol (usually a sentence) and apply the grammar rules forward until the symbols at the terminals of the tree correspond to the components of the sentence being parsed. Allen (1995) states that the state of the parse at any given time can be represented as a list of symbols that are the result of operations applied so far, called the symbol list. For example, the parser starts in the state (S), after applying the rule S → NP VP, the symbol list will be (NP VP). It then applies the rule NP → N ART; the symbol list will be (N ART 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 it matched (Allen 1995). The top-down 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. 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). Thus, such parser encounters problem with empty grammar rules.

Top-down parsing, when performed left to right, encounters trouble with rules that exhibit *left recursion* (Pritchett 1992; Marcus 1983). Left recursion arises when the first category on the RHS (Right Hand Side) of a rule is more general than the one on the LHS (Left Hand Side). Again, 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).

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

However, it has a reduplication of effort. Consider the following sentence using:

Namichi saree ajjeese. [The man killed a dog.]

The parser would rewrite (S) to (NP VP)

Then rewrite (NP) to (N ADJ VP)

(N VP)

(N ADJ VP)

Thus, the top-down parser needed to check the word ADJ twice, once for each rule. This reduplication of effort 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.

### 2.2.4 Bottom-up Parsing

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.

The bottom-up parsing involves repeating the above operation, 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 in a manner permitted by the grammar.

At this point, a dim parsing algorithm will once again combine the initial NP and V into a sentence. However, when this fails to work out, it will eventually try something else and check if the sequence NP VP NP occurs as an RHS (it does not), then if VP NP does (it does not) and, finally, whether the last NP can be subsumed under some other category (the parser have already checked this NP-as-RHS issue several times - the answer remains no). Again, the parser faces an impasse: assuming that the parse tree contains this VP leads nowhere as it has seen by exhaustively checking all the possibilities. So, it must backtrack still further, ridding itself of this VP hypothesis, as does so. After some further thrashing around, the parser arrives at the following configuration:

S → NP VP  NP → N ART  VP → NP V

For the sentence Namichi saree ajjeese.

Given the implicit order of proceeding, the next thing to check is whether NP V occurs as an RHS. It does, in the shape of the other VP rule in Oromo's grammar; that is, that which allows VP to dominate NP followed by V.

Returning to the start of the sentence, the parser checks (for the *n*th time) whether V can exhaust an RHS, and then checks if NP VP can. As previously, the answer to the latter question is yes, and so the parser can proceed to add S to the parse tree.

This S, unlike the one the parser has found previously, spans the entire string, and so the parser has a success on its hands. The parsing strategy that the parser has been walked through has, at last, found a parse. If what is wanted is the first parse tree found and no others, then the parser can halt. If, however, the parser is expected to find all the possible parsing of this string, then the parser will have to be let run a while longer. It will not find any more in this case, since this string is unambiguous with respect to the Grammar Rule, but the parser only knows that it has found an S. It cannot know at this state that this is the only spanning S to be found. To establish that, it must continue in just the manner it has already seen, chasing right to the end of every dead end until it has exhausted them all. But it would try the patience to follow it on this fruitless journey!

What has been illustrated is a kind of bottom-up parsing, as the parse tree is built from the bottom upwards. As the parser saw, bottom-up parsing involves searching through a space of alternatives, not all of which lead to a successful solution.

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) says 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. Thus, 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.

## *Chart Parsers*

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. The main idea behind this technique is that the

essence of improving parsing efficiency. As indicated by Norvig and Russell (1995:697), 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 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 intermediate results. It also maintains the record of rules that have matched but are not completed. To be more specific, for example, once it is discovered that Saree maraataan kaleechaa sun "the mad dog of yesterday" is an NP in the sentence Saree maraataan kaleechaa sun due, "The mad dog of yesterday is dead", 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[6] 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. (See the detailed description and implementation of this idea in Chapter five). But for the time being, consider the following example:

$$[0, 5, S \rightarrow NP\ VP\bullet]$$

means that an NP followed by a VP combine to make an S that spans the string from 0 to 5[7]. The symbol • in an edge separates what has been found so far from what remains to be found[8]. The numbers before the symbol S indicates the indexing of the grammatical categories. Edges with • at the end are called complete edges. The edge

$$[0, 2, S \rightarrow NP \bullet VP]$$

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[9].

Allen (1995) summarizes that chart-based parsers are more efficient than those that rely on a search because the same constituent is never constructed more than once. Furthering the comment, he says 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

---

[6] It is sometimes called edge, hence interchangeably used
[7] the numbers before the symbols indicates the indexing of the grammar rules
[8] It is because of the • that the edges often called **dotted rules**.
[9] Some authors call **incomplete edges.** In some papers( Earley 1970), edges are called states, the idea is that an incomplete edge marks an intermediate state in the process of finding a complete constituent.

algorithm used. Even if C is very small, this exponential complexity rapidly makes the algorithm unusable.

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.

### 2.2.5 Comparison of Stochastic and Rule-Based Approaches (similarities and difference)

The following points are areas where rule-based and stochastic approaches share similarities which are drawn from Brill (1994; 1996) and Prichett (1992)

- Both of them may or may not need pre-processed corpora
- Both of them use contextual information from an already set of possible lexical category to disambiguate words.
- Until recently, non of them handle unknown words in a way that is completely portable

Differences between the two approaches include the following based on Brill.

- Rule–based approach uses contextual and morphological information to deal with unknown words while stochastic approach uses probabilities to deal with such words.
- A stochastic approach has no rule-based mechanisms for disambiguation. Instead it uses only probabilities for dealing with disambiguation (Brill 1996)
- Rule-based parsers generally perform as good as or better than that of stochastic parsers. In fact; parsers developed using rule-based approach have the following advantages over parsers developed using stochastic approaches.

Parsers developed using rule-based approaches require less storage than those developed using stochastic approach. That is, it is compact. In addition, some argue that parsers developed using rule-based approach are ten times faster than the fastest stochastic parsers (Brill 1994; 1996)

Parsers developed using rule-based approaches are also reported to have better accuracy than those developed using stochastic approach. Finally, twisting or modifying rule-based parsers for the purpose of correcting errors is easy and straightforward. On the other hand, tuning statistical parsers is very difficult for it is hard to predict the effect of tuning the parameters of the system

Considering all such similarities and differences, i.e. advantages and disadvantages, it seems reasonable to select the Intelligent (hybrid of Rule-based and supervised learning) System approach to deal with automatic sentence parsing for Oromo language. Hence, the current paper employs rule-based parsers to develop an automatic sentence parser for the language under consideration.

### 2.2.6 The knowledge required by the parser

The input to the algorithm is Oromo sentence, represented as an unannotated sequence of tokens, followed by an end-of-sentence punctuation mark. For instance '*Abbabaan saree ajjeese.* ' The output consists of two objects: A fully annotated binary tree, which is the parse tree of the sentence, in tree format and table to display the information required.

### Word Dictionary (lexicon)

The system to be developed has a knowledge base to guide the parser. The main components of the parser are a lexicon of selected Oromo words represented in a Knowledge Base, a morphological analyzer to strip off the affixes, and a syntactic parser. A parser is incomplete

without the definition and design of the lexicon of Oromo. The lexicon contains information about all the different sample words of Oromo that is used in this study. It includes all the relevant information and restrictions. When a word is ambiguous, it is described by multiple entries in the lexicon, one for different use.

Because words tend to follow regular morphological patterns, many forms of words are not explicitly included in the lexicon. For example, for the verb 'deem-' (dependent root form), there are different forms of the verb—'deemte, deemuu, deeme, etc'. Thus the base form of the verb and its affixes is stored in the lexicon differently and use Oromo grammar rules to combine verb with its affixes to derive the other entries. This is, of course, based on the formalism of Head-driven Phrase Structure Grammar (HPSG).

## The Grammar Rule

The most common way to represent grammars is as a set of grammar rules. These rules capture generalizations to classify words into what are often called 'parts of speech' or grammatical categories[10]. Grammar rules underlie many linguistic theories, which in turn provide the bases for many natural language understanding systems (Flickinger 2002). The grammar of Oromo is compiled into a LR (Left to Right) table.

### *Grammar Conventions*

1. Left recursive rules are handled in the parsers

[10] The term grammatical categories are more inclusive than parts of speech. It includes both lexical categories and types of phrases. Hence grammatical category is used through out this paper. For detailed information see Sag and Wasow (1999).

2. The grammar uses the *TAB* to separate a mother node from daughter nodes, but underscores between two child nodes, i.e. XP X_YP.
3. The grammar also puts each rule on a separate line.

## 2.7 Learning Modules

The technique employed here uses a technique suggested by Eric Brill (1993), called the supervised learning algorithm which was proved to be efficient in guessing unknown words with the combination of many supplementary information like morphological rule, contextual information, phrase expansion rule. Moreover, morphological and capitalization are also used as a clue for the learning algorithm (Thompson 1999; Peng (n.d); Cardie 1993).

As concluding remark, discussions were made in this chapter on various issues like the various techniques of parsing, grammar rules, morphological and contextual information which will practically be used in chapter five to Create a Lexicon, lexical grammar rule, mapped rule and which is used to develop the parser. The next chapter discusses grammatical categories in Oromo.

### 2.7 Steps in ASP

Three things are generally essential to automatically parse a sentence using the An Intelligent (hybrid of Rule-based and supervised learning) System Approach, an approach selected for this study. These are,

- Creation of a dictionary listing of the allowable Grammatical categories for each word in the small tokens (strings)
- Writing of grammar rule in the language
- Creation of a matrix of MappedRules table

# Chapter Three

## *Oromo Phrase Structures*

### 3.1 Introduction

This chapter discusses the structure of Oromo grammatical[11] categories which are the building blocks of sentences in Oromo. But before exploring into the grammatical categories of the language, the paper begins with the general and brief discussion of the lexical categories of the language. The lexical categories are what the traditional grammar calls Parts of Speech. But since grammatical categories are more inclusive the paper inclines to using the word. And to differentiate between words and phrases, we use lexical categories to refer to individual words and non-lexical or phrasal categories to refer to types of phrases.

The lexical categories discussed in this chapter are nouns, verbs, adjectives, adverbs, adpositions and conjunctions. Pronouns are also considered separately but they fall under noun category. Other Oromo words such as interjections and numerals are also topics discussed in this chapter. To better approach the topic to be discussed in this section, the chapter begins with a brief review of the writing system and punctuation marks in Oromo.

Moreover, the analyses and discussions made in this chapter are based on the data extracted from Abebe (2002), Baye (1981; 1986; 1987[12]), Askale (1997) and Tilahun (1989), Sag and

[11] Grammatical categories are more inclusive than lexical categories. It includes both lexical and phrasal categories. To differentiate between the two, we use lexical categories to refer to part of Speech in a traditional grammar and phrasal categories to refer to types of phrases.
[12] The year is according to the Ethiopian calendar

Wasow (1999), Wakshum (2000) and Morka (2001). Further details on the subject can be obtained from these sources.

## 3.2 The Oromo Alphabet and Writing System

The Oromo writing system is a modification to Latin writing system. Thus, the language shares a lot of features with English writing except some modification. Fortunately the study gets advantage of the Oromo writing alphabets called commonly "qubee Oromo" that has been designed and used so far by the language experts in the area. The writing system of the language is straightforward which is designed based on the Latin script. Thus letters in English language are also in Oromo except the way it is written. A detailed description of Oromo Writing System can be found in any text related to the language but readers can be referred to Wakshum (2000) and Morka (2001) for the detailed discussion of the language's writing system.

## 3 .3 Punctuation Marks in Oromo

Analysis of Oromo texts reveals that different Oromo punctuation marks follow the same punctuation pattern used in English and other languages that follow Latin Writing System. The full stop (.) in statement, the question mark (?) in interrogative and the exclamation mark (!) in command and exclamatory sentences marks the end of a sentence and the comma (,) which separates listing in a sentence and the semi colon (;). Listing of ideas, concepts, names, items, etc are separated by the commas.

## 3.4 Word Categories in Oromo

As in the case with the grammatical categories of other languages, like English for example, the grammatical categories of Oromo have undergone a series of improvement in terms of its word categories and other syntactic features. Thus, the eight traditional grammars is now summarized and divided into five categories in the language. In traditional word categories, Oromo words are categorized into the following eight categories (or Grammatical Categories). These are the Noun, Verb, Adjective, Adverb, Adposition, Pronoun, Conjunction and Interjection.

Modern syntacticians such as Baye (1981, 1986, 1987;)[13] divide Oromo words into five putting pronouns and adjectives under the noun, conjunctions under adposition (pre-and postposition) categories, respectively plus adverbs, adjectives and verbs. Others like Askale (1997) put adjectives and adverbs under the same lexical categories. Whatever the case, there are five syntactic categories serving as heads in phrase construction. Thus based on the above classification, the language has five grammatical categories[14] headed[15] by five word categories. In this categorization interjections, which are "words"[16] with out syntactic functions, are not considered as Grammatical Categories[17].

The current study adopts the classification scheme developed by Baye (1981, 1986, 1987). This is because traditional classification scheme is repetitive and adds redundancy to the

[13] Date is according to the Ethiopian calendar
[14] Grammatical categories are more inclusive. They include both word and phrase categories (see (Sag and Wasow 1999)
[15] See Sag and Wasow (1999) for the discussion of Headedness in a phrase.
[16] Some people, like Baye (1999, lecture notes), do not call these as words
[17] See Baye for such categorization and why interjections are not considered as Grammatical Categories

parser to be developed by this paper. Rather a subcategorization scheme is used so that the grammar rule is compact[18] and more expressive. The following sections of this chapter explores into the discussion of the grammatical[19] categories of Oromo as a background to the tasks to be carried out in chapter four and five, which are the central and major contribution of this thesis in the area.

### 3.4.1 Categories of Nouns

The definition of nouns in Oromo is similar to the definition in other language like Amharic. Oromo nouns are words used to name or identify any of categories of things, people, places or ideas or a particular one of these entities. For this study, the Oromo noun categories consist of nouns, adjectives and pronouns. Positions occupied by words like 'hoolaa' 'sheep' are considered as the nouns positions in the following sentence. 'Hoolaan marga dheeda' (the sheep grazed grass). Moreover, two numbers are recognized in Oromo nouns: singular and plural. A singular noun is marked by zero morpheme whereas a plural noun is marked by various forms. The following are illustrative examples.

| | Singular | | Plural | | pl.marker |
|---|---|---|---|---|---|
| 1. | mana | 'house' | manneen | 'houses' | {-een} |
| 2. | sa'a | 'cattle' | saawwan | 'cattle'(pl) | {-wan} |
| 3. | ganda | 'village' | gandoota | 'villages' | {-(o)ota} |
| 4. | waaraabessa | 'hyena' | waraabeyyii | 'hyenas' | {-yyii} |
| 5. | aanaa | 'district' | aanaalee | 'districts' | {-lee} |

[18] for subcategorization scheme and its compactness, readers are referred to Sag and Wasow (1999)
[19] this term is more inclusive than word categories. See Sag and Wasow (1999).

Since nouns sometimes used as adjectives, like in the following sentences

Tulluun mana *citaa* ijaare. [Tullu built a thatched house]

adjectives are categorized under different categories. Like nouns Oromo adjectives can be either primitive or derived. Adjectives come after the nouns they describe. In the example, the adjective *guraacha* "black" in *saree guraacha* comes after the noun it modified. It is also true in the adjective case that not all words after nouns could be adjectives. For example, in *mana citaa* "thatched house", the word *citaa* "thatched" is not adjective but noun as adjective. Moreover, nouns but not adjectives occur in subject and/or object positions. On the other hand, like nouns, Oromo adjectives can be either primitive or derived.

As can be observed from the data, Oromo nouns are pluralized by suffixing various forms of suffixes, such as, {-een}, {-wan}, {-(o)ota}, {-yyii} and {-lee}. It is possible to use more than one type of different plural markers in some nouns. For instance, ***mana*** 'house' can be pluralized both as ***manneen*** or ***manoota***. Most nouns, however, prefer one plural marker to the other (Abebe 2002). The word ***sagalee*** 'sound', for example, can be pluralized by suffixing {-(o)ota} or {-lee}, but it prefers the former form to the latter. When a noun stem and a plural marking suffix combine, various morphophonological processes take place among which the last vowel of the stem drops and be compensated by gemination of the preceding consonant as in (1—5) above..

Adjectives are similar to nouns in various forms. For example, Baye (1981) concluded that both nouns and adjectives are inflected for number. He further discussed that both may be pluralized by affixing the above indicated plural maker for nouns, especially {-(o)ota}

excluding those derived adjectives which are pluralized by reduplication. For example, the noun 'mana' [house] in Oromo is pluralized by adding the suffix {-ota} and becomes 'manota' [houses] but by deleting the final vowel. The same is true for adjectives. For example, the adjective 'guraacha' [black] can be pluralized in a similar fashion like nouns and becomes 'guraachota' [the black].

The two sub categories also share similar characteristics for the inflection of gender. However, it should be noted that adjectives cannot substitute nouns in a sentence construction. Only pronouns seem to substitute for each other since they can occur in the same position in a sentence of Oromo. For example, consider the following sentences in which one is correct and the other is not.

Abbabaan barsiisaadha. [Abebe is a teacher]

Inni barsiisaadha [He is a teacher].

*Guraacha barsiisaadha. [Black is teacher], which is ungrammatical.

There are also personal pronouns which are included under this category. See the following table.

| Person | *Accusative* | *Nominative* |
|---|---|---|
| 1sg | (a)na (me) | an-i (I) |
| 1pl | Nu (Us | nu-hi/nu-ti[7] (We) |
| 2sg | Si'I (you) | Ati (you) |
| 2pl | Isin (you) | isin-ϕ (you) |

| | | |
|---|---|---|
| 3sgm | Isa (him) | in-ni (he) |
| 3sgf | Ishii (her) | ishii-n (she) |
| 3pl | Isaan (they) | isaa-ϕ (they) |

Other types of pronouns are not considered in this study because of resource factor.

## 3.4.2 Categories of Verbs

The discussion of this section is based on the information collected from Baye (1981, 1986, 1987 and Askale (1997) and Abebe (2002). These works consist of all the information required by the current study. Verbs are forms which occur in clause final positions and belong to a distinct category from that of nouns (Nominals). For example in the following sentence,

Caalaan farda *bite*[20]. [Chala bought a horse]

Leensaan *dhufte*. [Lensa has come]

Tulluun dheeraa*dha*. [Tullu is tall]

the italicized part are all verbs. Baye (1986) divides verbs into a number of sub categories based on the type of constituents they are associated with. These are intransitive, transitive, ditransitive, modals and auxiliaries verbs. The intransitive verbs are those verbs which do not take any phrase as their complement[21]. For example in the sentence ' Abbabaan furdate' (Abebe got fat), 'furdate' [got fat] is an intransitive verb which has no complement. There are also what Sag and Wasow (1999) call strictly transitive verb. These types of verbs are those which take one complement in Oromo. For example,

[20] Note that the italicized words are under focus of the respective topic and/or subtopic.
[21] The complement is constituents of a phrase, or words that expand the phrase.

inni [teechuma] $_{NP}$ *cabse* (he broke the chair)

Caalaan [mana]$_{NP}$ bite (Chala bought a house)

the NP in these two examples are complement to the verbs 'cabse' broke and 'bite' bought. Finally the third category of verbs in Oromo is what is called the Ditransitive verbs. These verbs take two complement. The complement for such verbs is usually noun phrase and adpositional phrase in Oromo.

Tulluun [abbaaf] $_{PP}$ [konkoolataa] $_{NP}$ bite.

For the detailed treatment of these subcategorizations see Baye (1986).

### 3.4.3 Categories of Adverbs

Oromo adverbs are words which are used to modify verbs. Adverbs usually precede the verbs they modify or describe. Example;

Abbabaan *kaleessa* dhufe. [Abebe came yesterday]

Saree *guraacha* [black dog]

In this example, the adverb *kaleessa* "yesterday" precedes the verb *dufe* "came" that it modifies. However, it should be noted that any word that comes before a verb is not necessarily an adverbs. For instance, in *muka cabse* "broke wood", the word *muka* "wood" precedes the verb *cabse* 'broke'. In this case the word *muka* is a noun and in turn is modified by the verb *cabse*. Hence, the verb functionally shares the feature of an adjective (modifier). There are different types of adverbs. These are adverbs of time, place, manner, frequency, degree, etc. in general, adverbs are treated as the subclass of verbs. Days of the week in Oromo language may be used also either as a noun or as an adverb.

Readers again referred to Baye (1986) for further discussion and examples.

### 3.4.4 Adpositions in Oromo

The term Adpositions[22] refers to words, which will have meaning only when they are attached or used together with other words such as nouns, verbs, pronouns and adjectives. Adpositions are characterized by having no inflectional or derivational morphology and belong to the closed system.

Adpositions can appear as

- A simple adpositions that stand alone as separate words

Examples Abbabaa *walin* "with Abebe "

*Gara* mana "to house"

- A simple adposition prefixed or attached with other words (e.g. nouns and verbs).

Example harka-*an* "by hand"

Ummata-*f* " to/for the public"

- As compound adpositions consisting of two parts, adpositional prefixes and post positions put after nouns. The postpositions can either be single adposition that stand by their own or an adposition not separated from a noun.

Examples sanduqa *gubba-rra* "Inside the box"

box (noun) a postposition "inside", postposition "over"

[22] Two types of adpositions can be recognized in Oromo depending on weather they are bound or free. See (Baye 1987) and (Askale 1997) for definitions of free and bound adpositions.

### 3.4.5 Conjunctions in Oromo

Conjunctions in Oromo are coordinating or subordinating. They coordinate words, phrases, clause and sentences. A list of Oromo coordinating conjunctions is found in (Hamiid 1995) together with a detailed discussion on such coordinating and subordinating conjunctions.

This paper adopts the current trend that conjunctions and adpositions appear in the same categories—the adposition Grammatical Categories. One problem that arises by categorizing adpositions and conjunctions into different categories is the problem pertaining to distinguish conjunctions from adpositions. The problem in distinguishing the two mainly arises from the fact that the same words are mostly used as both adpositions and conjunctions. However, in cases where it is possible to separate adpositions from conjunctions, they are parsed separately. That is when the parser is able to distinguish between the two sub categories a distinct category is given to both of them. See chapter four for how conjunctions may be parsed in this paper.

### 3.4.6 Numerals

These are words representing numbers. They can be cardinal or ordinal numbers. A list of the Oromo Cardinal numbers is found in (Tilahun 1989, Hamiid 1995). In Oromo, the ordinal numbers are formed from the cardinal numbers by suffixing the suffix *{–affaa}*.

| Example | Cardinal | gloss | ordinal | gloss |
|---|---|---|---|---|
| | *lamma* | two | lamm-*affaa* | second |
| | *shan* | five | shan-*aff*aa | fifth |

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

Example *dhiba lamma*, and "two hundred"

*Dhiba lamm-affaa* "two hundredths"

*Dhiba lammaa-fi shan* *"two hundred and five*

In Oromo, there are also numerals that indicate distribution. These numerals are called distributive numerals.

Example *lama lama* "two two"

There are also special numerals in Oromo that correspond to the English "half", "quarter" etc.

Examples of these include *walakkaa* "half" and ***siisoo*** "one third".

### 3.4.7 Interjections

Like English, Oromo has many words or phrases used to express such emotions as sudden surprise, pleasure, annoyance and so on. Such Oromo words are called interjections. These Oromo interjections can stand-alone by themselves outside a sentence or can appear any where in a sentence.

Examples *ashuu!* "wonderful!"

wayyoo

ani bade**!**

A long list of Oromo interjections is found in (Tilahun 1989; Hamiid 1995).

Based on the above lexical categories, the next section explores the types of phrases found in Oromo. The idea of headedness discussed in this chapter of this paper may indicate that the types of phrases found in the language depend on the lexical categories of the language. Moreover, (Sag and Wasow 1999; Baye 1986) divide the types of phrases based on the

lexical categories. Thus this paper will depend on this classification for the purpose of the problem under consideration but keeping the idea of headedness in mind. Moreover, this paper depends entirely on Sag and Wasow (1999) and Baye (1986) for the analysis of Oromo phrasal categories.

## 3.5 Phrasal Categories

As it is indicated above phrasal categories depend on the lexical categories of a language. They use the lexical categories as the head of their phrases. Thus all phrasal categories are hierarchical in nature. This hierarchical nature of categorization is very important for it enable us to classify feature structures in more subtle way that will allow intermediate level categories of various sorts. For example, verbs may be classified as intransitive or transitive; and transitive verbs may further be sub classified as strict transitive (those taking a direct object and nothing else) or ditransitive. Thus the hierarchical system let us talk about the properties shared by two distinct types by associating a feature or a constant with their common super type. But before talking about the phrases in Oromo, let's define the word phrase.

### 3.5.1 A Phrase

A phrase can be defined as a syntactic combination of a word with one or more other words. A phrase is constrained or restricted by two things: in terms of the constituents'[23] and the lexical categories like nouns, verbs, etc. Thus, we can determine the number of phrases by

[23] A constituent is element or member in a phrase.

the number of words. A question of how to check whether a structure is a phrase can be answered using the following four guiding principles (Baye 1987)[24]. These are:

1. If the constituents of the phrase can be moved together to another place without separation.
2. If the phrase can replaced by a pronoun (for noun phrase).
3. If one of the constituents of that phrase is missed, the meaning of that phrase will be corrupted.
4. If an insertion of other word in between that phrase affects the meaning.

Based on the type of lexical categories in Oromo, there are five phrase types in the language. They will be reviewed in the following subsections.

## 3.6 Noun phrases

A noun phrase is made of one noun and one or more other lexical categories including the noun itself. For example, in the phrase 'mana citaa [thatched house], there are two nouns which make the noun phrase: mana [house] and citaa [thatched].

Thus, noun phrase and phrases in general must meet the above criteria to be called a phrase. In the following sentence 'Abbabaan mana citaa qaba' (Abebe has owned a thatched house), 'mana citaa' [thatched house] is a noun phrase. But to check whether it is really a phrase or not, we can see the above criteria. The following arrangement is impossible for the above reasons.

A. Qaba Abbabaan mana citaa. (legal movement)

[24] The year is according to Ethiopian Calendar

B. * citaa Abbabaan mana qaba. (illegal movement because of the above reason 1 and 4.)

C. * Mana Abbabaan citaa qaba. (illegal because of rule 1&4)

the above sentences with asterisks have illegal phrase construction because of the above rules. Thus we checked that "mana citaa" is a phrasal structure.

As indicated above, nouns can appear in a number of positions, such as in the positions of the three nouns in "Abbabaan kitaaba Caalaaf bite" [Abebe bought Chala a book]. These same positions allow sequences of a noun followed by an article, as in Abbabaan kitaabicha Caalaaf kenne" [Abebe gave Chala the book.]. since the position of the article can also be filled by demonstratives (kun, sun, etc.), possessives (koo, kee, keessan, etc), or quantifiers (e.g. xiqqoo) , the more general term "Determiner" abbreviated as (D) is used.

Moreover, each constituent[25] in a phrase has its own positions and functions. For example, 'mana' and 'citaa are both constituents of the phrase ' mana citaa'. A phrase is usually headed by one word. The head word is the core component of a phrase. With out a head a phrase can't be built. On the other hand a head can stand-alone by itself. A head word can determine not only phrase type but also lexical categories. If the head is a noun, then the phrase is a noun phrase, etc (Sag and Wasow 1999; Levine and Green 1999).

The noun phrase NP as a summary may be represented as

HEAD noun

[25] Each element or member in a phrase is called constituent.

AGR PER 1st, 2nd, 3rd

NUM sg, Pl

NP has a lot of constituents in Oromo. As indicated above one of the constituents is the determiner. Consider the following example,

A) [Namni tokko]$_{NP}$ [saree ajjeese]VP ( A man killed a dog)

B) [Namichi]$_{NP}$ [saree ajjeese]$_{VP}$ (the man killed a dog)

We can see that NP consists of determiners of type articles (tokko "a" in (A) and –ichi "the" in (B)). However, the position of these determiners in Oromo is different from English in that determiners come after the nouns they modify. Not only determiners but also all modifiers for nouns come after it in the language. An NP may also consist of two nouns like 'mana citaa' [thatched house]. In Oromo the order of words especially the head word and the modifiers and specifiers are different from the word these words have in English. For example, an NP in Oromo consists of one noun word as head word and another noun plus an adjective as modifiers and specifiers like in the following example.

Mana citaa bareedaa [a beautiful thatched house].

In addition to the above developments Oromo has an NP which may appear as accusative, nominative, genitive, dative and instrumental. This type of existence of nouns in a different form for different function is called case. In the following subsection it will be reviewed in brief. However, for detailed treatment of case in Oromo, readers are referred to Abebe (2002).

### 3.6.1 Accusative and Nominative Case

The accusative case form is the basic form of nouns and pronouns in Oromo (Abebe 2002; Baye 1981; Gragg 1982; Owens 1985). This means that nouns and pronouns in direct object position do not have overt case marker, as shown in the following sentences. While the nominative case (words in their subject position form) are inflected for agreement in terms of case.

A. [tulluu – n] $_{NP}$ [***mana***]$_{NP}$ ijaar-e

" Tullu built (a) house."

B. Tulluu–n ***farda adii*** $_{(NP)}$ yaabbat-e

" Tulluu rode (a) white horse."

C. Tulluu-n ***intala-tii***$_{(NP)}$ beellam –e

" Tulluu dated the girl."

D. tulluu- n ***intala –tii gurraa- ttii*** $_{(NP)}$ beellam- e

" Tulluu dated the black girl."

E. ***nam- ni*** of jaalat -a

"Man loves himself"

F. ***fard -i*** marga dheed-e

"A horse grazed grass."

G. ***Annaan*** gadi namme

In the above example, the phrase 'Tulluu-n' is an NP as nominative case (subject case) and 'mana' [house] is an NP as accusative case. Thus in the above example one can see that NP as subject has case marker, i.e. '-n', 'ni', and 'i' while NP as object form has no case marker in a sentence.

The object NPs '***mana'*** house' (head noun) in (a), '***farda adii***'' [white horse] (a head noun and modifying adjective) in (b), '***intala-itti*** ' [the girl] and '***mucaa'***' [the boy] (head noun in (c) & (d) respectively, and ***intala-ittii gurraaa- ttii*** 'the black girl'(a head noun along with singulative marker and modifying adjective) in (e) are not all overtly marked for accusative case.

Similarly, personal pronouns ***(a)na*** 'me', ***nu*** 'us', ***si'i*** 'you' (second person singular), ***isin*** 'you' (2pl), ***isa*** 'him', ***ishii*** 'her', ***isaan*** 'them' in object position do not affix accusative case marker. Some of them are shown in the following.

It can be noted from (a-f) in the above examples that nominative case in Oromo nouns is marked by '-ni', '-i', '-n' and 'ϕ' (empty set). Abebe (2002) generalized these subject markers as in the following case.

i. '-ni' occurs after a noun which ends with a short vowel that is dropped, (e.g. in E above),

ii. '–i' occurs after a noun that ends with a short vowel which is preceded by consonant cluster; the short vowel of the stem is again deleted (e.g. F),

iii. '-n' occurs after a noun that ends with a long vowel(e.g. A—D), and

iv. 'ϕ' occurs after a noun that ends with a consonant (e.g. G).

An adjective modifying a head noun in external argument position attaches similar suffixes as the nouns in the above examples for subject markers.

A. ***nam-ni furdaa-n*** dhibee hin-danda'-u

"A fat man cannot resist disease"

(b) [***fard-i gurraach-I***] $_{NP}$ collee -dha

" Black horse is smart"

As can be observed in (A and B), subject marker in the nominative case is copied onto the modifying adjectives. The forms of the nominative marking suffixes on the adjectives are phonologically conditioned in the same way as they are made on nouns. Detailed treatment of case in Oromo is found in Abebe (2002). The same is true for personal pronouns in Oromo.

There are personal pronouns such as ***nu*** 'us' and ***si'i*** 'you' which do not seem to fit into the rules in above. ***Si'i*** 'you (accusative or object case)' and ***ati*** 'you (Nominative) are different forms from one another and may be considered suppletive. On the other hand, ***nu*** 'us' and ***nuhi/nuti*** 'we' share some common phonetic form that have been summarized from above in the above rules.

In general NP constituents are

1. ***A noun as head word***
2. ***Specifiers like adjectives, adposition, etc***
3. ***Quantifiers like numbers***

Furthermore, Oromo simple noun phrase is head final. A more detailed discussion of noun phrase will be presented in the sub topic 'Sentence in Oromo'. The last point to make about Oromo Sentence is that it has discontinuous morpheme to indicate negative markers. For example, Abbabaan ***hin***ddhuf***n***e

### .7 Verb Phrases

Before the discussion of the verb phrases (VP), it is necessary to introduce the word complement here. In a simple language, a complement is a word or a phrase that the head word can take as its constituents to make it grammatical. Some words do not take complements like Abbabaan dhufe 'Abebe came"; some take one complement like 'Abbabaan muka cabse'; and still others take two complements 'Abbabaan konkoolataa naaf bite'. Based on this definition, Oromo verb phrases can be captured by dividing them into three categories. These are intransitive, strictly transitive and ditransitive verbs[26].

Abbabaan dhufe (Abebe came).

Abbabaan teechuma cabse (Abebe broke chair)

Abbabaan konkoolataa naaf bite (Abebe bought me a car).

The structures which appear as constituents of VP are all types of adverbs, adpositional phrase and noun phrase. A more detailed description will be presented in the sub topic of "Sentences in Oromo".

## 3.8 Adjective Phrase

Adjectives are specifiers for noun phrase. They usually come after the noun (usually the head word) they specify. For example, 'mana guddaa (big house). In adjectives nouns can act as adjectives like 'mana citaa' [thatched house] or verbs as adjectives like 'mana gubate' [burnt house]. Again a further detail is in the next subtopic "Sentences in Oromo".

[26] See Sag and Wasow (1999) and any other textbook on the classification of verbs in such a way.

## 3.9 Adverb Phrase

Adverb phrase is made up one adverb as head word and one or more other lexical categories including adverbs itself as modifiers and specifiers in Oromo. For example, in Oromo it is possible to have two adverbs in adverb phrase like in a phrase 'kaleessa galgala' [yesterday night]. As indicated above adverbs and their phrases are used to modify verbs. Hence they precede verbs in a sentence. In general adverb phrase can have one adverb as head word and noun phrase, another adverb, etc as constituent. See the detailed discussion in Baye (1986).

## 3.10 Adpositional Phrase

Adpositional phrases are combination of nouns and adposition. They usually specify verb phrase. This phrasal category sometimes is called adpositional objects (Baye 1986).

A. Inni ***kara*** mana deeme [He went to the house]

B. Lammaan kophee Caaltuu-***f*** bite. [Lemma bought a pen to Chaltu].

Adpositions in a adpositional phrase can be either stand independently like in (A) or affixed to the adpositional object like in (B) above.

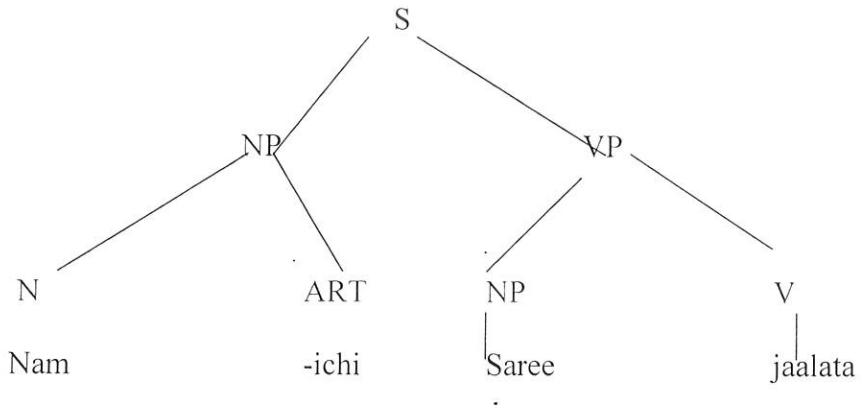
## 3.11 Sentences

The combination of zero or more Noun Phrase and one or more verb phrases make up a sentence in Oromo. However, a sentence is considered as a special kind of phrase which consists of noun phrase (NP) and the verb phrase (VP). Thus we can talk in terms of phrases when talking about sentences.

Before proceeding, we need to reflect for a moment on the traditional terminology ***parts of speech (POS)***. There are certain feature structure[27] that are appropriate for certain POS (lexical categories), but not for others. For example, CASE is appropriate only for nouns, adjectives and pronouns in Oromo. While the feature AUX(ILLARY) is specifiable only for verbs (to distinguish helping verbs from all others). Like wise, the features PER(SON) and NUM(BER) are used for nouns, verbs, and determiners. Thus, we have to guarantee in the parser to be developed that the right feature go with the right lexical categories. Moreover, it should be noted that the lexical categories discussed in this chapter are important, since they serve as the head of a phrase in a sentence of Oromo.

As discussed so far the head will always tell as its value a lexical category in Oromo. Head does the same job assigned to the POS; but it also does more here, namely it provides us a way to account which features are appropriate for which POS. Moreover, as Sag and Wasow (1999) say, HEAD enable us to introduce complex features: features within features. Furthermore, it will be of immediate use in providing us with a simple way to express the relation between a ***headed phrase*** and its ***head daughter***. To explain these terms, let's describe a tree structure which is common to almost every discipline. Any sentence can be represented in up side down tree diagram. For example, the sentence 'Namichi saree jaalata' [The man loves dog].

[27] Feature structure is a way of representing grammatical information which has a particular value. For example the feature value for gender is either male or female. It can be conceived as a directed graph where feature names label arcs that point to appropriately labeled nodes. Usually it is written in upper case.




A tree is said to consist of nodes connected by branches. A node (for e.g. S in above example) above another one (for e.g. NP or VP) in a branch is said to dominate it. The nodes at the bottom of the tree, that is, those that do not dominate any thing else, are called terminal (or leave) nodes. A node right above another node on a tree is said to be its mother node and to immediately dominate it. A node right below another one is said to be its daughter. Two daughters of the same level are sisters to one another.

Keeping the above simple definition of daughter in mind, now let's come back to the idea of ***headed phrase*** and ***head daughter***. Grammar rules in Oromo (and in any other language for that matter) require that the mother and one of the daughters bear identical (unified) values both for POS and features. The constituent on the right hand side of the rule that carries the unifying (matching) feature value is always the head daughter.

The general principle governing all trees built by headed rules as stated by Sag and Wasow (1999) say that in any headed phrase, the head value of the mother and the head value of the daughter must be unified ( or have identical value). Moreover, they state that phrase structure rules are not different in kind from word structures, except they are governed by grammar rules rather than lexical entries. Thus, we can say a sentence (or a phrase) is

grammatically correct (or well formed) based on grammar rules. Fro example, Sag and Wasow (1999) say that a sentence is well formed just in case each local sub tree within it either

I. contains a lexical entries, or

II. satisfies some grammar rules and principles.

The schematic representation of a sentence as forwarded by Sag and Wasow (1999) is:

| | | | | |
|---|---|---|---|---|
| S → PHRASE | | | PHRASE | |
| HEAD | NOUN | | HEAD | VERB |
| | AGR | | | AGR |

The structure says that a sentence consists of a noun phrase and verb phrase, noun and verb as head, unified by their agreement in a sentence, respectively (see Sag and Wasow 1999). It should be noted that this is a general representation of sentence of all types. But the purpose of this paper is to develop a parser for simple statement which describes physical or ideal; realistic or abstract ideas, actions and emotions. Such kind of sentences in Oromo ends with a period. Moreover, the paper will also include the feature of type agreement for all grammatical categories and past tense for verbs in developing parser again for the same reason.

## 3.12 Summary

As a concluding remark even though it has been discussed about lexical categories, grammatical categories including sentence, it is difficult to talk about all types of sentence in Oromo because of the limitation of time. Thus, this paper will treat the simple sentences in Oromo.

In this study pronouns are treated as nouns and hence do not have their own lexical category. This classification is entirely based on woks of scholars in the area and giving attention for the classification's conveniences for parsing. In some cases, there might be cases that may not go in line with structural linguistics. The next chapter presents lexicon, and the process of data preparation for the parsing process.

# CHAPTER FOUR

## Data Preparation for Parsing

## 4.1 Introduction

This chapter discusses the parsing process designed for Oromo language and the LEXICON identified for the purpose. The design is mainly based on the properties of the language discussed in chapter three but keeping in mind the chart based parsing discussed in chapter two.

This and the next chapters are the main contribution of this paper. This chapter focuses on drawing the lexical categories, Grammar Rules and the lexicon which are the components of the parser and the language. Problems encountered while trying to parse Oromo sentences and possible solutions tried to deal with the problems are also part of the discussion in this chapter.

The chapter begins by briefly introducing knowledge acquisition and appearances of lexicon and grammar rules in the knowledge base.

## 4.2 Knowledge Acquisition

Knowledge of the language plays an important role in order to design efficient parser for the language. Such knowledge, among other things, may be obtained in various ways. The knowledge required for the NLP in general and sentence parsing in particular can be

obtained in different ways. In this study, grammar rule, morphological rule and a lexicon obtained from the analysis of Oromo text were used as a knowledge base. However, an effort has been made to minimize the number of grammar rules of the language. This is because that the knowledge of the grammar plays a central role in developing an efficient parser. On the other hand, its size determines parsing speed. Thus, these pros and cons must be negotiated.

## 4.3 Appearance of lexical and Phrasal categories

The lexical categories in the language appear in the tree structure form. The following example illustrates the appearance or general format of a parsed sentence for the following Oromo sentence.

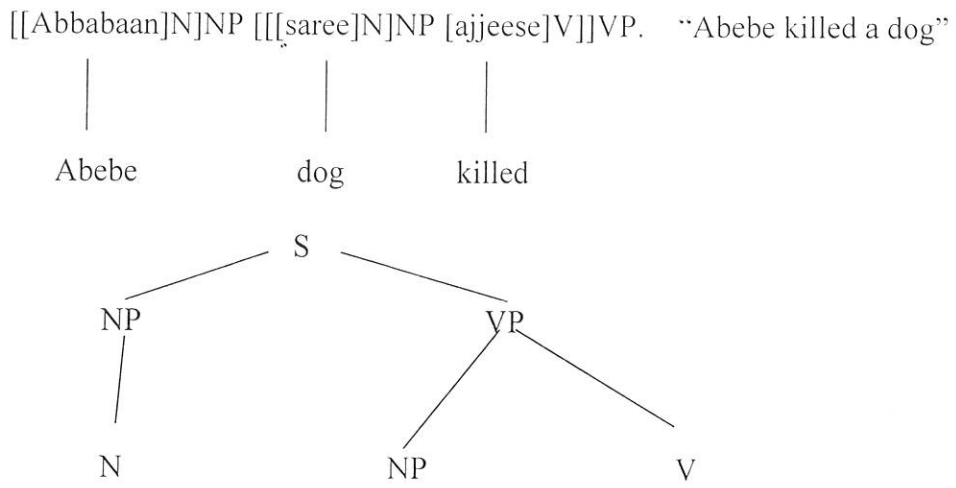



**Figure 4.1** ***Appearance of Lexical and phrasal categories during parsing***

Since it was difficult to display the constituent elements of the above sentence in particular and all sentences in this paper in general, according to their grammatical category, the following table for the output of the parser was designed. In the table, the output of the

sentence is displayed from left to right order beginning from the start of the sentence until it reaches end marks. Note that the table includes each words feature horizontally again from left to right order.

Table 4.1 *appearances of lexical categories with their features*

| Stem | Category | Num | Gen | Case | Tense | Affix Name |
|---|---|---|---|---|---|---|
| Abbabaa | N | Sg | M | NOM | | N |
| Saree | N | Sg | M | ACC | | 0 |
| Ajjees | V | Sg | M | | PAST TENSE | E |

For the sake of compactness and clarity, parses will only be assigned to words under consideration rather than the whole words appearing in the examples as illustrated in the example below, which considers discussions on nouns. The punctuation marks that mark the end of an Oromo sentence is also omitted from all examples. See figure 4.2 for the sentence "Abbabaan saree ajjeese."

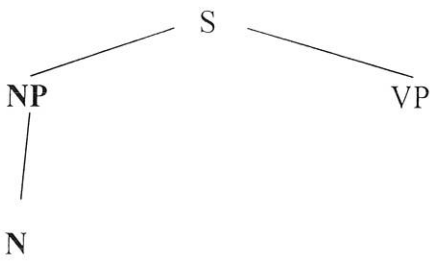



**Figure 4.2** ***Appearance of Lexical and phrasal categories of nouns in parsing***

## 4.4 Parsing of Noun Phrases

As discussed in Chapter three, noun phrases consist of a number of constituents like nouns, adjectives, specifiers, etc. Noun themselves are divided into proper[28], common, collective and pronouns. Nouns can be singular or plural. They can also indicate gender and case. In Oromo, nouns are inflected also for cases. Table 4.1 below shows the attributes of Oromo nouns together with their respective values.

**Table 4.2**: *Attribute and attribute values of nouns in Oromo*

| Lexical category: Noun | | | |
|---|---|---|---|
| Attributes | Gender | Number | Case |
| Values | {m,f} | {sg, pl} | {NOM, ACC} |

The paper does not divide nouns into different subcategories (proper, common, collective, etc). All nouns are parsed as "N". Then, their attributes in terms of number, gender, case are as shown in the above Table 4.2. A noun phrase in this work is parsed as NP (Noun Phrase) in the output of a parsed sentence. A noun phrase can be branched as **N** which becomes a head on the left and specifiers and modifiers on the right of the phrase as in the following example. The specifiers and modifiers are made of Adjectives, adverbs and numbers in Oromo.

[Namni [guraachi [tokko]SPEC]ADJP]NP sawwaan oofe.
Man black one

But for clarity reason, how each constituent of each phrasal category is parsed is discussed here in detail. But reference is always made to constituents' phrasal category where

[28] are all nouns that are different from any of the pronouns discussed in Chapter three

necessary. Thus, the discussion of phrasal category is included in each constituent element acting as a head of that phrase. For example, in a noun phrase the head word is noun. As constituent of this phrase may be adjectives, adverbs as discussed above. Hence, the parsing process for noun and its phrase is discussed in here.

Nouns in Oromo can be used as subject and/or object case. In this work the case makers of nouns are detached from nouns before they are displayed as output in the table as indicated in table 4.1. The affixes are detached using the morphological analyzer before the entry is given to the parser. Thus, the affixes are detached from nouns and then only the root form of the nouns is displayed on a table. For example, in the sentence

[[Namni]N]NP saree ajjeese "He killed a dog,"

Man dog killed

the subject marker **ni** is detached by the morphological analyzer before the noun is displayed on a table. Note that in the above example the NP has only one constituent that is N. the process of de-affixation is not only true for nouns but also for pronouns in subject (or nominative case) position in a sentence. Moreover, the process of detaching of morphemes from the input is done on each word regardless of its category before and after the parser contacts the database to check the stem's existence in the database.

The NP after being analyzed for its constituent is branched as N at the terminal node of the tree as indicated above on page 62 in figure 4.2. The NOM in table 4.1 indicates subject (nominative) case marker for Oromo nouns in this sentence and for nouns and adjectives in general.

Oromo initials in names such as obboo "Mr.", aaddee "Mrs.", etc are all are parsed as N as in the following example.

Obbo Tulluun dhufan "Mr. Tullu came."

Mr Tulluu came

Days of the week in Oromo can either be used as a noun or as an adverb. When they are used as a noun they are assigned the parse N as in

[[Kaleessi]N]$_{NP}$ [[[dafinoo]N]NP[-dha]V] $_{VP}$"Yesterday is Monday"

Yesterday Monday is

in which NP is both a phrase and a single word and thus be assigned the label **N**. But the final terminal node is the words lexical category which is displayed as output. See the following example;

"[Tulluun]NP [[saree]Np [ajjeese]V]VP."

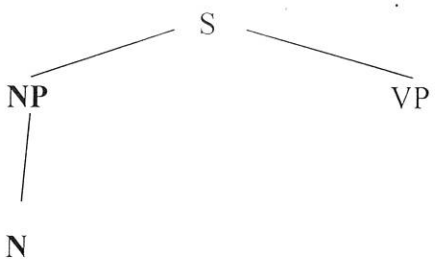



**Figure 4.3** ***how a single word as a phrase is treated in this work***

In the example above, the word "Tulluun" is a phrase but a single word. Thus, rather than parsing this directly as **N** it first assigned NP out of which **N** is branched.

Months of the year in Oromo are assigned the parse N as in

[Amajjiin]$_{NP}$ [ji'a calqabaati]$_{VP}$. January is the first month"

January month first is

There are Compound nouns having a single meaning in Oromo. Such types of nouns were difficult to treat with the current parser. So to parse such type of nouns in this work the compound words have been written by inserting a hyphen between the constituent of the compound words. Such nouns include,

[Mana baruumsaa]$_{NP}$ "School".

[Mana buna]NP "Bar".

Thus, to parse such compound nouns they are hyphenated as follows and be parsed as N.

[Mana-barumsaa]N

[Mana-buna]N

## 4 .5 Parsing of Verb Phrase

Verb phrases (VP) in Oromo have a number of constituents. The head is a verb in a verb phrase. All other modifiers and quantifiers are on the left of the verb in a sentence. For example in the following phrase

[kaleessa galgala dhufe]VP

yesterday night came

the verb is found at the end of the phrase while the words as adverb "kaleessa" and "galgala" are on the left in the phrase. Thus the output of this phrase in the following sentence looks like the following figure.

"Tulluun kaleessa galgala dhufe."

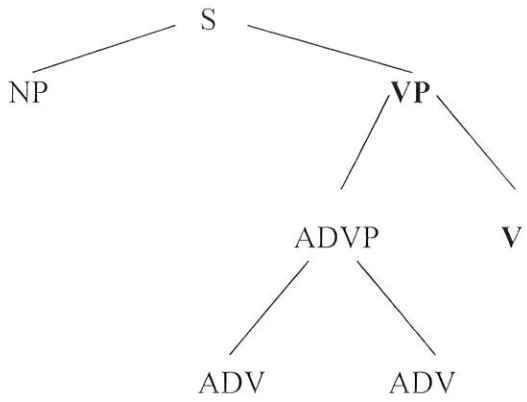



**Figure 4.4 Parser output of a verb phrase**

**Table 4.3** ***parser output for the above verb phrase***

| Stem | Category | Num | Gen | Case | Tense | Affix Name |
|---|---|---|---|---|---|---|
| Tulluu | N | SG | M | NOM | | n |
| kaleessa | ADV | | | | | 0 |
| galgala | ADV | | | | | |
| Dhuf | V | SG | M | | PAST TENSE | E |

As indicated under the sub topic of parsing of noun phrases, verb phrases are headed by verbs in that phrase. In addition, there are attributes that are unique to verbs. For example, tense is unique to verbs. The attributes and attribute values of Oromo Verbs are indicated in the table below.

**Table 4.4**: *Attributes and values of attributes for verbs*

| Category: Verb | | | |
|---|---|---|---|
| Attributes | Gender | Number | Tense |
| Values | {m, f} | {sg, pl } | {1,2,3,4,5,6[29]} |

In this study, while parsing verbs, all the above attributes are identified by the parser. That is, distinction is made between verbs based on their numbers, genders and tenses. In the following sentence the verb phrase can be displayed as in the table 4.5 in the output of the parser.

Namoonni sunniin [[dhufan]V]VP. "Those people came."

People those came

**Table 4.5.** *The output of the above sentence*

| Stem | Category | Num | Gen | Case | Tense | Affix Name |
|---|---|---|---|---|---|---|
| Nam | N | PL | | NOM | | oo, nni |
| Sunniin | ADJ | PL | | NOM | | 0 |
| Dhuf | V | PL | | | PAST | an |

As can be seen from the above table, the tense and number markers are stripped off from the verb before the output is displayed. Note that in the above example the tense marker is not visible. Gender and tense markers are visible when the verb is singular.

[29] The numbers indicate the six tenses in Oromo. For details see Baye(1986)

In this study auxiliary verbs are parsed as AUX after a verb phrase reached terminal node in a sentence. The AUX indicates auxiliary verbs in all forms. Verbs that are parsed as AUX include the following.

- The word dha "is" jira "is" qaba "have/has" and ture "was"
- All forms of dha, jira and ture.
- All forms of the negatives of dha and ture (miti "not", hinturre "was not").

When there is auxiliary verb in a sentence, verb phrase is attached to its complement. In such cases, the morphological analyzer is used before parsing process begins. After the auxiliary verb is detached from its complements by the morphological analyzer, the verb is parsed as AUX. Thus, the following sentence is parsed as

Inni [[barsiisaa]NP [dha] AUX]VP

He teacher is

It should be noted again that in parsing of V and AUX, the attributes of verbs indicated above are identified. That is, the parsing for V and AUX indicate the tenses, number, gender of verbs except for person and polarity.

In this study, compound verbs are hyphenated while writing so that the parser identifies them correctly. Such verbs include

- A verb + jira
- A verb + ture
- Infinitivals and all their other forms

Here is an example

Inni dhufee jira. "He has come."

He came has

In the above example, the phrase dhufee jira "has come" is given as dhufee-jira to the parser.

This paper does not deal with all types of tenses. It deals only with past tense. The other tense are treated (parsed) as OTHER TENSE. The inclusion of other tenses and their characteristics may avoid such problems. Moreover, such verbs can be captured if a tagger is used as a pre-processor to the parser. Moreover, the paper did not address infinitivals and all their other forms for there is a complex morphophonological process carried on them, which is not possible to handle such process with the current parser.

- Verbs other than in the past tense are parsed as V but with a tense feature of OTHER TENSE like in the following sentence.

Inni barumsa [danda'a]V "He can learn."

The tense and gender marker in verbs are not the characteristics of auxiliary verbs. Rather they take another verb like jira and ture.

## 4 .6 Parsing for Adjective Phrases

Adjective phrase in Oromo is made of an adjective as headword and other constituents like another adjective and adverb. The headword in adjective phrase is found on the left of the phrase whereas; the other constituents come on the right of the head word in the phrase. See the following for the discussion.

Namichi mucaa [guraacha kaleessa]ADJP dhaane. "The man hit yesterday's black boy."

Man boy black yesterday hit

In the above sentence, the phrase guraacha kaleessa "yesterday's black" is an adjective phrase and can be represented as the following tree format.

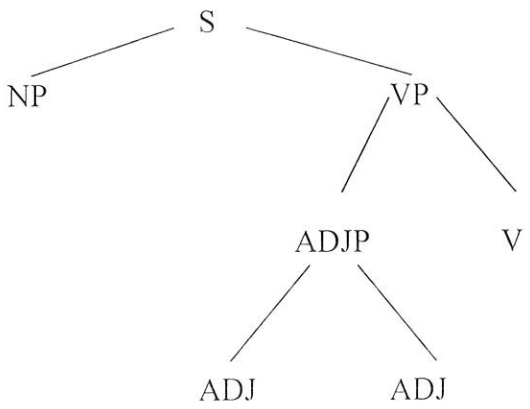



Figure 4.5 Parser output of a adjective phrase

Table 4.6 *parser output for the above adjective phrase*

| Stem | Category | Num | Gen | Case | Tense | Affix Name |
|---|---|---|---|---|---|---|
| Nam | N | SG | M | NOM | | n |
| Mucaa | N | SG | M | ACC | | 0 |
| Guraacha | ADJ | SG | M | ACC | | |
| Kaleessa | ADV | | | | | |
| Dhaan | V | SG | M | | PAST | E |

As indicated above the head word in adjective phrase is an adjective. Adjectives in Oromo have properties indicated in table 4.7.

**Table 4.7:** *Attributes and attribute values of adjectives*

| Category: Adjective | | | |
|---|---|---|---|
| Attributes | Gender | Number | Case |
| Values | {m, f} | {sg, pl} | {NOM, ACC} |

Adjectives are parsed as ADJ along with their attributes in this study. Thus, the parse ADJ is used with all its features indicated in the above table 4.7. Adjectives are parsed as ADJ whereas adjective phrase as ADJP in this work. The following sentences are some examples that illustrate parsing of ADJ and its phrase. The same is true for adjectives of having attributes indicating subject marker which is projected from the noun phrase it modifies as discussed in chapter three. Thus, the process is applied on adjectives in removing subject marker which is the same to noun phrase.

Inni nama [[guraacha]ADJ]ADJP "He is a black man."

Manni sun [[bareedaa]ADJ]ADJP -dha "His families are kind."

Some adjectives are Compound adjectives, as discussed in chapter three, i.e., they can either be attached to other words (the two words, forming a compound, word not separated) or multiple word combinations. Compound adjectives are assigned the parse ADJ once at the end of the compound adjectives (i.e. multiple word combination) rather than to each component of the compound adjectives. Compound adjectives that are not multiple word combinations are assigned the parse ADJ, like an adjective that is not compound. Here are examples on parsing compound adjectives.

Abbabaan [[harka-qallaa]ADJ]ADJP -dha "Abebe is poor."

Nitiin [[milk-dhaabeettii]ADJ]ADJP -dha "the lady is unlucky."

➢ Numerals in a sentence mostly function as an adjective. The parse ADJ is assigned for such numerals, as in the following

Baraataa [[tokko]ADJ]ADJP qofa kutaa keessa jira. "There is only one student in a class."

Namni [[lama]ADJ]ADJP walgettirratti argame. "There were two people on the meeting."

There are some adjective phrases which appear as compound words with which conjunctions attached. In such cases, both adjectives are given the parse ADJ in the adjective phrase.

## 4.7 Parsing Adpositional Phrases and their Problems

Postpositional (Adpositional) Phrase (PP) is formed from a noun and an affix –(a)f in Oromo. The word "naaf" is an adpositional phrase and {-(a)f is a morpheme used as a post position to form the phrase. In such cases, adpositions are detached from their noun by a morphological analyzer. The terminal node is then treated as noun and adposition from left to right after the morphological analyzer is applied on the phrase. The label PP is used for adpositional phrase in which the affixed adposition is used as head of that phrase. Adpositional phrase is not only made up of noun and adposition as suffix but also pronouns are also used in place of nouns. This phrase is used in indirect object position in a sentence of Oromo. Thus, detaching is carried out before it is used as input the parser.

Inni kitaaba naaf (parsed as na N af P) bite.

He book to me bought

For instance, in the sentence Tulluun kitaaba [naaf] pp bite "Tullu bought me a book", the word naaf "to me" is composed of the adposition -af "to" and the pronoun na "me". But, the whole word naaf belongs to the adpositional phrase category.

In Oromo language, there are quite a large number of words, like na-af, that are adpositional phrase. In parsing of such kind of phrases a preprocessor, the morphological analyzer, is used. This pre-processor separates the adposition af and the pronoun na, and inputs these as equivalents of na-af to the parser so that the parser parses the word na-af separating it as [na]Pro [af]P.

Adpositional phrase may have independent words which act as phrase in Oromo. Such kinds of words do not have any problem at phrase level because they can be parsed as PP (adpositional phrases). Hence, the lexical category adposition is given to the constituent which act as head of that phrase in adpositional phrase.

In this study, adpositions are parsed as P, as in the following. Moreover, such kinds of words do not have any problem at phrase level because they can be parsed as PP (adpositional phrases).

[[kara]P mana]PP deeme. "He went to his home."

However, there some problems that the parser encountered while parsing. For example, in the following example, the word waliin "with" has a final letter **n**, which conflicts with the rule of subject (nominative) case for nouns and adjectives. Thus, the parser detaches the

morpheme **-n** by treating it as nominative case marker and parses the word as adjective or noun.

Tullun [Asteer [waliin]P]PP dhufe. "Tullu came with Aster."

Tullu Aster with came.

In this study, the adpositional phrase, PP, has two forms of morpheme as adposition. These are {-af} and its other form {-f} when the final word has long vowel as in the following examples.

Hojjii na**af** barbaade. "He searched a job to me."

Inni Abbabaa**f** mana ijaare. "He built a house to Abebe."

As indicated above, in treating adpositional phrases, a morphological analyzer is used as a preprocessor to separate adposition from nouns or pronouns. Then the output of the morphological analyzer is given to the parser to determine its lexical category.

Eleemoon mana haadhaa[**f**] bite. "Elemo bought a house to his mother."

Elemo (noun) house (noun) mother(noun) to(adposition ) bought

- Some times it is difficult to separate the adposition from their host in an adpositional phrase. Thus, parsing the adposition together with the host to which it is attached with the label of a noun or pronoun headed by adposition (i.e. PP), and the adposition put after the noun separately with appropriate parsing. Here are some examples

  Inni [makiinaa[dhaan]P]PP dhufe. "He came by car"

  Ishiin [farda[an]P]PP deemte."She went on horse back."

The second adpositional phrase is very difficult to separate it from the nouns. This is because the rule given to the morphological analyzer does not handle such affixes. It was also tried to

include the rule for these two cases as special but it again conflicted with the subject marker in a sentence.

Note that, the same words are treated as adverbs by Baye (1986). Kara 'to' and so on are treated as adverbial nouns. The existence of such words representing different categories without changing their form creates a great problem to the parser. So, all such types of words are treated as adpositional phrases in this paper in all their existence at word level.

## 4.8 Parsing of Adverb Phrases

As discussed in chapter three, Oromo adverb phrases can appear as

- distinct primitive or compound adverbs appearing as one word (e.g. amma "now", daf "hurry up", kanaafuu "Therefore", har'a "Today.")
- many adverbs acting as a phrase (e.g. kana malees "in addition", xiqqo xiqqo "piece by piece" )
- compounds of adposition and other words appearing as one whole word, called adverb phrase (e.g. assirra= as(ADV) "to" + irra(adposition) "above").
- short adverbial clauses (e.g. haata'u malee inni bạrsiisaadha.)
- As gerunds used as adverbs and so on

In the first two cases, parsing is straightforward: all adverb phrases are parsed as ADVP (adverb Phrase) that appear as multi words. There is a case when a single word acts as a phrase at non-terminal level but assigned the parse ADV at the terminal node. See the following examples

inni [kaleessa galgaala]ADVP biyyaa dhufe "He came yesterday from his country."

Yesterday night

The treatment discussed under adpositions consisting of two parts is applied in the third case. This is for the purpose of consistency with earlier discussion. An adpositional phrase is headed by adpositions. Thus, instead of treating the following words as adverbs separately, they are treated as one word separated by a hyphen between each compound adverb and they are parsed as ADVP. The following illustrates this.

[Kana malees]ADVP 'however' inni kitaaba bite. "They engaged in production after the battle went off."

Short adverbial phrase (noun + P or ADJ +P ) can either be parsed as ADV or the parse for a noun or adjective headed by adposition, i.e. NP or ADJP. In this thesis, adverbial phrases are assigned the parse ADVP or ADJP depending on whether the adposition is attached with a noun or an adjective.

Inni muka [[bira]ADV]ADVP ciise "He slept near a tree."

Note that a morphological analyzer is applied on the input word of the sentence before it is given to the parser in similar way to other phrasal categories in cases when affix is attached to an adverb as in the above example. Adverbs suffixed with conjunction are also parsed as ADV, as in

[[Yeroo ammaa]ADV]ADVP qotee bulaan haala gaari-[irra]ADV jira.

"These days farmers are doing well."

Days of the week will be assigned the parse ADVP if they appear as an adverb in a sentence, as in

Inni [[Dafiinoo]ADV]ADVP dhufe "he came on Monday."

In this work, the use of ADV is applied more generally, and as such it does not differentiate among such adverbs as adverbs of time, place, manner and so on.

## 4.9 Problems with Compound Words in a phrase

In Oromo language nouns, adjectives, adverbs, verbs and conjunctions can appear as multi words. Here are some examples.

| | |
|---|---|
| Mana buna | "Bar", a noun |
| Haa ta'u malee | "However", a conjunction |
| Kana malees | "in addition" |
| Dhufee jira | "has come", verb |

The above listed words function as one word in terms of their meaning. Thus, only one category should have been assigned to such multi words. Such cases for nouns, adjectives and verbs are already considered when dealing with compound nouns, adjectives and verbs. But, there is a need to generalize such special cases. For instance, the multi word mana bunaa "bar" in the sentence Tulluun [mana buna] NP guddaa qaba. "Tullu has a big bar." can be parsed as NP phrasal level but it should not also be parsed as Tulluun [mana] N [buna] N guddaa qaba at word level. Similarly, haa ta'u malee in Inni baraataa ture [haa ta'u malee] ADV hiyyeessa should not be parsed as Inni baraataa ture [haa] ADV [ta'u] ADV [malee] ADV hiyyeessa. The following solutions were used to solve the problems at word level for every lexical category.

In this study, the words are treated according their phrasal category by putting a hyphen between each such kind of words. This is because it was difficult to parse these words with current technique.

## 4.10 Parsing of Conjunctions

Conjunctions are not treated as independent lexical category but under adpositional category and hence have no phrasal category. Thus, they are treated at word level at which they are parsed as CONJ. This is the case when the meaning of conjunction and adposition is different. In such cases parsing them differently becomes important. Efforts have been made to parse conjunctions as they are in a sentence but there are some problems. Among the problems that will be encountered while parsing conjunctions are the following.

- Most of the conjunctions are also adpositions. That is, the same word can be used as both conjunction and adposition (see conjunctions and adpositions under chapter three).

- Conjunctions may be found attached with other lexical categories (e.g. nouns, verbs, adjectives and adverbs). Here are some examples.

  Tulluun-[fi] CONJ Caalaan dhufan. "Tullu and Chala came."

  noun conjunction

In such cases, all conjunctions are parsed as CONJ after the morphological analyzer is applied on the input word to separate the conjunction from the adjective, adverb, noun or verb category. After the morphological analyzer separated the conjunction from its host, the output is given to the parser to assign the word of the output to its lexical category.

However, the morphological analyzer fails to strip the conjunction from the word it attached. Thus, in that case, the word receives the category of the host for conjunction. Moreover, conjunctions may be parsed as adposition at word level and hence accepted as it is because it is impossible to distinguish between the two at this level. Research should be carried on this issue.

- Conjunctions may appear distinctively as separate single words. Such conjunctions pose no problem and are assigned the parse CONJ, as in the following.

  Buddeena [moo]CONJ dabboo barbaadda? "Would you like injera or bread."

  Abbabaan [Yookiin]CONJ ilmi isaa har'a dhufu. "Abebe or his son will come today."

- Conjunctions may appear distinctively as multi words. Such cases may be treated as per the discussion for multi words. That is, the multi word is parsed the parse CONJ once at the end of the multi word rather than parsing the components separately after they are separated by a hyphen, as in the following. Note that these words may also be used as adverbs.

  Inni baraataadha [haa-ta'u-malee]CONJ nama isa gargaruu hinqabu. "He is a student but has no helper."

  but

- Conjunctions may appear as compound conjunctions which consist of two parts, as for instance, in

| | |
|---|---|
| [Erga] du'anii booda dhufe | "He came after their death" (erga… booda) |
| utuu isaan hinbariin qorataaman | "without their knowledge, they took the exam" (utuu… Ø) |

In such cases, the solution selected is to parse the whole phrase as one and assign the noun phrase since they are no other lexical category to which conjunction is attached.

## 4.11 Parsing of Numerals

Numerals are mostly used in the sense of an adjective or a noun. If the numerals are used in the sense of adjective, they are parsed as ADJ, as in

Namni [afur] ADJ kaleessa dhufan. "Four men came yesterday."

The parse ORD is assigned to the Oromo ordinals such as tokkoffaa "first", shanaffaa "fifth" and so on, as in

Caaltuun [shanaffaa]ORD bahe "He stood fifth."

Compound numbers such as dhibba lamaf kudhan "two hundred and ten" or kudha shanaffaa "fifteenth" may require special treatment. Two alternatives where considered in this study to handle such cases. The first alternative is to treat such numerals as multi words and hence assign only one appropriate parse form ADJ, ORD or CRD, at the end of the multi word. The second alternative is to assign each component of the numeral an appropriate parse. In the following examples, (a) refers to the first and (b) the second.

1 a. Nama [dhibba lamaaf kudhantu]ADJP dhufe. "Two hundred and ten people came."

b. Nama [dhibba]ADJ [lamaaf]ADJ kudhantu]ADJP dhufe. "Two hundred and ten people came."

From the two alternatives, the first (option) is used to parse such compound numerals by putting hyphen between each numerals for the same reason described for other lexical categories with similar problem.

Adpositions may also appear being not separated from numerals as in the examples below.

Guyyaa lamaaf hojii dhiise. "He was idle for two days."

Day for two

Guyyaa lamaan booda deeme. "He went after two days."

In such cases, since the words function as an adjective, they are assigned the same parse as numerals that function as an adjective, as in the following.

Guyyaa [lamaaf]ADJ hojii dhiise. "He was idle for two days."."

Guyyaa [lamaan]ADJ booda deeme. "He went after two days."

The case of distributive numerals, as in below may require special treatment.

Qarshii lama lama of haraka qabna. "we have two birr each."

In such cases, distributive numerals are assigned the parse CRD once at the end of the multi word for the distributive numerals are more of nouns as they indicate the English one, two, three and so on as in

Qarshii [lama]CRD [lama]CRD of haraka qabna. "We have two birr each."

## 4.12 Parsing for Interjections

According to Baye, interjections are not treated as words that are at word or phrase level, and as such they do not fall in any of the Oromo lexical categories. Thus, a special parse for them is assigned as ITJ. The following is an example of parsing interjections.

[Baga]ITJ ! dhuftan. "welcome."

## 4.13 Parsing of Punctuation

The white space and the comma are used as a delimiter to identify individual words in a sentence written in Oromo in this study. Moreover, the following marks are also used as marks which sign the end of a sentence in Oromo: the full stop (.) with declarative sentence, the question mark (?) with interrogative sentences and the exclamation mark (!) with command sentences. During parsing these punctuation marks are discarded from the parsing process since they have no importance.

## 4.14 Summary

This chapter (and the next chapter) is central to this study and has discussed the data drawn for Oromo language based on the analyses made in chapter three. But these data were enough for they included the main categories that should be addressed. These data are useful for knowledge acquisitions, i.e. to represent the structural or linguistic information contained in the NLP considered.

The next chapter discusses issues on parsing algorithms, interface design, creation of database (the knowledge base), and the actual experiment conducted.

# CHAPTER FIVE

# Sentence parsing Algorithm and the Experiment

## 5.1 Introduction

The previous Chapters presented the lexicon, the grammar rule and other information used to serve as a tool for the preparation of the parser knowledge base.

This chapter discusses the actual and major tasks involved in the process of automated parsing. Among the tasks discussed in this chapter are preparation of the knowledge base, design of the database and the interface and the development of the prototype.

Discussions in some detail are also made on the parsing algorithms, the experiments conducted and the results achieved on the small sample corpus used for the experiment. The chapter begins with the discussion on the sample text used for the experiment and the manual parsing process.

## 5.2 The Sample text and the Manual parsing Process

For the purpose of the experiment conducted, a sample narrative text of two pages consisting of 352 sentences in Oromo was obtained from "Seerluga Afaan Oromoo" by Askale (1997). This sample was selected since it was prepared with the aim of testing students' knowledge of Oromo grammar and, thus, it is comprehensive and has also a direct relevance to this study. The handout is used as a reference for the course Oromo Syntax in the Department of Ethiopian Language and Literature, Oromo Unit. The study was limited to such small

sample for no text annotated with grammatical categories was available and it was also laborious, expensive and time consuming to manually parse a large corpus. In addition, preparation of the Lexicon and Grammar Rules for large corpus requires expense and time apart from the fact that it is demanding. The sample text was given to two linguists (one native speaker of Oromo and one non-speaker). There were some differences between their parsing and thus a consensus was made to get one parsed output of the text. That output was used for the experiment.

## 5.3 Preparing the Sample Data for the Experiment

The sampled data was divided into two sets; the training set (taken from the first page of the sampled text until it reaches 85 % (300 nodes in the text) of the total sample) and the testing set (the 15% left at the end of the text). The training set consists of 300 nodes of sampled from the text selected. A sample of the text taken is found in the appendix by the name sample text. During the preparation of the experiment, the punctuation marks (. , ?) that appear in Oromo Text are avoided.

## 5.4 Evaluation Procedures

For the purpose of this research, only the percentages of correct Parse assignment to measure the performance of the Parser.

The following were the procedures followed during the experiment to evaluate the performance of the Parser.

1. The Parser was first trained on the training set obtained earlier. The Parsing algorithm, the tree construction algorithms and morphological analyzer (where necessary) were then run on the training set to see how well the Parser performs on the training set.

2. The result obtained on the training set was then evaluated by comparing it with the manually Parsed text used as a training set. Aiming to improve the Parser accuracy, causes of error were identified and corrected and step one was repeated again. These two processes repeated until the result obtained was found to be efficient, i.e. Parse assignments of the manually and automatically Parsed sentences were more or less the same.

3. The Parser was then tested on the test set, called test text, obtained earlier. This test text is essential for it helps to see the generality of the technique used. The test text help to see how well the algorithm and hence the system works in predicting the new or unknown words in a sentence.

4. The output of the Parser on the test set was then compared with the manually Parsed training set.

4. Step 4 (and also step 2 and 3 if found necessary) was repeated by identifying and correcting errors until the test set Parsed manually and automatically were fond to be almost the same.

## 5.5 Lexicon Preparation and Component Building

The actual data preparation processes carried out to design the knowledge base for the parser discussed in this section.

### 5.5.1 The Lexicon

The lexicon designed as a knowledge base is prepared by using the criteria forwarded by Singh (1991); Biber (1998); Samrin (1967) and Allen (1995). The words not in the lexicon are updated by the parser itself. Some of the criteria these people suggest are dialectical uniformity (all the data in the corpus must represent a single dialect), the corpus should consist of natural words (accepted (grammatical) word structure by the native speaker), Varied (consist of various lexical categories in the language), Complete (all the closed and functional system of the words must be included), Repetitious (repeating words that exist in different categories in the database) and interesting (a corpus which can be used again and again). Moreover, they comment on the number of the corpus in the database. In this study, attempt was made to follow these criteria while designing the lexicon.

Based on the analysis of Oromo text, the lexicon designed in figure 5.1 was prepared. This lexicon serves as a basis of evaluation of the training set designed, this is because some of the words in the lexicon are taken from the training set of the sampled text.

**Figure 5.1:** *sample Lexicon in knowledge base*

| Stem | Category | number | gender | case | tense | Source |
|---|---|---|---|---|---|---|
| dha | AUX | | | | | 0 |
| deem | V | SG | M | | PAST | |
| Abbabaa | N | SG | M | ACC | | 0 |
| Mana | N | SG | M | ACC | | 0 |
| Saree | N | SG | M | ACC | | 0 |
| Ajjees | V | SG | M | | PAST | 0 |
| furdat- | V | SG | M | | | 0 |
| dhufe | V | SG | M | | PAST | 0 |
| barsiisaa | N | SG | M | ACC | | 0 |
| nama | N | SG | M | ACC | | 0 |
| guraacha | ADJ | SG | M | ACC | | 0 |
| Bite | V | SG | M | | PAST | 0 |
| kitaaba | N | SG | M | ACC | | 0 |
| Na | N | SG | M | ACC | | 0 |
| Isa | N | SG | M | ACC | | 0 |
| Ishii | N | SG | F | ACC | | 0 |
| Ichi | ART | SG | M | | | 0 |
| Waame | V | SG | M | | PAST | 0 |
| Nyaat | V | SG | F | | PAST | 0 |

The lexicon in figure 5.1 is part of the entire lexicon; named here on Lexicon, designed for this study. From the lexicon above or the entire lexicon designed; one can see that the column provides information on the attributes of a given word in the lexicon. For each entry in the lexicon, the lexicon holds information on the features (attributes) that word may have when used in a sentence. As shown in the above table, there are five features that a word may have in a sentence. If the word 'deem' [the root form of the word ***go***] is included in the lexicon, then it has the following features: Stem, NUM(BER), GEN(DER), TEN(SE) and SOURCE.

The word SOURCE in the lexicon refers to the data source, i.e., whether the word is already in the Knowledge Base or the parser has added it as a new word into the knowledge base after predicting its lexical category in a sentence parsed. If the word lexical category is predicted by the parser and afresh added to the knowledge base then it will be assigned 1, otherwise if the word originally in the knowledge base it is assigned 0.

## 5.5.2 Grammar Rules

Based on the study of Oromo phrase structure, a grammar rule table was designed. This grammar rule table is the core component for the training set designed and then to construct tree out of the input sentences. There are two grammar tables in the database: GrammarRules and MappedRules

Table 5.2 below shows the design of the grammar rule in the database. In addition to the grammar rule in the database there is what is called Mapped Rule. The parser itself while

reading the grammar rule from the database automatically creates this mapped rule. The purpose of this mapped rule is to control Recursive rule in the main grammar rule. Thus, the format of this mapped rule is similar to the grammar rule except this rule adds additional column for ParentID, which assigns index for the parent of the rule from which this mapped rule is derived.

**FIGURE 5.2** *Sample Grammar Rule in the database*

| **LHS** | **RHS** |
|---|---|
| S | NP_VP |
| NP | N_ADJ |
| VP | NP_V |
| NP | N |

The grammar rule contains the LHS (Left Hand Side) rule which contains the non-terminal node in tree construction. On the other hand the RHS (Right Hand Side) rule contains constituents of the phrase rule on the left hand side. As pointed out earlier in chapter two, if the parser encounters rule of the following kind, it creates its own rule under the name Mapped rule to control recursiveness in the rule. The following rule is mapped to VP N_V after the parser reads the constituents of NP from the database based on the above table.

VP NP_V is changed to VP N_V

Therefore, the mapped rule helped the parser not to enter an infinite loop as a result of recursive rule. Some people (including Allen 1995) recommend to avoid (or replace with basic rule instead of derived ones) recursive rule from the grammar rule.

### 5.5.3 Affix List

Another table designed in the parsedb was the affix list table. The affix list in the database provides information on the affixes attached to a word during word's inflection for number, tense, gender, case and so on. Hence, this table contains the information of which affix is associated to which lexical category in a sentence. Moreover, it tells whether the affix attached to the word is at the initial, middle or final position of a word. Table 5.3 shows a sample of the AffixList table.

**FIGURE 5.3** *the format of Affix list table in the database.*

| AffixName | Its_Association | Its_position |
|---|---|---|
| -ota | N, ADJ | Final |
| -Oota | N, ADJ | Final |
| -E | V | Final |
| -T | V | Final |
| -N | N | Final |
| -Ni | N, ADJ | Final |

## 5.6 Database Design

To store the words in the lexicon, grammar rule, Affix list and mapped rule, a database with four tables was constructed. Figure 5.4 shows the database schema designed for the Oromo sentence parser developed as a prototype.

**Figure 5.2** *Lexicon and GrammarRule database schema*

*Lexicon*

| Stem | NUM | GEN | CASE | TEN | SOURCE |
|---|---|---|---|---|---|

*GrammarRule*

| LHS | RHS |
|---|---|
| | |

MappedRule

| LHS | RHS |
|---|---|
| | |

*Affixlist*

| Morph | ASSOCIATION | Its_Association |
|---|---|---|
| | | |

The mappedRule table was designed to control recursive grammar rules in the language on both left and right hand sides. As soon as the parser encountered any recursive rule in the GrammarRule of the database, the parser splits that recursive into its constituents without violating the grammar rule of the language. In this figure 5.2, Lexicon is a list of words along with their lexical category and all features in the database. The table LEXICON field with data is found in the figure.

A database designated by the name ParseDb was then created to store the information in the four tables using Micro Soft Access. The structure of the four tables is shown in Figure 5.5 below.

**Figure 5.3:** *Sentence Parser table designs*

| **Table name: Lexicon** | | | |
|---|---|---|---|
| Field Name | Field Type | Field size | Description |
| | | | |

| | | | |
|---|---|---|---|
| Stem | Text | 30 | |
| Category | Text | 5 | |
| Number | Text | 2 | |
| Gender | Text | 2 | |
| Case | Text | 4 | . . . |
| Data source | Number | 3 | |

| **Table name: GrammarRules** | | |
|---|---|---|
| Field Name | Field Type | Field size |
| LHS | Text | 5 |
| RHS | Text | 20 |

| **Table name: MappedRules** | | |
|---|---|---|
| Field Name | Field Type | Field Size |
| LHS | Text | 5 |
| RHS | Text | 20 |
| Index | Number | 5 |
| ParentInd | Text | 5 |

| Table name: Affix List | | |
|---|---|---|
| Field Name | Field Type | Field Size |
| AffixName | Text | 20 |
| Its_association | Text | 4 |
| Its_Position | Number | 3 |

## 5.7 Parsing Algorithms

This section presents the chart and other algorithms used to develop Oromo sentence Parser. The following section discusses the algorithm used in this study. A detailed discussion on how these algorithms work is found in Allen (1995). A module is also discussed for the part that actually performs the Parsing.

### 5.7.1 The Sentence Extraction Algorithm

The sentence extraction algorithm is initiated when the button labeled Open File to parse from file is pressed. In this algorithm, the input string is initialized. There is also a sentence counter which holds the position of the sentence for parsing. After the input string is ready, the word extraction is called to split the complete word into sentence constituent parts. This function removes all the spaces and linefeeds. The word extraction algorithm splits the words based on the space between each word. After the word is extracted from the input sentence, the chart parsing algorithm is invoked which starts parsing the words in the sentence into its phrasal and lexical categories. But before it starts constructing the tree for the parsed sentence, the parsing algorithm calls the morphological analayzer which deals

with the agreement of words in the sentence, detaches the morpheme from each word to get the stem. Finally the tree construction is called and the tree is constructed together with the grid table which displays the information required after parsing. The block diagram below shows the parsing of the algorithm designed. For detailed description of each box and the process involved see the Appendices (B-F).

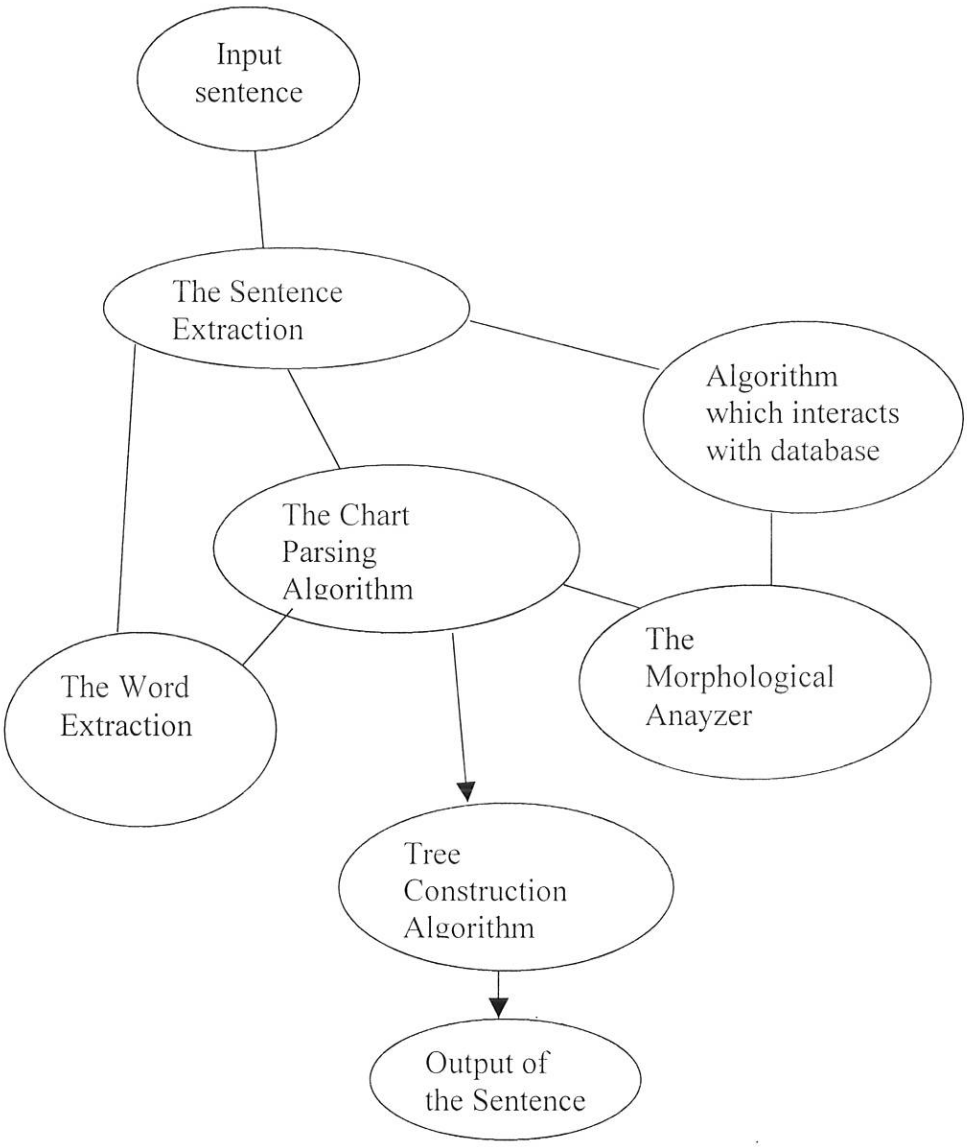



**Figure 5.4 The Parsing System**

## 5.8 The Interface Designed

A sample of the interface designed for the sentence Parser application program is found in the appendix by the name Interface. When the Oromo sentence Parser application program run, the Oromo sentence Parser window will be displayed. This is in fact 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 three buttons at the bottom, just below the menu bar. Below is a brief description of each of these three buttons

The Parse From File button

Clicking this button displays the Parse From File dialog box. This dialog box allows a user to Parse a text written and saved in a file.

1. The Drive, Director and File boxes in the dialog box allow the user to specify the path and file name of the file to be Parsed.
2. Pressing the Parse button Parses each word in the specified file starting from the beginning of file (BOF) to the end of file (EOF) and displays the Parsed text in the output box, labeled Parsed Sentence.
3. Pressing the Save button allows to save the Parsed text in a file by the name and path specified.
4. The progress bar indicate the progress of Parsing while the Parser Parses sentences in the file specified
5. the Next button takes user to the next sentence to be parsed from the file
6. The Back button returns the user to the sentence before the current sentence under the cursor.

The Parse Sentence-by-Sentence button

Clicking this button displays the Parse from File Sentence by Sentence dialog box. This dialog box allows the user to parse sentences in a specified file starting from the BOF to the EOF, but displaying one sentence at a time.

1. The Drive, Directory and File List boxes works as described earlier
2. Clicking the Start button, activates the continue button and display the first sentence extracted from the file in the Unparsed Sentence Box (USB).
3. Pressing Parse, Parses the sentence in the USB and display the Parsed sentence in the Parsed Sentence box (PSB) and also writes the Parsed sentence by opening a file
4. Pressing continue, clears the content of the USB and TSB and then displays the next sentence extracted from the file.
5. Clicking the Parse button again does what is described in step 3 above but this time appending the Parsed sentence in the file opened in step 3.
6. Step 3 and 4 will be repeated until all sentences in the file are exhausted.
7. Pressing the Save button allows the content of the file opened in step 3 to be saved by the name and the path specified.
8. The Back button returns to the main screen from anywhere.
9. The Progress Bar functions as described earlier

The Parse Typed Sentence Button

Clicking this button displays the Parse Typed Sentence dialog box, which allows typing a sentence directly into the input box and parsing it. The buttons in this dialog box works as follows.

1. The input Box labeled write Sentence to Parse is where the user types the sentence (or word) to be Parsed.
2. The output Box labeled Parsed Sentence is where the Parsed Sentence (or word) is
3. Pressing the Parse button Parses the sentence (or word) in the input box and displays the Parsed sentence (or word) in the output box Displayed both in the tree form and grid form to display all the information of available words in the sentence.
4. The Refresh button, clears both the input and output boxes.
5. The Back button and the progress bar function as explained before.

The Exit button

Clicking this button closes and exists the Sentence Parsing application program

## 5.9 The Experiment

As outlined in the evaluation procedure the experiment for this study was carried out in two phases, experiment on the training set and TestSet.

### 5.9.1 Experiment on the Training Set

In this phase, the Parser was first trained on the sample sentences selected from Oromo Grammar book, which is used as TraingSet, and was then run on the same data, i.e. TraingSet. Errors discovered at this stage were more of human than the parser itself (errors made during manual Parsing of the ManuParsedText and in the preparation of GrammarRules and Lexicon), the ManuParsedText, GrammarRules and Lexicon were reviewed making corrections where necessary (for about twenty one times). The Parser was then retrained and the test redone again on the TraingSet. The final results obtained before

and after such corrections were made were then reported under results of the experiment, section 5.9.

### 5.9.2 Experiment on the Test Set

In this phase, the prototype of Oromo language, which was trained on the training set, was run on TestText. The final result achieved on testing the Parser on the unseen part was then reported under results of the experiment.

### 5.9.3 Result of the Experiment

The following section discusses experiment carried on the training and test sets . it also describes the result (before making no correction and after making corrections to the lexicon, grammar and the algorithm) on the training set.

### 5.9.4 Result on the Training Set

The result obtained when the Parser was trained and run on the same data, TraingSet, is shown in table 5.1.

**Table 5.**1 Parsing result before making no error correction

| Data/set | No of sentences | No of erroneously parsed sentences | Accuracy |
|---|---|---|---|
| TrainingSet | 300 | 48 | 84% |

As one would expect the accuracy achieved should be high when a Parser is trained and tested on the same data. But, due to human made errors the accuracy was not as high as it was expected

The final accuracy obtained after human made errors are identified and corrected is shown in Table 5.2 below.

**Table 5.2** *Final Parsing result on the training set after human made errors are corrected.*

| Data/set | No of sentences | No of erroneously parsed sentences | Accuracy |
|---|---|---|---|
| TrainingSet | 300 | 15 | 95% |

As can be seen from table 5.2 the result achieved after correcting human made errors indicates a better, in fact a high, accuracy.

## 5.9.5 Result on the Test Set

The test on the unseen part of the training corpus provides the result shown in table 5.3.

**Table 5.3**: *Parsing result on the test data, i.e.* ***TestSet***

| Data/ set | No of sentences | No of erroneously parsed sentences | Accuracy |
|---|---|---|---|
| **TestSet** | 52 | 6 | 88.5% |

The result obtained when testing the Parser on the test set is approximately 88.5%. This experiment indicates that the accuracy level achieved using the small amount of training set

is rather acceptable. Thus, the Parser developed seemed acceptable with such accuracy assuming the ManuParsedText as the whole of the world knowledge.

## 5.10 Discussion on the Error Analyses

During the experiment the following causes of errors were identified

**1.** Human made errors during the manual Parsing process were identified to be one cause for wrong Parse assignments. Especially the grammarRules of the language was incorrectly inserted in the database, which disturbed the performance of the parser. There were about fifteen errors due to incorrect grammar rules in the database. One of the problems with the grammar rule was its recursive nature. Examples of such recursive rules were VP NP_VP. Later it was identified that such rules made the parser enter an infinite loop. To avoid such problem a MappedRules table was designed to solve the problems.

**2.** The auxiliary verbs attached to their complements were one of the major problems to the parser. This was because as it was indicated in chapter four of this chapter the auxiliary verb is attached to its complement in Oromo. So a morphological Analyzer (which works only for this verb) was designed which detach the auxiliary verb from its complement. After such technique was applied to the parser then its accuracy rate increased significantly. Examples of such cases include dha, ti and all its other forms. Removing this word and repeating the experiment resulted in increased accuracy of the parser before designing the Morphological Analyzer to solve such problems.

3. The other problem encountered while parsing unknown words is that memorization of grammar rules. Thus, while the parser is trying to guess a word not in the database, it uses grammar rules already it has seen. So this created a big problem. To avoid such problems the parser is forced to clear the grammar rules from the mapped rules before it starts the next parsing process. This technique increased the efficiency of the parser in guessing unknown words in an input sentence of Oromo Language. Examples of such rules include the rules of auxiliary verb dha. The parser was unable to parse such type of verbs in a phrase.

4. The other serious problem is the inability of the parser to differentiate between nouns as Adjectives and the adjectives themselves in the same position with the nouns in a sentence. There were about twelve errors due to this problem. It has been tried to identify the shortcoming of this one but finally it was observed that the differentiation of such words may be solved by using a preprocessor or at semantic level in terms of their meaning. Thus this problem remained unsolved through out the experiment. Such examples include the following

Tulluun [mana]N [citaa]N ijaare. "Tullu built a thatched house."

and

Tulluun [farda]N [guraacha]ADJ bite. "Tullu bought a black horse."

5. The other error was during tree construction. While the parser was trying to construct a tree of sentence for the parent node out of child nodes in the process, some were incorrectly constructed. Five errors were due to the failure of the parser to construct correct tree out of its child node for a given input sentence. So a correction was made to the algorithm of tree

construction and the error rate reduced significantly. However, this problem is still unsolved with the auxiliary verb dha. For example, in the following sentence the parser uses the morphological analyzer to detach the auxiliary verb from its complements and correctly identifies sentence constituents. But the problem with the parser is its inability to construct a tree after node identification in a sentence.

Inni [barsiisaa]N [dha]AUX

In this sentence the parser identifies that dha and barsiisaa are AUX and N but unable to attach N to the VP phrase. The alternative solution implemented was to enable the parser detect error and report which node should have been included in the VP phrase construction. This error seems to exist throughout the experiment.

6. Some errors persist throughout the experiments (e.g. dha "is" an auxiliary verb).

### 5.10.1 Dealing with Incorrectly Parsed words in a Phrase

The following were some of the ways used in the experiment to deal with such incorrectly parsed words in a phrase

1. Review of the manually Parsed sentences and make corrections to the errors.
2. Review of the grammar rule used in the database (or knowledge base), which otherwise seriously affect the performance and hence the accuracy of the Parser, and make corrections to the errors and reduce the number of grammar rules. This is because as the number of grammar rules increase the efficiency and accuracy of the parser decreases in terms of time and speed.
3. Avoid or correct words that are assigned wrong Parses and that raise the error rate significantly
4. Avoid spelling the same word differently in the TrainingSet and the table named lexicon

5. Preprocess affixed words in the trainingset to get their root forms from the lexicon if they happen to exist at all.

## 5.11 Summary

In this chapter, detailed discussions were made on the data prepared for the knowledge base, the database designed and the algorithms used. Based on the algorithms, a program was written in Visual Basic and a prototype was developed with an interface easy to be used by any individual.

An experiment was also conducted using the prototype tested on the small sample sentences selected for the experiment. The results achieved and the discussions made out of the experiment are also reported in this chapter.

From what has been discussed in this chapter, the prototype developed has high accuracy, 95 percent for the training set and approximately 88.5 percent for the test set.

The next Chapter closes this thesis work providing conclusions and recommendations.

# CHAPTER SIX

# CONCLUSIONS AND RECOMMENDATIONS

## 6.1 Conclusions

The thesis has tried to describe the development of an Automatic Sentence Parser for Oromo language using the chart algorithm and efforts have been also made to develop a prototype.

The goal of Information Retrieval has been indicated that it is striving to answer users' request correctly in an Automatic Information Retrieval Systems and as a result to increase precision and recall. The contribution of NLP in achieving such goal of Information Retrieval Systems has been clearly pointed out. Furthermore, it has been pointed out how NLP plays a significant role in enhancing computers capability to process NL and communicate something to each other. NLP is also used in the machine translation and other components, which are the core component for the area. To that end, sentence parsing is one components of NLP in contributing to solve the problem of Information Retrieval Systems in answering users' requests. Thus, this paper has tried its best to put in a solution to such problem in Information Retrieval systems.

The main aim of this paper was to develop Oromo Sentence Parser. In this study, important concepts and terms in relation to Parsing have been made clear from the beginning. A detailed review of areas where the outputs of sentence parser are useful was illustrated. Moreover, the different approaches to automated Sentence parser have also been described in some details. Rule-based and stochastic approaches, which are the major approaches to

parsing, were briefly reviewed and the relative advantages of each approach have been included in the discussion. Finally, the Intelligent (hybrid of Rule-based and supervised learning) System approach to parsing was selected in this research to develop Oromo Sentence parser. Reasons for such choice have been discussed. The grammar concepts and the linguistics terms related to this approach were also discussed in some detail.

Literature in the area of Oromo grammatical categories was reviewed and discussed. This is because the knowledge of the grammar of the language is the core component in designing the parser. Moreover, it is pointed out that efforts have been made to reduce the number of grammar rules because of the efficiency problems. These reviews of grammatical categories were used in designing the components of the automatic sentence parser. Almost all lexical categories in a phrase, grammar rules which is the core component of the parser have been exploited as much as possible. In this study, parsing of grammatical categories indicate features like gender, number, tense, case and so on. The parser developed is applicable to sentences other than complex ones which have clauses as phrases in them. In cases where it is not possible to parse words in at phrase level special parsing technique has been introduced.

Some of the major problems that the researcher faced in the process of drawing the lexical category and the steps taken to deal with the problems were also presented.

Steps in preparing the actual grammar rules and Lexicon and the techniques used to implement them in the database were presented in full details. The Lexicon and grammar

rules obtained were presented in two tables, which form the knowledge base of the parser, lexicon and GrammarRules. A database with four tables was then designed to hold the information in Lexicon and GrammarRules.

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 allows 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. Evaluation of the parser performance was made based on the evaluation procedures outlined in the thesis. In the study only one parameter, the percentage of correctly parsed sentences in the sampled text, was used to measure the performance of the parser.

The results achieved based on the small sample were high, 95% on the training set and approximately 88.5% on the test set. Before achieving such accuracy, the experiment was repeatedly done on both the test set and the training set but identifying errors and making corrections. The chart-based algorithm was used in this study with some modification. The supervised learning algorithm suggested by Eric Brill (1993; 1996) was implemented to enable the parser guess unknown words in an input sentences. Most errors identified were due to human made errors rather than the algorithms used. Finally, a discussion on the possible causes of errors was discussed with their solutions before closing up the thesis.

Although the accuracy of the parser developed in this study is somewhat acceptable it may not have an immediate practical application for the parser was not trained on large quantities of data. Moreover, the current system does not include the technique of parsing complex sentences with clauses. Lastly, it is hoped that this thesis has described an approach to study Oromo language, which in future answers the growing desire for parsed texts by researches addressing different issues in Oromo language.

It is also hoped that this first work in parsing Oromo Language will encourage Ethiopian students and researchers to take part in parsing which ultimately lend to a higher level 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 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 Oromo language. Thus, the following could be recommended as possible research areas.

1. Replicate this work using a large data and incorporating all types of sentences with all attributes like case, number, gender, person, tense, definiteness and so on to increase the coverage of current system in parsing various sentence types.
2. Conduct similar researches in other local languages by adopting the procedures followed in this study.

3. Develop parsers for Oromo and other local languages using other approaches (e.g. Stochastic Hidden Markov Model approach) and compare the results obtained using the An Intelligent (hybrid of Rule-based and supervised learning) System approach used in this paper.
4. 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 Oromo sentence parser
5. Conduct research on tagging Oromo texts and identify the problems associated with it, which, I think, will provide exhaustive information as input to Sentence parser.
6. Conduct research on sentence synthesizer (constructor) for Oromo Language.

## Bibliography


Abebe Keno. 2002. Case Systems in Oromo. MA Thesis. Addis Ababa University

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.

Askale Lemma. 1997. *Seerluga Afaan Oromoo*. Unpublished Hand out for Oromo Syntax. AAU

Baye Yimam. 1981. *Oromo Substantives: Some Aspects of Their Morphology and Syntax*. MA Thesis. Addis Ababa University.

Baye Yimam. 1986. *The Phrase Structure of Ethiopian Oromo*. PhD Thesis: London. University of London.

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

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

Brants, T. 1997. Internal and External Tagsets in Part of Speech tagging. University of Dessaarlandes: *computational linguistics*: Saarbrucken, German in proceeding of Euro speech.

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

Brill, Eric (1996) *Unsupervised Learning of Disambiguation Rules for part of Speech Tagging.*

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

(n. d) Corpus Annotation. At:- http://www.comp.lancs.ac.UK/Ucrel/annotation.html/

Doskocs, Tamas E. 1986. Natural Language Processing in Information Retrieval. *Journal of the American Society for Information Science.* Vol. 37(4): 191-196.

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 3/15/2002

*Galmee Jechoota Afaan Oromoo.* 1996. Addis Ababa. Academy of Ethiopian Language Studies

Gragg, G. 1982. *Oromo Dicationary*: East Lansing. Michigan State University.

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.

Hamiid Muudee. *Hamiid Muudee's English-Oromo Dictionary.* Vol.1: Atlanta. Sagalee Oromoo Publishing, Inc.

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

Lehmann, W. P. 1976. *Descriptive Linguistics* (2nd ed): New York. Random House, Inc.

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.

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.

Owens, J. 1985. *A Grammar of Harar Oromo (North Eastern Ethiopia)*: Bumbers. Buske.

Peng, F. (n.d) *The Sparse Data Problem in Statistical language Modeling and Unsupervised segmentation.* At: http://aiz.math.Uwaterloo.ca/~f...proposal3.pdf

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

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.

Sag, I.A. and T. Wasow. 1999. *Syntactic Theory: A Formal Introduction*: Standford. Centre for the Study of Language and Information.

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.

Schildet, H. 1987. *Artificial Intelligence Using C*: Berkeley. McGraw-Hill, inc

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.

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

Tilahun Gamta. 1989. Oromo-English Dictionary: Addis Ababa. University Press.

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.

Volk, M. 1988. *Parsing German With GSPG: The Problem of Separable-Prefix Verbs*. MA Thesis.

Wakshum Mekonnen. 2000. *Development of Stemming Algorithm for Oromo Texts*. MA Thesis

Warner, A.J. 1987. Natural Language Processing. In: Williams, M. E. (Ed). *Annual Review of Information Science and Technology*. 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

# Appendices

## Appendix A: The Sample Text

[Tulluun]NP [[nama]NP [[guraachaa]NP[dha]AUX]VP. [[Inni]N [garuun]N]NP [[[nama]N [magala]N]NP [fakkaata]]VP. [Tulluun]NP [[[mana]N [haadhaatti]N]NP guddatte]VP. [[Seerri] [daima-gudiffaachuun]N]NP [[[oromo]N [keessattii]N]NP [[beekamaa]Ndha]AUX] VP. [[Seerri]N [kun]ADJ]NP [[[seera]N [kabajamaa]ADJ]NPdha]VP.

[Daima-gudiffachuun]NP [[bakka]N [guddaa]ADJ]NP [qaba]V]VP. [[Warri]N [guddifataan]N]NP [[marga]N [fidan]V]VP. [Dubartiin]NP [[daima]N [hinqabne]V]VP. [Dubartiin]NP · [[[ilma]N [quburraa]ADJ]NP [fudhatte]V]VP. [Sinqeen]NP [[barbaachiisaa]NP[-dha]AUX]VP. [Caacuun]NP [[barbaachiisaa]NP[-dha]AUX]VP. [Oromitichi]NP [[daima]NP [guddifatte]V]VP. [[Ilmoon]N [isaa]N]NP [guddade]VP. [[Seerri]N [kun]ADJ]NP [[duri]NP [bahe]V]VP. [[Namonni]N [bayyeen]ADJ]NP [[[seera]N [kana]ADJ]NP [fudhataan]V]VP. [[Ilmi]N [kun]ADJ]NP [[[ilma]N [keenya]ADJ]NP [jedhe]V]VP. [Nitiin]NP [[mucaa]NP [hingodhanne]V]VP. [Gadaan] [[[seera]N [kana]ADJ]NP [hammatte]V]VP. [[Gaddiin]N [okooleen]N]NP· [[kaleessa]ADV [muramte]V]VP. [[[Ibiddi]N [isaan]N]NP [hingubane]V]VP. [Bokkuun]NP [[[seera]N [guddaa]ADJ]NP [qaba]V]VP. [Tulluun]NP [[[seera]N [kana]ADJ]NP [keessatti]PP [guddadde]V]VP.

[Tulluun]NP [[[nama]N [bareedaa]ADJ]NP[-dha]AUX]VP. [[Namummaan]N [issaa]ADJ]NP [[[nama]N [garii]ADJ]NP[-dha]AUX]VP. [Namni]NP [[[isaatti]N [mufaatee]N]NP [hinbeekne]V]VP. [Tulluun]NP [[saree]N [[bayyee]ADV [jalaata]V]VP]VP. [Sareen]NP [[isa]N[-af]P]PP [[hiriyyaa]N[-dha]AUX]VP. [[Hiriyyaan]N [garii]ADJ [[nama]N [hinarganne]NP.

[[Namni]N [baayyeen]ADJ]NP [[Tulluu]N [hinjaalataan]V]VP. [Tulluun]NP [[[amala]N [[dafee]ADV [araa]V]VP]VP. [Tulluun]NP [qoosaa]NP [[baayyee]ADV [jalaata]V]VP]VP. [Tulluun]NP [[nyaata] [[[akka]ADV [malee]ADV]ADVP [nyaade]V]VP]VP. [Inni]NP [[[mana]N [garii]ADJ]NP [hinijaaree]V]VP. [Tulluun]NP [[[mana]N [olaa]N]NP [keessa]PP [jiraata]V]VP]VP. [[Yeroon]N [namaan]N]NP [[loolu]NP [nidalanna]V]VP.

Namni tokko kaleessa dhufe. Namichi sun Tulluu bayyee jaalata. Tulluun ammoo isaa niloole. Tulluun amma baayyee furdaade. Inni garuu kana hinbeeku. Tulluun amma nitii fudhe. Ishiin nama magala fakkatti. Ishiin baayyee nyaatte.

Nitiin isaa barsiistuudha. Manni barumsicha ishii ebise. Ishiin tun naaf kitaaba bite. Kitaabni ishiin fidde baayyee gaariidha. Barsiistonni baayyee ishii jaalatu. Tulluun nitii guraattii hinjaalatu. Kaleessi isaanifi guyyaa gaariidha. Ishiin hoola guraacha guddaa tokko bitatte.

# Appendix B: The Sentence Extration Algorithm

**To extract the first sentence from a file**

1. Initialize a variable, inputstring, to hold the whole content of the file.
2. Initialize a variable, SentenceCounter, to hold location.
3. Assign the whole content of the file to the variable initialized, i.e. inputstring, using the built in string manipulation function.
4. Starting from the first word in the inputstring, extract each word up to the end including the full stop (. Or ?) from the inputString
5. Assign what you extract in step 4 to a variable, inputsentence, which holds only the sentences extracted. (Note that at this stage one sentence is extracted.)
6. Update the SentenceCounter by 1
7. Assign the sentence extracted in step 5 to a text box, unparsed sentence text box, to display the first sentence extracted
8. Read the previous sentence.
9. Repeat step 4 on wards until the end-of-file mark is found.

## Appendix C: The word Extraction Algorithm

When an Open a file to parse from **it** button is pressed, a function that uses the built in string manipulation is used to extract the first word from the input text. Then the following procedures will be executed to split all words of the sentence

1. Initialize a variable to hold the individual word
2. remove all spaces between words in a sentence
3. remove also ENTER
4. if the sentences does not end with a full stop "." Or a question mark "?"
5. display an error message
6. check whether the sentence starts with capital letter
    i. if the sentence starts with non-capital letter
    ii. display an error message
    iii. end if
7. split the sentence into words
8. else call the parsing function

# Appendix D: The Chart Algorithm

When the Parse button is pressed, the Chart-based algorithm is executed in three steps, Arc introduction, arc addition and chart parsing steps. This algorithm is a modified version of Allen's algorithm for chart parsing.

**The Arc introduction Step**

The initialization step will use the following steps for each word identified using the sentence Splitting algorithm discussed above.

1. initialize sentence position and word counter
2. start reading the sentence from the input
3. fetch the grammar rule from the database
4. if the grammar rule in the database is not empty
5. for each sentence, get information about sentence rules from database
6. for each word

    get word category information from database

    if there is no word information in the database

        analyze the new word into its root and morpheme form

        check the database for stem information

else get word category from the database

end if

assign grammar rule of a given word category from the database if it is not empty

collect every constituent of the grammar rule and add them to temporary each

make it active arc (* incomplete arc i.e. that looks for addition of a constituent to be complete its arc

select the completed arc and add them to the main arc

if the main arc is completed, read the next word for its interpretation

end if

5. Repeat step 3 on wards until the end of the sentence.

**The constituent Addition Step**

The Constituent addition step follows the following procedures.

1. Initialize a variable, ActiveArc
2. Starting from the next word, i.e. next to the word identified in the initialization step, to the last word the following steps will be executed iteratively (taken from Allen 1995).
3. To add a constituent C from a position $P_1$ to $P_2$:

    1. Insert C into a chart from position $p_1$ to $P_2$
    2. for any active arc of the form $X \rightarrow x_1, \ldots c \ldots x_n$ from position $P_0$ to $P_1$, then add a new arc $X \rightarrow x_1, \ldots C \ldots x_n$ from position $P_0$ to $P_2$
    3. for any active arc of the form $X \rightarrow x_1, \ldots x_n \ldots c$ from position $P_0$ to $P_1$, then add a new constituent of type X from $P_0$ to $P_2$ to the agenda.
    4. do until there is no input left:

        if the agenda is empty, look up the interpretations for the next word in the input and add them to the agenda

select a constituent from the agenda ( let's call it constituent C from position $P_1$ to $P_2$)

for each rule in the grammar of form $X \rightarrow C\ x_1 \ldots x_n$, add an active arc of form $X \rightarrow C\ x_1 \ldots x_n$ from position $P_1$ to $P_2$.

add C to the chart using the arc extension algorithm above

4. finally clear the grammar rule from the temporary Arc
5. do this iteratively until the end of grammar rule and end of words in the sentence
6. else if parsing has reached the end position then write the end position
7. start selecting the right arcs and add them to the variable called, I array
8. call the tree construction function
9. while constructing tree check for agreement by calling the analyzeagreement function
10. display the output of this process

**The Chart parsing Step**

The chart parsing step follows the following steps

1. get the sentence rule from grammar rule table and word category from Lexicon table
2. Assign the value of rule in the data source to sentence rule
3. if the active arcs are not found
4. exit
5. otherwise, split the grammar rules into its components
6. for each main arcs component and for each active arc assign the main arc component to the left hand side grammar rule
7. check whether the parse is correct or not

8. if right load the active arc introduced and read the word with root form to get word information from the Lexicon table

9. if active arc is found the parse is correct

10. else report, not found message

11. Starting from the next category to the last category it will execute the following steps iteratively

    3.1 Assign the value in the raw of the current category column last index to a variable max1.

    3.2 if the value of max1 is greater than the value of max compared in step 1 of this phase above, then change the value of max with max1 and assign the index of the category to the variable loc.

    3.3 if there is another category start again from step 3 of this phase

    3.4 in another array called C it does the following procedures

        3.4.1 assign the loc. value of lastword +1 found above at the place of last index in the array C

        3.4.2 for each word string from lastword-1 up to the first, assign the corresponding location value from BACKPTR array to a corresponding place in array C

## Appendix E: Morphological analyzer algorithm

The morphological analyzer does the following things to help the parser during parsing.

1. initialize the variable morph
2. get word category from the function called word category
3. read the first word from the sentence
4. if the word is at sentence position 1 then the word is noun, or adjective but if at sentence position 2 then the word is a verb
5. assign the value of full word from the sentence to stem
6. if the string 'DHA' is appended detach it first
7. desuffix function for' DHA'
8. check for number of the word using word information and Affix list table
9. do for other features( gender, number, case, negative marker, and tense) the same iteratively until the end of the sentence
10. finally report the result to the parsing function

# Appendix F: The Tree construction and Extraction algorithm

Finally, the Parse-text function will execute to produce words with their appropriate Parsing as out put

1. split the right hand side rule of the main arc
2. fill the first level children in the tree
3. fill the rest deep components
4. do
    a. collect the deep arcs in the tree
    b. expand the right hand side rule
    c. load the children to the right hand side rule node
    d. repeat until the terminal node is reached

**Tree extraction step**

Fetch the right hand side rule from the database

Split the right hand side rule and assign index to each

Extract tree based on the index value of the rule

Now analyze the agreement case (for number, gender, case, tense and display the output in the tree form and grid form.

## Declaration

This thesis is my original work and has not been submitted as a partial requirement for a degree in any university and that all sources of material used for the thesis have been duly acknowledged.

DIRIBA MEGERSA

JUNE 2002

The thesis has been submitted for examination with our approval as university advisors.

Ato Mesfin Getachew

Ato Million Meshesha

Dr. Haile Eyesus Engdashet