ADDIS ABABA UNIVERSITY
COLLEGE OF NATURAL AND COMPUTATIONAL SCIENCES
SCHOOL OF INFORMATION SCIENCE

AUTOMATIC SYNTACTIC PARSER FOR AFAAN OROMO COMPLEX SENTENCE USING CONTEXT FREE GRAMMAR

By:
GADISA HAYILU DABALO

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

OCTOBER, 2016

ADDIS ABABA UNIVERSITY
COLLEGE OF NATURAL AND COMPUTATIONAL SCIENCES
SCHOOL OF INFORMATION SCIENCE

AUTOMATIC SYNTACTIC PARSER FOR AFAAN OROMO COMPLEX SENTENCE USING CONTEXT FREE GRAMMAR

By:
GADISA HAYILU DABALO

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

ADVISOR: DEREJE TEFERI (PhD)

Name and Signature of the Board of Examiners for Approval:

Name	**Signature**
_____	_____
_____	_____
_____	_____

Declaration:

This thesis is my original work which hasn't been submitted as a partial requirement for a degree in any university.

_____

Gadisa Hayilu Dabalo

October, 2016

The thesis has been submitted for examination with my approval as university advisor.

_____

Dr. Dereje Teferi

October, 2016

Dedicated to:

My father, **Hayilu Dabalo** and my mother, **Damme Worku**; who have brought me to be a man I am today without having academic educations themselves.

ACKNOWLEDGMENTS

God almighty first must be thanked in the name of Jesus Christ for He has been helping me to accomplish my work in all directions as well as my life in peace.

My deepest gratitude is for my family and relatives who are sincerely eager for my success. Even though listing their name does not bear their help, my siblings, i.e., my sister **Birahane** and my brother **Gezahegn Hayilu** have been doing what their portion permitted them right my birth to this day to make me who I am. So I thank them once again and I wish them a long and successful life.

My special thank is also for my advisor: **Dereje Teferi** (PhD) who have been devoting such a long time upon providing his constructive comments and direction that he have been giving me so that the study should be done successfully.

The language experts: **Adugna Barkessa**, one of the academic employees of Addis Ababa University and lecturer at Afaan Oromo department; who really deserves special thanks for his willingness to give me corrections, comments,… at any occasion either in person or over any medium, and he have been providing me many of his publications and materials so that I could get supporting material early and easily. Similarly, **Birhanu Terefe** one of the PhD candidates of this year, has also strongly helped me by providing me required materials. He also gave me directions about what and how to do the work. These experts have shown their special and deep contribution for the success of this work in everything related to the case study that required their interventions.

There are also people who directly and indirectly supported me who deserve my deepest thanks. Among these, my friends like **Tesfaye Tadele**, who was a student of the moment but beside it he has sacrificed his time on generating Afaan Oromo complex sentence. **Yadeta Gonfa**, with I have been spending time in the library as the same time encourage each other on general views that could help us as we were both working on NLP.

I have been having good relationships with these guys right we joined same university and studied bachelor degree with one another on same field.

The staffs of Addis Ababa university school of Information science have supported me on issues concerning them and helpful for this study. So I thank them once again.

Finally, I would like to express my gratitude to those people whose name is not mentioned but their encouragements were great in continuously asking me the progress of my work: when they saw me as well as over cell phone conversations.

TABLE OF CONTENTS

CONTENTS PAGES

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ADJP	Adjective Phrase
BU_STRATEGY	Bottom-up strategy
CC	Conjunctions
CFG	Context-Free Grammar
CYK	Cocke Younger Kasami
D	function assigning probabilities
DC	Dependent Clause
IN	Initials
IPA	International Phonetics Association
JJ	Adjectives
LHS	Left Hand Side
N	Non-terminal symbols
NEG	Negative
NLP	Natural Language Processing
NLTK	Natural Language Tool-Kit
NN	Nouns
NP	Noun Phrase
Num	Numbers
P	Productions or Rules
p	Probability
PCFG	Probabilistic Context-Free Grammar
PoS	Part-of-Speech
PP	Prepositional Phrase
Prep	Prepositions
PRN	Pronouns
RHS	Right Hand Side
S	Start symbol
SCFG	Stochastic Context-Free Grammar
VP	Verb Phrase
VV	Verbs
WFSST	Well Formed Substring Table
XML	Extensible Markup Language

LIST OF SYMBOLS

“ ” ( )	contain strings in Afaan Oromo and its counter English translation respectively.
|	Disjunction form of grammar rules
→	Arrow indicating grammar productions
•	Dot indicating edges in dot rule
Σ	Terminal symbols
ε	Epsilon indicating empty string
α β γ	Greek letters representing arbitrary strings
∈	Element
∪	Union
(Σ ∪N)*	strings formed from finite sequence of terminal symbols
>>>	active python interpreter to write commands
(""" """)	grammar rules of sentence in NLTK are constructed in triple quotes

ABSTRACT

The primary purpose of innovating computer was calculating complex mathematical and arithmetic operations, intuitively: additions, subtractions, divisions and multiplications. Apart from these, the advancement of computer technology is allowing computers to deal with human language which is governed by the discipline natural language processing, even though the media between the two entities varies i.e., computer uses binary digits (0s & 1s) and human uses natural language. When we say processing natural language, the all about of the language couldn't be computerized overnight because language has complex and deep features. Instead, it has knowledge levels: phonology, morphology, syntax, semantics, pragmatics and discourse, so that the processing could be done phase by phase. Once processing human language became promising and possible, it is the role of researchers to process and automate one's language at least through one of the knowledge levels. Although Afaan Oromo is spoken by large number of people, it is yet one of the computationally infant languages because none of its feature is practically functioning for public service except some academicians (students) tried to touch as the result of their academic evaluations. This computational infancy brought unavailability of the basic tools like parser for the research community. Thus, the purpose of this study was to syntactically parse Afaan Oromo complex sentence using context-free grammar, which is used as a component in grammar checking, information extraction, question-answering, semantic analysis and machine translation. Parsing is hierarchically categorized under syntax, and is used to generate valid parse tree for a sentence given grammar and lexical rules. It was investigated using rule based approach and bottom-up parsing strategy. Chart parsing algorithm was selected as a result of its efficiency for managing ambiguities encountered during parsing. Moreover, NLTK and python programming language were selected due to their selectivity for dealing with linguistic data. 250 Afaan Oromo complex sentences constructed from minimum of three and maximum of seven word lengths constituting one independent and one dependent clause with either closed or disclosed single subject. Then, seven different experiments were done by defining grammar and lexical rules of each sentence. The parser encountered parsing errors which were corrected manually by modifying grammar rules. Finally, the parser scored a promising average accuracy of **94.31%** which could attract anyone who wants to continue investigating syntactic parser on Afaan Oromo complex sentence.

Keywords: syntax, parsing, syntactic parsing, Natural language processing, Chart parser, context-free grammar parsing, Afaan Oromo, constituency, parsing complex-sentence.

CHAPTER 1

1.1. INTRODUCTION

The prevalence of computer technology brought the possibilities that machine could understand human language so that interaction between the two entities might be effective and possible. The discipline computational linguistic is all about manifesting these possibilities because it encompasses a broad set of techniques for generation, manipulation, analysis and automation of natural or human languages.

Natural language processing (NLP) as a field has knowledge levels (phonetics and phonology, morphology, syntax, semantic, pragmatics, and discourse) which introduce and initiate researchers to conduct scientific studies around these levels. Each knowledge level is broad and common for all languages even though the computational level and resources of the languages varies from one to another. For example the statistical report of population and housing census (2007) of Ethiopia indicates 34.5% ethnic distributions of Ethiopian population were Oromo, although it is computationally infant hence none of its component is publically and scientifically computerized and standardized.

In fact, it is difficult to computerize these NLP hierarchies overnight; instead developing language processing systems and building reusable modules that can be joined together (when required) to construct NLP applications is the immediate solution. Since NLP's main target is to design and implement computer systems that understand natural language; it is possible to push Afaan Oromo one step forward computationally by investigating and developing syntactic parsing. Syntactic parsing is "the task of recognizing sentence structure and assigning a syntactic structure to it" (Jurafsky & Martin, 2007).

In real world terminologies the term 'parsing' is used in different domains (XML, compiler designing, NLP) having different meaning, and application areas. For example parsing XML file is just how to display and read a text apart from its tags or markups; which is most dominantly covered in integrative programing in which two or more programming languages are required to accomplish the required task (Jokipii, 2003). Parsing is also used in compiler programming for reporting errors in source program (Aho, Lam, Sethi & Ullman, 2007).
Parsing is again broad in NLP because it is available at "morphology, syntax, semantic, and pragmatic levels; in the form of a string, a tree, or a network" (Loftsson, 2007).

Bender (2009) described parsing as rules for part of speech assignments for words, rules for combining words into phrases(syntactic parsing), rules for calculating semantic representations from syntactic structures(semantic parsing), or rules for deriving inflected word forms from stems(morphological parsing).
So the existence and applicability of the term 'parsing' in different unit or features of NLP is identified by the type of data provided as an input and expected output.
Among different domain areas and views of 'parsing' described above, 'Parsing in NLP' was the initial selection in which other (in terms of hierarchy) types of parsing were in turn encompassed under it.

In order to limit its scope, syntactic type of parsing in NLP was the purpose of this study. Even if it somehow appeared as if it was limited; based on parsing approaches syntactic parsing again categorized into context-free grammar (CFG), and statistical.

Additionally, based on grammar structure, parsing in NLP can be categorized into phrase structure and dependency structure; As per Fan et al. (2013) these are the two major paradigms in syntactic analysis which represent parse tree in different form. The parse tree in constituency parsing represents the syntactic structure of a sentence in a form of tree: the innermost layer of brackets denotes the part-of-speech tags, which are lexical classes of the tokens. On top of the PoS tags are the actual phrase-level constituent labels, which represent syntactic roles and imply the semantic scope of each role via the nested bracketing. Phrase structure encompasses both CFG and statistical.

In any category parsing is dependent on grammar which defines rules of one language. Chomsky (1959) justified hierarchy of grammar into four types: type-0 (Unrestricted grammar), type-1 (Context-sensitive grammar, type-2(Context-free grammar) and type-3(Regular grammar).

Regular grammar or regular expression grammars are used for exploring morphological parsing and partial parsing. While context-free a grammar (CFG) are used in syntactic phrase structure parsing and is used for investigating full parsing, and it is the target of this study.

There are two type of parsing approaches (rule based and statistical): rule based parsing is what is so called parsing using CFG where grammar and lexical rules are defined manually so that parse tree will be generated for sentences fitting the rules defined in the grammar.

CFG rules are constructed using four parameters: N, Σ, P, S; which stands for non-terminal, terminal, production or rule, and start symbol respectively.
Statistical parsing is also called stochastic parsing or PCFG hence it extends CFG by assigning probability to grammar productions. So probabilistic context free grammars are also part of syntactic parsing, and it is an extension of CFG which assigns probability of the grammar production to the right hand side CFG rules.

In 2007 Jurafsky and Martin justified two major strategies of parsing: top-down strategy or goal-directed search, and bottom-up strategy or data-directed search. There are also counter algorithms used for parsing: top-down method called re-cursive descent parsing, and a bottom-up method called shift-reduce parsing, and dynamic programming technique called Chart parsing (Bird, Klein & Loper, 2009).

In top-down parsing, as the name indicates we start from top where the grammar production having start symbol S starts, and it expands all constituents that had made an S until the terminal symbol is reached. In contrast, bottom-up parsing strategy starts from input words and finds phrases forming constituents of the terminal until the start symbol is reached. The first two algorithms are simpler but weak in managing parsing ambiguities like infinite loop and generating incomplete parse tree respectively.

However, chart parsing (Jurafsky and Martin, 2007) can easily fix these ambiguities using a table to store sub-trees for each of the various constituents in the input as they are discovered so that the constituents must be computed only once.

1.2. MOTIVATION

Many NLP applications are developed for languages like English made it well-resourced, and simplified human-computer interactions. If computational resource of given language is lesser, no matter how small or large number of speakers there are, it is considered as under-resourced languages. So the disproportion between Afaan Oromo speakers and its computational level inspired researcher to conduct this study.

The other additional motive was: an automatic parser has immediate advantage to be a component for applications like machine translation, semantic analysis, question answering, grammar checking, and information extraction. Natural language has many features, whose automation is mostly the role of researchers specifically the speakers of that language.
These points have interested the researcher to prioritize parser among others. Except Diriba (2002), no one has investigated an automatic parser that could parse Afaan Oromo sentence; which implied that further studies on parsing need to be conducted.

1.3. STATEMENT OF THE PROBLEM

Parsing natural language sentence is a prerequisite for other applications like grammar checking, semantic analysis, machine translation, question answering, and information extraction (Jurafsky & Martin, 2007). For example, in machine translation, the goal is to have the computer translate the given text in one natural language to fluent text in another language without any human in the loop. This is one of the most difficult tasks in NLP. So machine translation approaches use PoS tagging and parsing as preliminary steps.

Also, Jurafsky and Martin (2007), described usefulness of parse trees in grammar checking: in word-processing systems, a sentence which cannot be parsed may have grammatical errors or at least hard to read. Syntactic parsers are also used in lexicography applications for building on-line versions of dictionaries. Currently, Afaan Oromo does not have these applications in which a parser could have been incorporated.

Since parsing is used to describe the process of automatically building syntactic analyses of a sentence in terms of a given grammar and lexicon, the resulting syntactic analyses may be used as input to a process of semantic interpretation, or phonological interpretations, where aspects like prosody, are sensitive to syntactic structure (Pulman, 1991).

Additionally, manually parsing, and engaging experts to parse sentence by hands are tiresome, and consumes long time. Thus, the only solution is to explore an automatic sentence parser, so that any user can parse without expectation and intervention of language' experts. Syntactic parsing in Afaan Oromo is tried by Diriba (2002) on simple sentence but the model developed couldn't manage grammars of complex sentence. Also the algorithm used is WFSST which could not manage immediate dominance unlike chart parsing (Bird et al., 2009).

Therefore, the purpose of this study is to develop an automatic syntactic parser for Afaan Oromo complex sentence.

1.4. RESEARCH QUESTIONS

The following research questions will be answered in the research:

- What are effects of context-free grammar rules for exploring an automatic syntactic parser for Afaan Oromo complex sentence?
- What are the challenges of developing automatic syntactic parser for Afaan Oromo?
- What are the effects of grammar, sentence length and constituencies for a model development?
- What is the performance of the parser based on evaluation parameters like soundness, termination, and completeness?

1.5. OBJECTIVE

1.5.1. GENERAL OBJECTIVE

The general objective of this study is to develop an automatic syntactic parser for Afaan Oromo complex sentence using context-free grammar.

1.5.2. SPECIFIC OBJECTIVES

The following specific objectives are extracted from the general objective:

- To prepare data for the development of an automatic syntactic parser
- To develop a model of an automatic syntactic parser for Afaan Oromo
- To evaluate the accuracy of an automatic syntactic parser
- To conclude the results and forward recommendations, and show future directions that could enhance syntactic parsing for Afaan Oromo.

1.6. SCOPE OF THE STUDY

The scope of the study is to automatically generate parse tree of Afaan Oromo complex sentences that are constructed from three to seven word lengths forming one dependent and one independent clause. The raw sentence needs to be preprocessed, and grammar rules and lexical rules for every sentence must be defined.

1.7. LIMITATIONS OF THE STUDY

1. The experiments were made on small sized sample corpus because unavailability of prerequisite resources like well-organized complex sentences, part-of-speech tagged dataset, and annotated data called corpus. For these reasons complex sentence construction and its preprocessing like part-of-speech tagging and stripping punctuation marks, grammar rule constructions were done manually by the researcher which consumed time.
2. It was possible to explore parser for generic complex sentence; complex sentence in Afaan Oromo has a form of one or more dependent clauses, and one independent clause. So it is again time constraint that made the researcher to limit the formation of complex sentence to one dependent and one independent clause with limited word length. Because constructing complex sentence of more than one dependent clause and its counter grammar rules require much time.
3. It was possible to save time and human resource just by financially investing on corpus development, generating sentences, PoS tagging, data preparation, etc.; but budget was another major constraint that highly limited the work.

1.8. SIGNIFICANCE OF THE STUDY OR EXPECTED BENEFITS

Upon conducting this research on parsing of Afaan Oromo complex sentences, irrespective of its difficulties it can be used as reference to learn more from and help as an input for those who want to conduct researches in the area of NLP applications like machine translation and grammar checking, semantic analysis, question answering, and Information extraction.

Moreover, it is helpful for the speaker of Afaan Oromo to automatically generate parse tree of Afaan Oromo sentence as per its grammar.

It is also helpful to save time and money for experts who consume their time on parsing Afaan Oromo sentences manually.

1.9. ORGANIZATION OF THE PAPER

This thesis is organized in seven chapters. The first chapter covers the introduction parts, problems, objectives of the study; while chapter 2 is literature review and related works that used as a medulla of the study in which sciences, facts, approaches and algorithms of real word syntactic parser, are discussed in detail. Everything of Afaan Oromo required for this study is deeply discussed in chapter 3. The fourth chapter is methodology in which philosophy, tools and techniques of parser were covered. Sample corpus and points about them are discussed in chapter 5, called data preparation. Chapter 6 is experimentation; it is a chapter where facts of syntactic parser were investigated and result is scored. Also challenges and whatever points needing future directions are clearly discussed in chapter 7 named conclusion and recommendations.

CHAPTER 2

REVIEW OF LITERATURES & RELATED WORKS

2.1. REVIEW OF LITERATURES

2.1.1. OVERVIEW

Nowadays prevalence of science and technology allowed hot interactions between computer and human beings, although the media between them differ (computer uses binary digits 0s &1s, person uses natural language). These interactions are managed by the discipline, Natural language processing (NLP) which deals with design and implementation of computer systems that communicate with humans using natural language and whose goal is to build systems that can analyze, understand and generate natural languages(Loftsson, 2007).

Human language has deep features whose processing is managed at different knowledge levels in NLP: phonetics & phonology, morphology, syntax, semantic, pragmatic, and discourse. These aspects of linguistics are listed in their hierarchical order, with phonetics and phonology being the most basic, and pragmatics at the top; at every level ambiguity resolving is common goal (Jurafsky & Martin, 2007)**.**

Chomsky (1959) appreciated this linguistic level as central notion in linguistic theory. Since every level is broad, it is difficult to develop language processing systems overnight but it is possible to build reusable modules in which one could be the input of others that can be joined together to construct larger NLP applications (Loftsson, 2007).

Syntactic parsing is hierarchically categorized under syntax[1] and provides syntactic structure to a sentence given grammar and lexicon and it is "the task of recognizing a sentence and assigning a syntactic structure to it" (Jurafsky & Martin, 2007). Parsing follows either statistical approach or CFG or rule based approach with different dataset and algorithms.

[1] "The word syntax comes from the Greek sýntaxis, meaning 'setting out together or arrangement', and refers to the way words are arranged together"(Jurafsky & Martin, 2007). Syntax is the study of the principles and processes by which sentences are constructed in particular languages (Chomsky, 1959).

2.1.2. LANGUAGE & GRAMMAR

Chomsky (1959) defined language, grammar, and relationship between them: “Language is a collection of sentences of finite length all constructed from a finite alphabet of symbols”, “grammar can be regarded as a device that enumerates the sentences of a language” and “a grammar of L can be regarded as a function whose range is exactly L”.

Grune and Jacobs (2008) described language as first and foremost means of communication, to be used almost unconsciously; grammar is exact, finite-size, complete description of the language.

2.1.3. SYNTACTIC STRUCTURES

Addunyaa (2012) defined grammar as rule which governs language. Grammar is the set of rules that allow us to combine words into larger unit. It is used to study the structure (morpheme, syllable, word, phrase, sentence) of the language.

Aitchison’s argument (as cited in Addunyaa, 2012) knowledge of linguistic scholars about the structure of one language in both written and spoken must be superficial. So to manifest this structure it is mandatory to be part of the community to know culture of the language; because when one child born, the thing that makes it knows the language is the universal grammar that it born with. This grammar has basic principle and parameter to know the language. Principle is common and similar in any language but parameter is based on mother language (Chomsky, 1965).

Grammatical principles underlying languages are innate and fixed grammar; generative grammar (Chomsky, 1957) argues grammar produces an infinite number of utterances, using a limited set of grammatical rules and a finite set of terms.
Therefore, we can generate infinite sentences once grammar rules are obtained.

2.1.4. THE CHOMSKY HIERARCHY OF GRAMMAR

Chomsky hierarchy is a theoretical tool that allows comparing the expressive power or complexity of different languages' formal mechanisms (Jurafsky & Martin, 2007).

The communion of these formal mechanisms is each describes a formal language but the kind of grammars to write with each of this formalism is of different generative power[2]. For example, a context-free grammar can be used to describe formal languages that cannot be described with a finite state automaton (Chomsky, 1959), Table 2.1 shows this hierarchy.

Table 2.1: Chomsky hierarchy of grammars

Type/class	**Grammar**	**Language**	**Rule**	**Automata**
Type-0	Unrestricted	Recursively enumerable	$\alpha \rightarrow \beta$	Turing Machine
Type-1	Context-sensitive	Context-sensitive	$\alpha A \beta \rightarrow \alpha \gamma \beta$	Linear-Bounded
Type-2	Context-free	Context-free	$\alpha \rightarrow \gamma$	Push-down
Type-3	Regular	Regular	$A \rightarrow xB \; or$ $A \rightarrow x$	Regular

Type 0 Unrestricted grammars have no restrictions on the form of their rules, except that the LHS cannot be the empty string ε ($S \rightarrow \varepsilon$), and S does not appear on the RHS of any production. They characterize the recursively enumerable languages whose strings can be listed (enumerated) by a Turing Machine (Jurafsky & Martin, 2007).

Type 1 Context-sensitive grammars have rules that rewrite a non-terminal symbol A in the context αAβ as any non-empty string of symbols. It can be recognized by linear bounded automata (Jiang, 2009).

Type 2 Context-free rules allow any single non-terminal to be rewritten as any string of terminals and non-terminals. A non-terminal may also be rewritten as ε. This grammar can be recognized by non-deterministic pushdown automata (Jiang, 2009).

Type 3 Regular grammars are equivalent to regular expressions. They can either be right-linear or left-linear. A rule in a right-linear grammar has a single non-terminal on the left, and at most one non-terminal on the right-hand side. If there is a non-terminal on the right-hand side, it must be the last symbol in the string (Jurafsky & Martin, 2007).

[2] One grammar is of greater generative power or complexity than another if it can define a language that the other cannot define.

The hierarchy of grammar can be shown according to their complexity levels, i.e., $Type\ 3 \in Type\ 2\ \in\ Type\ 1\ \in\ Type\ 0$. figure 2.1 illustrate these relationships.

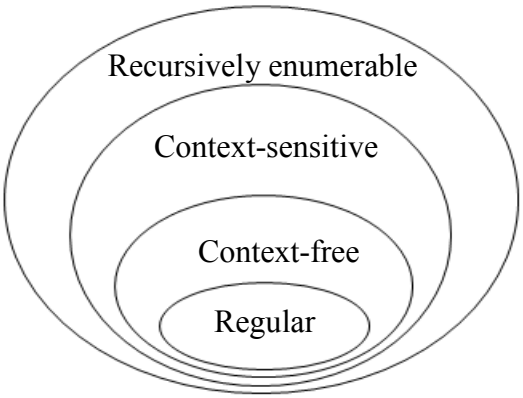

Figure 2.1: Chomsky hierarchy (source: Chomsky presentation, 1959)

2.2.PARSING

The term parsing[3] has different meanings based on the context or domain in which it is used. For example, parsing in XML is just how to display and read a text apart its tags or markups (Jokipii, 2003). Parsing is also used in compiler[4] programming whose role is reporting any errors in the source program (Aho, Lam, Sethi & Ullman, 2007).

Parsing string requires grammar to reconstruct one or more production tree(s) that indicate how the given string can be produced from the given grammar (Grune & Jacobs, 2008). Moreover, parsing is also broadly used in morphological, syntactic, semantic, and pragmatic; in the form of a string, or a tree, or a network (Loftsson, 2007). .

2.3. APPROACHES IN PARSING

Syntactic parsing follows either context-free grammar (i.e., rule-based) approach, or probabilistic context-free grammar (PCFG) approach.

[3] Parsing means taking an input and producing some sort of structure for it.
[4] Compiler is a program that can read a program in one language and translate it into an equivalent program in another language.

2.3.1. CONTEXT-FREE GRAMMARS (CFG) APPROACH

According to Earley(1969) language is a set of strings over a finite set of symbols called terminal symbols represented by lowercase letters, and non-terminals symbols which can be thought as ***syntactic*** classes and represented by capitals. CFG is used as a formal device for specifying which strings are in the grammar set.

According to Jurafsky and Martin (2007) CFG allows modeling of constituency[5] using four parameters called quad-tuples; N, Σ, P, S:

1. N is a set of non-terminal symbols (or 'variables')
2. Σ is a set of terminal symbols (disjoint from N)
3. P is a set of rules or productions, each of the form $A \rightarrow \alpha$, where A is a non-terminal, $A \in N$ and α is a string of symbols from the infinite set of strings, $\alpha \in (\Sigma \text{ UN})^*$.
4. S is a designated start symbol, $S \in N$

Suggesting Earley (1969); Jurafsky and Martin (2007) also divided symbols in a CFG[6] into terminal and non-terminals: The symbols that correspond to actual words in a sentence are called terminal symbols, while the constituted terminal symbols are called non-terminals. Arrow (→) is used to indicate rule or productions among these symbols. The item to right arrow can be one or more terminal or non-terminal, and items to the right the arrow are only one non-terminal symbols.

Lexicons of a sentence are represented with non-terminal associated with each word's lexical category or part-of-speech. Following this sequence of rule parse tree could be generated. This parse tree is represented either by bracketed notation which is more convenient and compact, or diagrammatic representation.

2.3.2. PROBABLICTIC CFG APPROACH

PCFG has similar features with CFG; CFG use context-free rule, probabilistic grammar use PCFG also known as stochastic context-free grammar (SCFG) but each rule or production is associated with a probability (Jurafsky & Martin, 2007).

[5] Constituency is groups of words behave as a single unit or phrase.
[6] The mathematical system for modeling constituent structure in natural languages is CFG.

In 2007 Jurafsky and Martin also justified five parameters (N, Σ, P, S, D) of PCFG. Recall the first four parameters were defined in CFG approach above but P and D, are special here: P is a set of rules or productions of a form A → β [p], where A is a non-terminal, β is a string of symbols from the infinite set of strings $(\Sigma \cup N)^*$, and p is a number between 0 and 1 expressing P(β|A). D is a function for assigning probabilities to each rule in P. Thus, p is the conditional probability of a given expansion β, given the left-hand-side (LHS) non-terminal A, represented as: $P\ (A \rightarrow \beta)\ or\ as\ P(A \rightarrow \beta|A\ or\ as\ P(RHS|LHS)$, The sum of their probabilities must be 1, considering all the possible expansions of a non-terminal which is called consistent[7]:

$$\sum_{\beta} p(\mathrm{A} \rightarrow \beta) = 1$$

2.3.3. ADVANTAGES OF PARSING

Since parsing is used to describe the process of automatically building syntactic analyses of a sentence in terms of a given grammar and lexicon, the resulting syntactic analyses may be used as an input to a process of semantic interpretation, or phonological interpretations, where aspects like prosody, are sensitive to syntactic structure (Pulman, 1991).

Additionally, parse trees are useful (Jurafsky and Martin, 2007) in applications like grammar checking, semantic analysis, machine translation, question answering, and information extraction.

Syntactic parsers are also used in lexicography applications for building on-line versions of dictionaries. Ney; Lari and Young's study (as cited in Jurafsky & Martin, 2007) "the advantages of stochastic parsing algorithms for incorporating into speech recognizers, both for language models and for non-finite-state acoustic and phonotactic modeling".

[7] PCFG is said to be consistent if the sum of the probabilities of all sentences in the language equals 1.

2.4. PARSING STRATEGIES AND ALGORITHMS

Parsing is used as a search to find all trees whose root is the start symbol S, which cover exactly the words in the input. There are two parse strategies: top-down or goal-directed search, and bottom-up or data-directed search (Jurafsky & Martin, 2007). There are analogies between parsing techniques and search algorithms of artificial intelligence: top-down parsing is analogous with depth-first search, bottom-up is analogous with breadth-first search and chart parsing is analogous with best-first search (A*search).

Bird et al. (2009); Jurafsky and Martin (2007) described major parsing algorithms like top-down method called re-cursive descent, bottom-up method called shift-reduce, and dynamic programming algorithms. Thus, algorithms and strategies are linked to one another.

2.4.1. TOP-DOWN STRATEGY & RE-CURSIVE DESCENT ALGORITHM

Top-down parser searches for a parse tree by trying to build from the root node S down to the leaves, then it finds the tops of all trees which can start with S, by looking for all the grammar rules with **S** on LHS (Jurafsky & Martin, 2007). At each layer of the search space it uses the RHS of the rules to provide new sets of expectations for the parser, which are then used to recursively generate the rest of the trees. Trees are grown downward until they reach the part-of-speech categories at the bottom of the tree or until the terminal symbols of sentence are derived. At this point, trees whose leaves fail to match all the words in the input can be rejected. Instantly, the parser back up and tries an alternative called backtracking to maintain failed tree (Jurafsky & Martin, 2007; Bird et al., 2009). The technical demonstration for recursive descent parsing in NLTK is *nltk.app.rdparser().*

2.4.2. BOTTOM-UP STRATEGY & SHIFT-REDUCE ALGORITHM

Bottom-up parsing starts from the input string, and tries to find sequences of words and phrases that correspond to the RHS of grammar production.

The parser then replaces these with the LHS of the production, until the whole sentence is reduced to an **S** (Bird et al., 2009)**.** The parse is successful if the parser succeeds in building a tree rooted in the start symbol S that covers all of the input (Jurafsky & Martin, 2007). 'Shift-reduce' parser is the particular algorithm that follows the notions of bottom-up strategy and technically demonstration as: *nltk.app.srparser()* in NLTK (Bird et al., 2009).

2.5. PROS & CONS OF RECURSIVE-DECENT AND SHIFT-REDUCE PARSING ALGORITHMS

Both algorithms are simple compared to dynamic programming parsing algorithms (Bird et al., 2009). Recursive-decent goes into an infinite loop when it encounters a left-recursive production of the form A -> A, which will assign an infinite set of parse trees to any structure containing an A (Pulman, 1991).

Jurafsky and Martin(2007) described quality of recursive-decent algorithm as it doesn't waste time with trees that do not lead to an S, but it does spend effort on S trees that are not consistent with the input as a result of they can generate trees before ever examining the input. While backtracking, recursive-decent algorithm has a chance of discards trees already correctly constructed in previous grammar traces.

Shift-reduce algorithm can generate trees that have no hope of leading to an S (Pulman, 1991; Jurafsky & Martin, 2007) but it never suggests trees that are not at least locally grounded in the actual input. Since both algorithms were not strong enough for parsing, dynamic programming provides framework for solving these problems once again.

2.6. DYNAMIC PROGRAMMING

Dynamic programming approaches uses table to store sub-trees for each constituents in the input. This is to avoid re-computation problem happened as a result of doing same string twice or more, this technique is called dynamic programming (Jurafsky & Martin, 2007). It includes algorithms like Cocke-Younger-Kasami (CYK), Chart, and Earley algorithm.

2.6.1. CHART ALGORITHM

Chart algorithm is preferable for syntactic parsing among other dynamic programming algorithms (Jurafsky & Martin, 2007). It has different elements to operate its task: Schmidt (2012) categorized these components into: Chart, Edges, and Agenda.

Thompson (2015), Pulman (1991), Jurafsky and Martin (2007) and Bird et al.(2009) categorized them into Chart and Agenda where the chart contains edges and vertices. In one or other cases these all components are used in operating chart parser.

- **Chart**

Chart is a form of well-formed substring table (WFSST) that stores partial and complete analyses of substrings, to use and/or re-use later if needed and to avoid duplicate paths through the search space defined by the grammar(Schmidt, 2012; Jurafsky & Martin, 2007). Chart records or memorizes previously expanded edges to obviate repetitions. Storing partial results of parsing is useful for showing the progress or, the point where the parser fails (Schmidt, 2012). The other tasks of chart is same problem is not solved more than once and it disallows backtracking problem which discards correct trees designed so far (Jurafsky & Martin, 2007).

- **Edge**

A chart consists of a set of edges and vertices; vertices represent the positions between words in an input sentence, and edges represent partial or complete analyses (Pulman, 1991; Bird et al., 2009). According to Schmidt (2012) edge is data structure used for storing information about a particular step in parsing process. It inhabits cells of the chart and contains: start and end position in input string, a dotted rule, edge probability (for PCFG).

- **A dotted rule**

Dotted rules record complete and incomplete constituents by adding a dot to the edge's right-hand-side (RHS). Material to the left of the dot specifies what the constituent starts with; and material to the right of the dot specifies what still needs to be found in order to complete the constituent (Bird et al., 2009). Material to the left of arrow is called LHS which are non-terminal symbol, and RHS can be either terminal or non-terminal obtained to the right of arrow, "'dot' sounds everything is complete to here" (Schmidt, 2012).

According to Bird et al.(2009) dotted edge has a form $[A \rightarrow c_1 \ldots c_d \bullet c_{d+1} \ldots c_n, (i, j)]$, which record a constituent of type A with span (i, j) starts with children $c_1 ... c_d$, but still needs children $c_{d+1} \ldots c_n$ to be complete ($c_1 \ldots c_d$ and $c_{d+1} \ldots c_n$ may be empty).

If d=n, then $c_{d+1} \ldots c_n$ is empty and the edge represents a complete constituent and is called a complete edge. Otherwise, the edge represents an incomplete constituent and is called an incomplete edge. If d=0, then $c_1 \ldots c_n$ is empty and the edge is called a self-loop edge.

If a complete edge spans the entire sentence, and has the grammar's start symbol as its LHS, then the edge is called a parse edge; S ->VP•, (0, 5) is a parse edge.

Complete edge and parse edge are different from each other: Parse edge is the complete edge in which no more parsing is required i.e., when agenda is empty. Complete edge is component of parse edge which indicates the completion of grammar rules.

To demonstrate dotted rule with example, let us take Afaan Oromo complex sentence: "**yoo beelofteetta ta'e laaqana nyaadhu**" (if you are hungry have a lunch). In this sentence adjective phrase (ADJP) is one of phrasal constituents constituting lexicons CC and JJ in a grammar ADJP -> CC JJ; from this the following dotted rules can be constructed:

a. ADJP -> •CC JJ, (0,0)
b. ADJP -> CC •JJ,(0,1)
c. ADJP -> CC JJ•,(0,2)

Active and/or incomplete: dot not the last element of RHS. Active edges are incomplete, in above example (a) and (b) are incomplete and active because the grammar requires further processing hence dot is not at right most of lexical constituent.
Inactive and/or complete: dot is last element of RHS; inactive edges are complete, in above example (c) is complete and inactive because no more processing of the given grammar hence the dot is at right most (Schmidt, 2012).
In (a) first 0 indicates constituents begin at ADJP while the second 0 reflects the dot lies at the beginning. In (b) ADJP begins at position 0, that CC has been successfully parsed and JJ is expected next.
In (c), dot is to the right of all its two constituents, represents the successful discovery of a tree corresponding to ADJP that spans the entire input (Jurafsky and Martin, 2007).

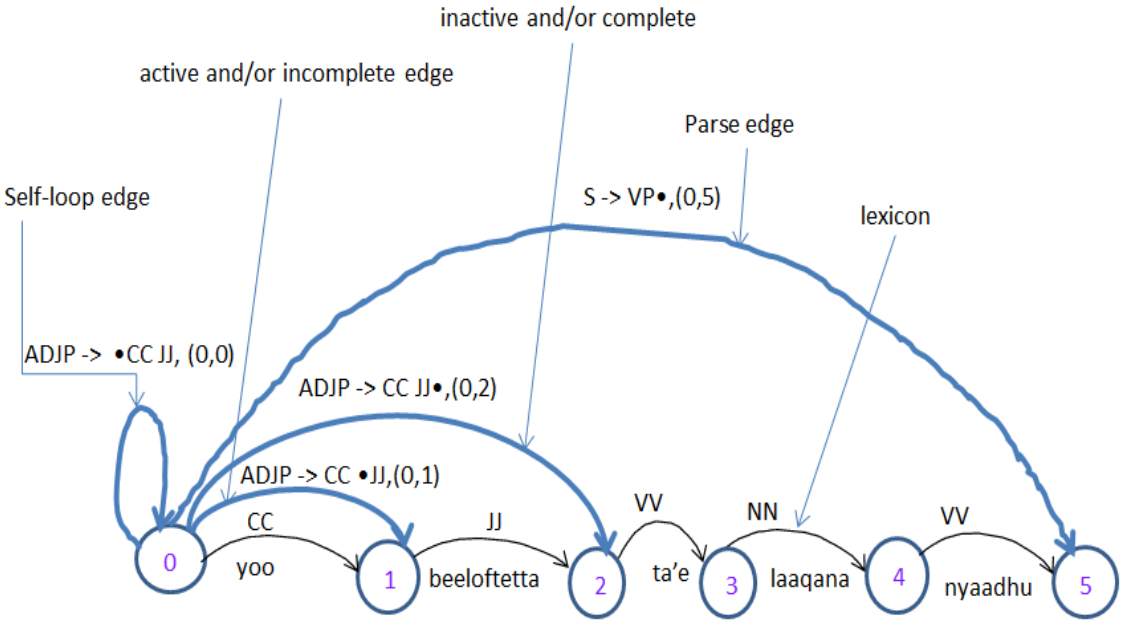

Figure 2.2: Chart terminologies

- **Agenda**

Agenda is list of constituents that need to be processed. Agenda orders the edges added to the chart; to determine what is processed first using either queue, stack, or ordering with respect to probabilities (for statistical parser) (Arnold(n.d)). In stack agenda, every time an edge is added, it is placed on the front of the agenda. While in queue agenda, every time an edge is added, it is placed on the end of the agenda. The choice between these determines the search strategy (Stack organization gives depth first search; queue organization gives breadth first). The addition of an edge to the chart may create new edges, which goes back on the agenda. But edge cannot be added throughout; the parse is over when the agenda is empty.

Chart parser includes different strategies like top-down, bottom-up, left-corner (top-down with bottom-up filtering) chart parser (Bird et al., 2009).The first two are extension of strategies defined so far with idea of parsing algorithms and strategies. Bottom-up chart parser consists of a single loop that removes an edge from the front of an agenda, processes it, and then moves on to the next entry in the agenda. When the agenda is empty, the parser stops and returns the chart (Thompson, 2015). Bottom-up chart parser is selected and described below.

2.7. THE BASIC RULES IN BOTTOM-UP CHART PARSING

To design bottom-up chart parser we should follow three major rules which Bird et al. (2009) discussed below with their diagrammatic illustrations:

i. Bottom-up Initialization Rule

The Bottom-up initialization rule says to add all edges licensed by the input:

$$For\ every\ word\ wi\ add\ the\ edge$$

$$[wi\ \rightarrow\ \bullet\ , (i, i+1)\ ]$$

Figure 5.3(a) illustrates the general notation of this rule:

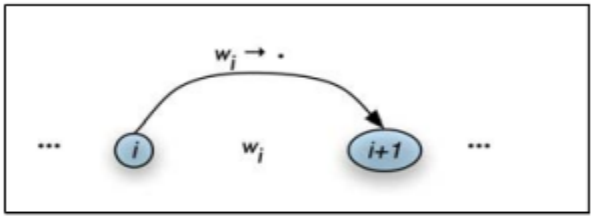

Figure 2.3: Bottom-up Initialization rule

ii. Bottom-up Predict Rule

Suppose the chart contains a complete edge e whose left hand category is A. Then the bottom-up predict rule requires the parser to add a self-loop edge at the left boundary of e for each grammar production whose RHS begins with category A.

$$If\ the\ chart\ contains\ the\ complete\ edge$$

$$[A\ \rightarrow\ \alpha \bullet , (i, j)\ ]$$

$$and\ the\ grammar\ contains\ the\ production$$

$$B \rightarrow A\ \beta$$

$$then\ add\ the\ self-loop\ edge$$

$$[\ B \rightarrow \bullet A\ \beta, (i, i)\ ]$$

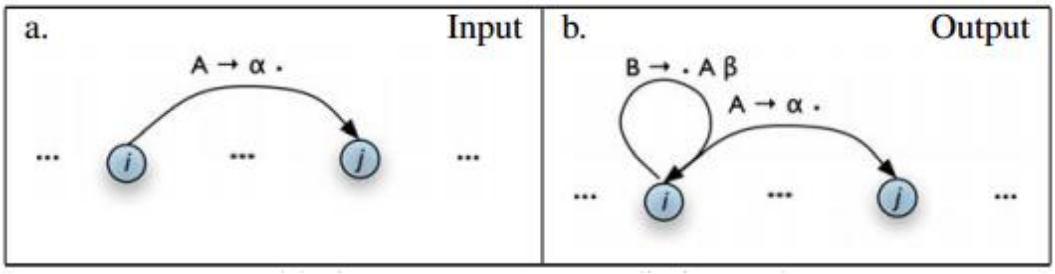

Figure 2.4: Bottom-up Predict Rule

iii. Fundamental Rule

According to (Jurafsky and Martin, 2007; Bird et al., 2009) fundamental rule is useful when the chart contains two contiguous edges where one of the edges provides the constituent that the other one need, as defined below:

If the chart contains the edges

$$[A \rightarrow \alpha \bullet B\ \beta\ , (i,\ j)]$$

$$[B \rightarrow \gamma \bullet , (j,\ k)]$$

Then add the new edge

$$[A \rightarrow \alpha\ B \bullet\ \beta\ , (i,\ k)]$$

It is used to combine an incomplete edge that's expecting a non-terminal B with a following, complete edge whose left hand side is B; diagrammatically shown below:

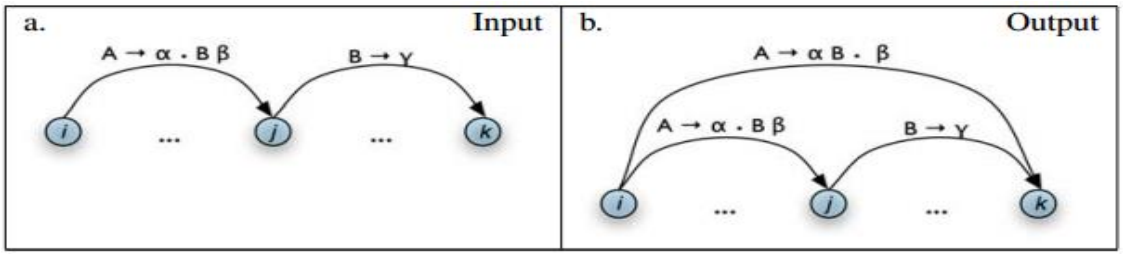

Figure 2.5: Fundamental rule of bottom-up chart parser

Bird et al. (2009) generalizes how bottom-up strategy does work, summarized in figure 2.6:

Create an empty chart spanning the sentence
Apply the Bottom-Up Initialization Rule to each word
Until no more edges are added:
Apply the Bottom-Up Predict Rule everywhere it applies
Apply the Fundamental Rule everywhere it applies
Return all of the parse trees corresponding to the parse edges in the chart

Figure 2.6: Bottom-up Chart parsing

2.8.RELATED WORKS

The following sections describe different works that different researchers tried parsing so far. Among Ethiopian domestic researchers Abiyot (2000) was the first person who developed word parser for Amharic language, which was actually morphological parsing that didn't include the property of parsing at sentence level. He investigated automatic word parser for Amharic words and verbs and their derivations. The purpose of the experiment was to check: the category of word as nominal and verb, tokenization of words to components and inflectional categories of the word based on extracted tokens. He experimented on 200 verbs and 200 nouns. The result showed 86% of the verbs and 84% of the nouns were recognized correctly.

In 2002 Diriba, the first person who tried to develop an automatic sentences parser for Afaan Oromo simple sentence using rule based approaches and chart parsing algorithm. He conducted the experiments on sample narrative text consisting of 352 sentences obtained from "Seerluga Afaan Oromoo". 85 % (300 sentences) of the total sample used for training and the remained 15% were used for testing set, and an accuracy of 88.5% was scored.

Atelach (2002) tried to explore an automatic sentence parsing for Amharic text using PCFG on100 simple declarative sentences composed of four word lengths. 80 sentences were picked randomly for training while the rest 20 sentences were used for testing the model. The sample corpus was automatically tagged using PoS tagger developed so far. The Inside-Outside algorithm was selected to explore experiments, and 85% accuracy was scored.

Daniel (2003) conducted similar experiment on parsing 350 Amharic complex sentences, which were collected from two widely used grammar books of the language entitled with "YaAmarigna Sewasew" and "Amharic for Beginners". Among these sentences, 280 selected as training set and 70 sentences for testing. PCFG parsing was implemented using Inside-Outside algorithm and PCFG bottom-up chart parsing. The final result indicated 81.6% accuracy result.

Mohammed and Omar (2011) developed an Arabic shallow parser system based on rule-based approaches using Support Vector Machine approach. The system was tested manually on 70 Arabic sentences (1776 words) in which each sentence lengthened between 4-50 words. The achievement showed an F-score of 97%.

Loftsson (2007) conducted partial parsing on Icelandic text where, the corpus comprises five categories of texts, i.e. Icelandic fiction, translated fiction biographies and memoirs, non-fiction and books for children and youngsters. No two texts were attributed to the same person and all texts start and finish with a complete sentence consisting of about 590k tokens (3,691 sentences). Each pair consists of a training set, containing about 90% of the tokens from the corpus, and a testset containing about 10% of the tokens. The parser was implemented in Java; the incremental finite-state parser for parsing Icelandic text comprises two modules: the phrase structure module and the syntactic functions module. Evaluation shows that F-measure for phrases and syntactic functions is 96.7% and 84.3%, respectively, when assuming correct PoS tagging. Only three out of eleven phrase types result in less than 95% accuracy, and for the three most common phrase types (NP, PP, and VP) the F-measure was about 97% or higher.

Abeba (2013) investigated a hybrid approach to Amharic base phrase chunking and parsing; to extract different types of Amharic phrases by grouping syntactically correlated words which were found at different level of the parser using Hidden Markov Model and to transform the chunker to parser. Bottom-up approach with transformation algorithm is used to transform the chunker to the parser. Then, 320 sentences were collected from Amharic grammar books and news of Walta Information Center. The training and testing datasets are prepared using the 10 fold cross validation and scored an accuracy of 93.75%.

From these works we can understand that the further work needs to be done on parsing for increasing accuracy and incorporating other sentence types to cover features of the language.

CHAPTER 3

OVER VIEW OF AFAAN OROMO

3.1. OVERVIEW

Afaan Oromo is one of the Ethiopian local languages spoken by largest ethnic group; Oromo people. It is an official language of Oromia (phonetically spelled as Oromiyaa) regional state. Statistical Report of population and Housing Census (2007) of Ethiopia indicated 34.5% ethnic distributions of Ethiopian population were Oromo.

The Latin alphabet called 'Qubee' is used for writing purpose in Afaan Oromo (Hinsene, 2009). Qubee has 33 letters (7 "*Qubee Dachaa*" (compound symbols) and 26 basic letters). Although, the basic letters seem like English, they are represented with distinct sounds given with IPA (International Phonetics Association) symbols, and capital and small counterparts, as shown in table 3.1.

Except the so called compound symbols, letters in English and *Qubee* in Afaan Oromo have similar case and category. Therefore, the Qubee categorized into "*dubbachiiftuu*" (Vowels) and "*dubbifamaa*" (Consonants). The compound symbols include: CH, DH, NY, PH, SH, TS, ZY, ch, dh, ny, ph, sh, ts, zy, which are also categorized under consonant.

Table 3.1: IPA of Afaan Oromo Qubee

Qubee		IPA	Qubee		IPA	Qubee		IPA
A	a	/a/	L	l	/l/	W	w	/w/
B	b	/b/	M	m	/m/	X	x	/t'/
C	c	/č'/	N	n	/n/	W	w	/w/
D	d	/d/	O	o	/o/	Z	z	/z/
E	e	/e/	P	p	/p/	CH	ch	/č/
F	f	/f/	Q	q	/k'/	DH	dh	/đ/
G	g	/g/	R	r	/r/	NY	ny	/ň/
H	h	/h/	S	s	/s/	PH	ph	/p'/
I	i	/i/	T	t	/t/	SH	sh	/š/
J	j	/ǧ/	U	u	/u/	TS	ts	/s'/
K	k	/k/	V	v	/v/	ZH	zh	/ž/

3.1. HIERARCHY OF AFAAN OROMO WORD

Language is a collection of lower and higher hierarchy in structures. The lower level structures combining with one another to build the top level, shown in figure 3.1.

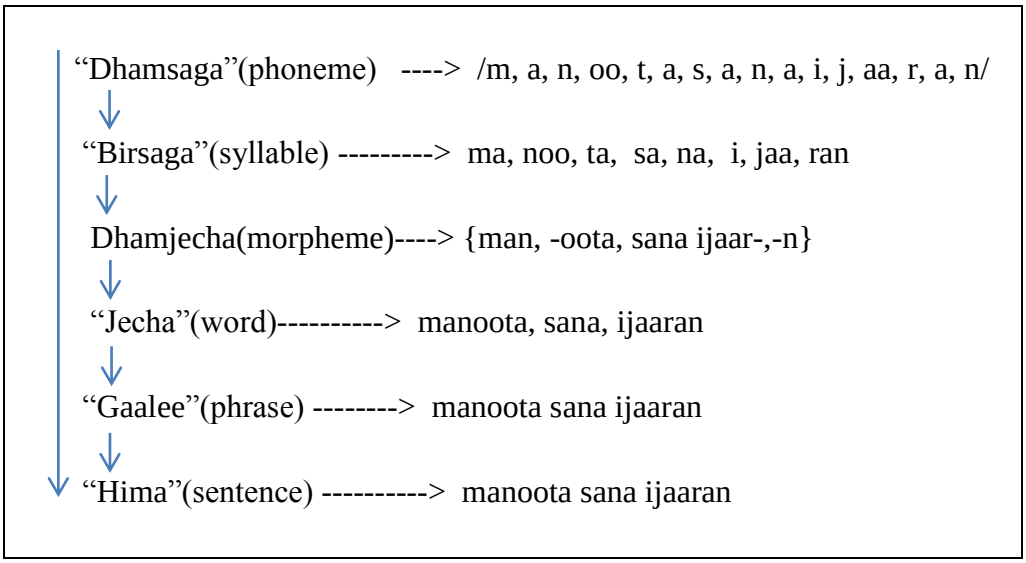

Figure 3.1: Hierarchy in Afaan Oromo word (source: Addunyaa, 2014)

3.2. WORD CLASS IN AFAAN OROMO

Every natural languages, has a standard word order that used to construct grammatical and meaningful sentence. For example, English follows Subject-Verb-Object word order, but Afaan Oromo and Amharic follow Subject-Object-Verb structure.

Greenbaum and Nelson's argument (as cited in Addunyaa, 2013): "one can set as many classes and subclasses as his/her language allows". In traditional word categories, Afaan Oromo words were categorized into eight lexical categories based on their meanings: noun, adjective, verb, adverb, conjunction, prepositions, interjection and pronoun. Moreover, Finch (as cited in Addunyaa, 2013) argued the key to determine word classes of one language is the individual characteristics of that language.

Later, scholars like Mirreessaa (2014) condensed part-of-speech of Afaan Oromo into five: noun, adjective, verb, adverb, and prepositions; which pursued by many domestic researchers like Getachew(2009), Mohammed-Hussen(2010), & Abraham(2013).

However, in 2013 Addunyaa criticized these categories of word class, and suggested the considerations of morphology, syntax and semantics as criteria before determining word classes of Afaan Oromo.

Based on these criteria Afaan Oromo word classes are categorized into four: noun, adjective, verb and preposition. The rest sub-classes are categorized under one of these because they are used to give functions for other words.

For example interjections are formed by exaggerating or by diminishing something but they do not have constant place like other words. Similarly, pronouns are used in place of noun and categorized under noun. Since both conjunctions and prepositions are used for joining one word with other, it is not necessary to assign different word class.

The other major idea that Addunyaa(2013) justified was **adverbs** in Afaan Oromo; they are not as such visible independently but noun and prepositions are cooperatively functioning as adverbs. Due to this reason the form or structures of adverbs are not visible in Afaan Oromo, but they provide functions. This implies adverb cannot form its phrase structure.

Many words that are used as adverb are collected together based on their semantics which brought critics against them. For example in a sentence: "Iddoosaan *mana yaala* deeme" (Idosa went to hospital); the underlined words are noun phrase, the PoS of these words is noun, nevertheless they are used as an adverb, but structurally not adverb. So phrasal categories do not focus on the function or service the words. Also it is possible to say adverb has no independent word class in Afaan Oromo.

3.3. "CIROO" (CLAUSE)

Clauses are a group of words that have subject and verb in their structure/constructions. So in order to say a clause there must be a minimum of one verb. Clauses are categorized into four: dependent, independent, prepositional, and complete clauses (Addunyaa, 2014).

- **Dependent and independent clauses**

Dependent clause is not complete; the existence of conjunction like if, after, until, as soon, etc., made it dependent. Let us see the following examples:

- ✓ "Osoo goota taate"(if you were brave)
- ✓ "Erga waraabessi darbe"(after hyena had passed)

In this example, both sentences cannot deliver complete message; but independent clause completes them so that complete message could be delivered:

- ✓ "Osoo goota taate hin dheessitu"(if you were brave you wouldn't fear)
- ✓ "Erga warabessi darbe sareen dutte"(after hyena had passed the dog barked)

The underlined phrases are independent clauses that completed the incomplete dependent clauses.

- **Prepositional and complete clauses**

These categories are based on their positions in a sentence. Phrases can be positioned either on the boundary of noun phrase, verb phrase or both. The one which obtained at the boundary of noun phrase is called prepositional clause; and the one positioned on the boundary of verb phrase is called complete clause. Being like this, they are used as a constituent to provide information for the phrase, but the complete clause is found on the boundary of verb phrase to make complete the head verb.

Example:

- ✓ "Saamtonni **warri biyya gamsiisan** sun *akka mana hidhaa seenan dhaga'ame*"(it is heard that, those robbers who challenged the country are arrested).

In this example, the bolded words formed prepositional clause. The italicized words are complete clause, and are constituents for the head of the phrase.

3.4. "GAALEE" (PHRASE)

Phrase can be defined as a syntactic combination of one or more than one word(s). It is restricted by the terms of the constituents (elements or members in a phrase) and the lexical categories.

According to traditional grammarians phrase couldn't have subject and verb. Contrasting this idea (Addunyaa, 2014) justified, phrase must have head, which can be subject or verb. And phrase can be constructed from one word (i.e., head word); this criticizes other public sayings as "phrase must be constructed from two and more words".

Therefore, phrase in Afaan Oromo can be built from a group of words that has a head, with different constituencies. For example all the following sentences are phrases.

- ✓ [sangaa]=ox
- ✓ [sangaa diimaa] =red ox
- ✓ [sangaa diimaa guddaa]= red big ox
- ✓ [sangaa diimaa guddaa kalessa argine sana] =red bix ox that we saw yesterday.

All sentences in this example are noun phrase formed from head **ox**, the other words are constituencies.

Note that all concatenated and co-occurred words cannot be a phrase, but it should fulfill three criteria: movement, replacement and attachment (Addunyaa, 2014).

- **Movement:** When words grouped to form a phrase move with one another in a sentence, the place of words can be changed but the meaning of sentence couldn't be changed; this process is called movement criteria.

  Example:
  - ✓ "Bishaan [fayaa namaaatiif] gaaridha" (water is good for health of person).
  - ✓ [Fayyaa namaatiif] bishaan gaaridha" (for health of person water is good).
  - ✓ "Dubartii ogeettin [utubaa manaa]ti" (wise woman is the pillar of house).
  - ✓ "[Utubaan mana] dubartii ogeettidha" (The pillar of house is wise woman).

In this example the phrases "fayyaa namaatif" and "utubaa manaa" are moving as one body showing their phrasal quality. Changing their positions is used to take an attention to the message they are delivering.

- **Replacement:** is to show words moving as one body should be able to be replaced by something like pronoun and affixes.

Example:
- ✓ "[Dargaggeyyiin seexaan guutuu]utubaa hawaasaa tokkoti"(The visionary youths are the pillar of community)
- ✓ "[Isaan] utubaa hawaasa tokkoti"(they are the pillar of a community)

- **Attachment:** no other word entered between the words forming phrase.

Example:
- ✓ "Sangaa diimaa guddaa kaleessa argine sana"(The red big ox that we saw yesterday).

If we insert the word called "*qalle"(slaughter)* between the any two words of the phrase it results ungrammatical phrase except at the end of sentence. This means:
- ✓ "Sangaa *qalle* diimaa guddaa kaleessa argine sana" – we can't say.
- ✓ "Sangaa diimaa guddaa kaleessa argine sana *qalle"*- we can say.

3.4.1. TYPES OF PHRASES IN AFAAN OROMO

Types of phrase in Afaan Oromo are based on types of word classes. It is discussed that noun, adjective, pronoun and verb are word classes which could be head to form phrase for each, as: noun phrase, adjective phrase, prepositional phrase and verb phrase (Addunyaa, 2014).

i. "Gaalee Maqaa"(Noun Phrase)

This formed by having either noun or pronoun as a head. It can be built from head only.
Example: "Mana" (house), "Muka" (tree), "kitaaba"(book) and etc.
These are nouns which can form noun phrase, and they can also form other phrase from their constituency and determiner.
Example: "*Mana* guddaa"(big house), "*Muka* lama"(two trees), "*Sangoota* fooni shan"(five butchery oxen).

ii. "Gaalee Maqibsaa" (Adjective Phrase)

It is a phrase whose head is an adjective, and formed either from an adjective only or by having other constituencies.
For example:

- ✓ "Cimaa"(clever)
- ✓ **"Barumsaan** cimaa"(academically clever), for masculine
- ✓ *"Akka* **oboleessa isaa barumsaan** cima"(like his brother academically clever)
- ✓ *"Baay'ee* **akka obbollessa isaa barumsaan** cima"(very clever academically like his brother).

In these examples a word called "cimaa" (clever) by itself is an adjective phrase because it is a head. It also formed other adjective phrases form its constituency (bolded words) and (italicized). Unlike noun phrase heads in adjective phrase are located to the right side and constituencies are to the left of it.

iii. “Gaalee Durduubee”(Prepositional Phrase)

Prepositional phrase is a phrase whose head is preposition. In Afaan Oromo never phrase is constructed with preposition as head only, but the adposition does.

Example: “Meshaa**dhaan**”(by weapon), “Farda**an**”(on horse), **“**Muka**rra”**(on the tree).

The bolded adpositions in above example are heads to form prepositional phrase. These are dependent prepositions which manage structures that aimed to construct prepositional phrase. Since there are independent prepositions, here they can be used as a head in prepositional phrase.

Example: “**Gara** manaa”(to home), “muka **jala**”(under the tree”), **“Wa’ee** mataa”(about the head). The bolded words are preposition which are used as head. Their position in the structure is both to the right and left side of words in a sentence.

iv. “Gaalee gochimaa”(Verb phrase)

It uses verb as head and like other phrases it can be constructed only from head.

Example: “Deeme”(He went), “Jalqabe”(H/she started”), “dhaabe”(he planted, stopped).

But all head words for verb phrase can’t be independent phrase, so they have the chance of constructing greater structure by suffixing themselves to the root word. For example: {-dha}, {-ti}, {-i}.

This can be independent when it gets noun phrase and adjective phrase as a constituency and they can generate larger structure.

Example:

- ✓ *“Gara manaa* deeme”(he went to home)
- ✓ *“Hojii gaarii* jalqabee”(he started good work)
- ✓ *“Barsiisaa*dha”(he is a teacher)
- ✓ *“Oboleettii* kooti”(my sister)

These verb phrases head verb having right constituencies. So the word preceding head in this example is verb phrase, while the others are noun phrase.

3.5. "HIMA" (SENTENCE)

Sentences are means by which we communicate with others and express our thought and feeling to others. Sentence in Afaan Oromo must fulfill the following criteria:

- a sentence must have at least a subject and verb
- a sentence must be acceptable in the community(speakers of Afaan Oromo) and
- a sentence must be completed/ended with punctuation

3.5.1. TYPES OF SENTENCE

According to the purpose they achieve, sentences are of four types: declarative, interrogative, imperative and exclamatory sentences. And in terms of their structure, they are classified into simple, compound, complex and compound-complex sentences (Addunyaa, 2014; Tullu, 2003). Since syntactic parsing deals with the syntax and structure (constituent parts from which the sentence is formed) of a sentence, the latter category is preferred for this study.

i. Simple sentence

A simple sentence is a sentence with one main clause or it is a sentence which consist only a single verb in its structure no matter how many subjects are there.

Example:

- ✓ "Tolaan mana ijaare"(Tola built a house)
- ✓ "Biliseen, intalli obbo margaa, barsiistuu taate"(Bilise ,the Mr marga's daughter became a teacher).

These sentences have only one verb, but subjects in it can be long or short by adding other indicators next to them.

No matter how the sentence lengthened, if it has a single verb and if it has not dependent clause in its construction, then it is simple sentence. Because clause also should have a verb the total number of verbs in a construction might not be two.

ii. Compound sentence

In compound sentence two or more simple sentences or two or more independent clauses combined together to form a sentence. Each main clause of a compound sentence has its own subject and predicate. These clauses usually combined by: coordinating conjunctions, semi-colon and adding markers like {*-e*} on verb to generate long sound. For example look the following three independent clauses:

- ✓ "Namni gaaridha"(the man is good)
- ✓ "Namni mana ijaare gaaridha"(the man who built house is good)
- ✓ "Namni barumsa barate" (the man learned an education)

These are simple sentences, when we combine them to one another they form the following compound sentence:

- ✓ "Namni mana ijaareefi namni barumsa barate gaariidha"(the man who built house and learned an education is good)

iii. Complex sentence

The thing that makes this sentence complex is an existence of dependent clause. Complex sentence can be formed from one independent clause and one or more than one dependent clause. Let us see the following example:

- ✓ "Yommuu gabaa dhaqxu na waami(call me, when you go to market)
- ✓ "Akka argite hin dubatiin"(do not speak carelessly) .
- ✓ "Yoo soromtes, yoo hiyyoomtes gorsa abbaa kee hin dagatin(don't forget your father's advice when you become rich or poor)

In these examples the first and second sentences are formed from one independent and dependent clause. But the third sentence is formed from two dependent clauses and one independent clause.

iv. Compound complex sentence

This kind of sentence has both characteristics compound and complex sentences. Which means: in one sentence there are simple sentences, or two or more independent clauses, also one or more dependent clause(s).Example:

- ✓ "Yommu dhaqes, yommuu gales, natti goree na gaafatee darbe"(he asked me when he had gone and come).

In this example there are two dependent clauses and three independent clauses. In second sentence there are two dependent and independent clauses (Addunyaa, 2014).

CHAPTER 4

METHODOLOGY

4.1. OVERVIEW

This chapter is about the discussions of research design including philosophy, methods, tools and techniques used for exploring parser for Afaan Oromo complex sentence. Making machine understand natural (human) language is not simple, because natural language has different features and complex property that could be difficult for a machine to read and understand so scientific methodology must be followed.

Research methodology can be viewed as a strategy or plan of action that links methods to outcomes which governs our choice and use of methods (Creswell, 2003). The following points describe the research design required for this study.

4.2. RESEARCH DESIGN

Afaan Oromo have had computational infancy because none of its feature is available for research community except academicians' trials for the purpose of their academic exercises.

This study is focused on automating a feature of Afaan Oromo upon developing syntactic parser for Afaan Oromo complex sentence. Along the increment of computational level of Afaan Oromo, parser is directly used as a component for other NLP applications like grammar checking, semantic analysis, machine translation, question answering, and information extraction. So this study is useful for increasing computational level of Afaan Oromo one step forward, hence other NLP applications incorporate syntactic parser.

This study is conducted based on quantitative experimental research because the results of the experiments were figuratively measured. It is also qualitative research because performance evaluation of parser like soundness, termination and completeness are measured non-numerically.

250 sample of Afaan Oromo complex sentence are randomly constructed by researcher following different research methods described below. Then, the parser constructs parse tree for each sentence which again analyzed by checking soundness, termination and completeness of parse tree. Finally, accuracy is measured based on this evaluation criteria.

4.2.1. METHODS

- **Literature Review**

Syntactic parsing starts from knowing the syntax of the sentence and grammar of the language at all. Since every work of this study started from scratch: sentence construction, choice of complex sentence that fit the objective, constructions of constituencies, grammar rule of sentence and its lexicons, part-of-speech tagging etc.; materials of Afaan Oromo were studied. Similarly, other resources were reviewed to select tools, algorithms and approaches of syntactic parsing. Moreover, literatures and related works to this study were also reviewed.

- **Discussion with Experts**

Upon doing this work linguistic expert had great contribution in correction and approval of every step related to the language: sentence selection, PoS tagging, data preparation, etc. Also for other general things to be known for the study related to Afaan Oromo there were occasions that experts intervened in the work.

4.2.2. APPROACH & STRATEGY

Rule based approach is followed in which grammar rules and lexical rules of each input sentence constructed manually. Bottom-up parsing strategy is preferred for syntactic parsing to construct parse tree according rules defined so far. Bottom-up strategy is technically known as BU_STRATEGY and defined in *parse.chart* (Bird et al., 2009).

4.2.3. ALGORITHM: CHART PARSER

Chart parsing operates on CFG using: chart, edge and agenda; chart is a form of well-formed substring table (WFSST) that stores partial and complete analyses of substrings in one triangular half of a two-dimensional array of (n+1)(n+1)(Schmidt, 2012). Edge stores information about a particular step in the parsing process. Agenda is set of edges waiting to be added to the chart. It is used to prioritize constituents to be processed and organizes the order in which tasks are executed using philosophy of Queue data structure (Bird et al., 2009).

This algorithm cannot be directly used as it is; rather it needs to be modified because Afaan Oromo varies from English language structurally. This modification is done by defining grammar and lexicons of each Afaan Oromo complex sentence in *CFG.fromstring*("""" *"""")* module.

Since the algorithm does not have iterator (even for English) that generate parse tree of every sentence according to grammar constructed so far, we have to add this feature to cover all selected sentences.

4.2.4. SAMPLE CORPUS

Many practical works in NLP uses large bodies of linguistic data, or corpora. "Corpus is a collection of text documents, corpora is the plural form of corpus" (Perkins, 2014). Unlike English, French, Germany, Chinese etc., Afaan Oromo doesn't have annotated corpus called Treebank for parsing so customizing is not possible because there were no predefined corpora and data packages. Due to this reason, researcher constructed 250 Afaan Oromo complex sentences having minimum of three and maximum of seven word lengths each formed from one dependent and independent clause for constructing automatic parser. Because constructing complex sentence of more than one dependent clause and its counter grammar rules require much time.

This corpus is a quantitative primary data because nobody has ever experimented yet this dataset. The dataset is found from books of Afaan Oromo and new data is constructed by guidance of experts.

4.2.5. TOOLS

- **NLTK**

Natural Language Tool-kit (NLTK) is suitable for linguists, engineers, students, educators, researchers, and industry users. It is free, open source, available for all platforms and community-driven project. NLTK has been called "a wonderful tool for teaching, and working in, computational linguistics using Python," and "has an amazing library to play with natural language" (Bird, Klein & Loper, 2009). NLTK is available and compatible for any platform like windows, Mac, UNIX (Bird, 2015).

So this study is investigated using this open-source tool: NLTK 3.0 which has built-in parser defined in *nltk.parse* module that could help us use the available resources in it freely.

- **PyScripter 2.6.0**

Instead of opening different python terminal and interfaces again and again for creating, editing, and running Python programs, installing PyScripter is preferably holds all these features once for all.

- **Python 3.4**

Python is prioritized among other programming languages due to the familiarity of the researcher with it. It is simple and powerful programming language with excellent functionality for processing linguistic data and its installers are available for all platforms (Bird et al., 2009). Thus, NLTK is operated using python 3.4 programming language, on windows 8 platform.

- **Other tools**

Microsoft office 2010 was used for documentation of the paper, and notepad++ 6.9.1 was preferred for editing, preparing and storing corpus in text format, and other ubiquitous tools like diskettes were used for carrying documents and necessary soft-copied materials required for the study.

- **Installations of the tools**

Bird (2015) suggested the following steps to install NLTK on windows.

1. Install Python 3.4 from: http://www.python.org/downloads/
2. Install NLTK from: http://pypi.python.org/pypi/nltk
3. Test installation: Start>Python34, then type "import nltk"
4. Install NLTK Data from: http://nltk.org/nltk_data/

4.1. Run the Python interpreter and type the commands:

```
>>> import nltk
>>> nltk.download()
```

NLTK comes with many corpora, grammars, trained models, book etc., which can be downloaded individually or entirely (by selecting "all" button).

4.2. Test that the data has been installed by running the following command on python interpreter (it assumes you downloaded the Brown Corpus):

```
>>> from nltk.corpus import brown
>>> brown.words()
['The', 'Fulton', 'County', 'Grand', 'Jury', 'said', ...]
```

After installation of nltk is completed we can freely modify and use required algorithms in nltk directory because they are open source. For example chart algorithm for parsing can be obtained in: **C:\Python34\Lib\site-packages\nltk\parse\chart.py.**

4.2.6. DATA ANALYSIS & EXPERIMENTS

Since the approach is rule based, experiments are carried out by constructing grammar and lexicons of sample corpus as in figure 5.1. Then the algorithm (chart parser) constructs parse tree for sentences as per grammar rules constructed so far and finally a model that could parse complex sentence could be developed. Figure 4.1 is flow chart that demonstrates the general operation of bottom-up chart parse.

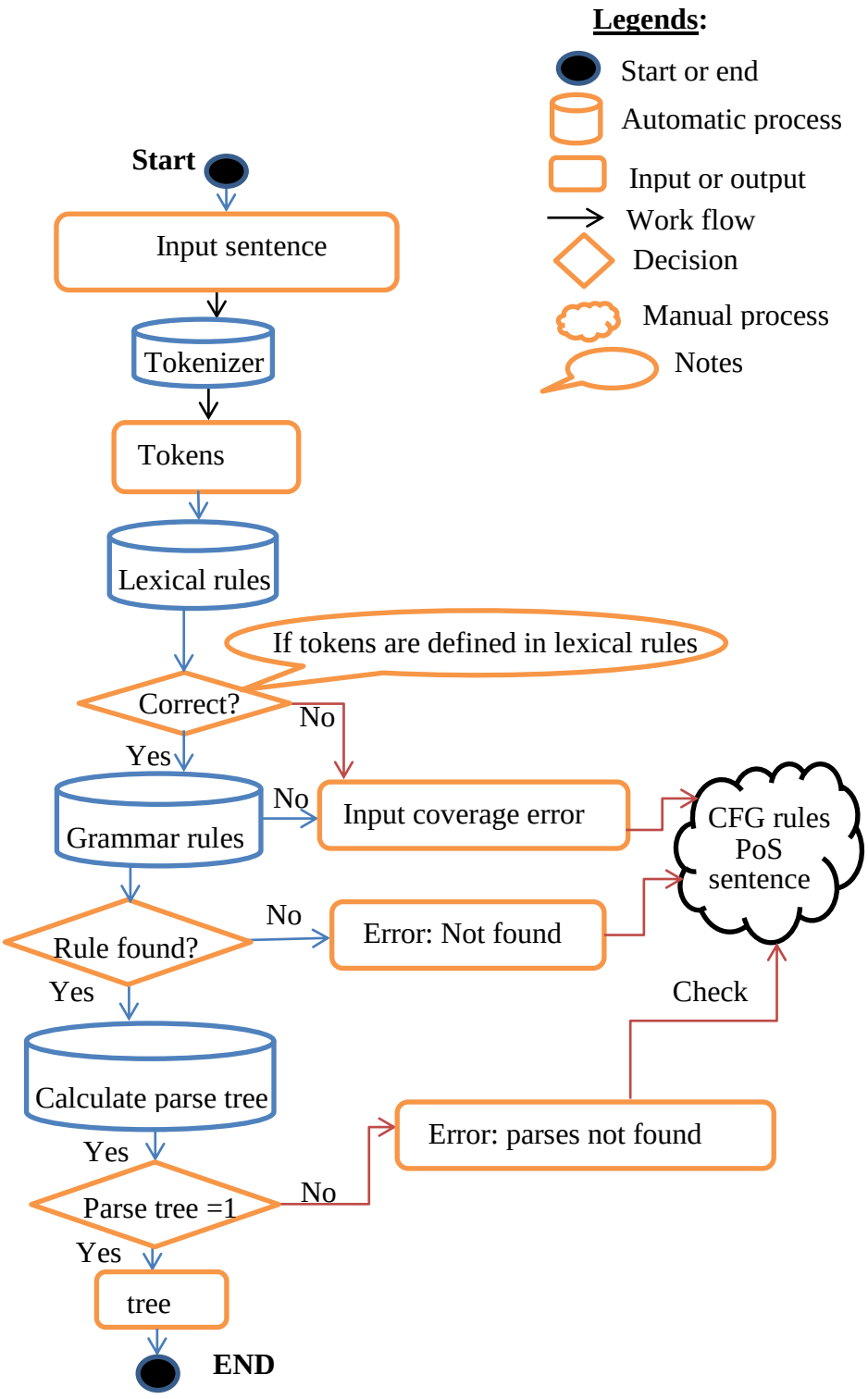

Figure 4.1: Flow chart of bottom-up chart parser (adopted from NLTK 3.0, installed for the study)

From this diagram we can deduce the overall operation of bottom-up chart parser. Grammar and lexical rules of datasets must be constructed for every sentence.

Lexicons of all data should be defined in lexical rules; unless the parser cannot parse it because CFG couldn't cover all input words.
No lexicon is defined as part of both lexical rules and grammar rules for it brings parsing ambiguity called local ambiguity. Since rules were constructed by researcher, none of lexicons had duplicate entry. If such ambiguity happens, chart parser hopefully reports it.

Raw sentence needs to be preprocessed by: stripping punctuation marks, determining PoS of every word of sentence, and listing sentence per new line; till this step everything is manual. Then, the tool tokenizes each sentence and cheeks the match between tokens and lexicons constructed in CFG rules. If it succeeds, the algorithm cheeks the coverage of lexicons by the phrasal constituents then cheeks the grammar rule of each sentence and if it exists it calculates parses. The value of parse tree was set to one so that only one parse tree per sentence is expected; if parses match this length, parse tree will be generated. Generating one parse tree is not enough so its validity is checked again manually using soundness, termination and completeness criteria. If the algorithm encounters error or the NO part of this diagram at any step, the correction is made manually by searching where the problem has happened either checking grammar rules, PoS, sentence form or others.

4.2.7. EVALUATION

Even though prominent authors like Jurafsky and Martin (2007) suggested automatic tools like PARSEVAL and EVALB for measuring accuracy of statistical parser. These tools are neither recommended nor applicable for evaluating accuracy CFG parser because the accuracy result could be meaningless as a result of small quantity of the grammars. Therefore, the accuracy of parser in this particular study is done just by counting, and dividing the correctly parsed sentences to the parsed sentences.

The success and accuracy of parser is evaluated using the following evaluation criteria:
Soundness: A parser is sound if every parse it returns is valid or correct because parser can sometimes generate more than one parse tree which is conceived as an invalid tree.
Termination: a parser terminates if it is guaranteed not to go off into an infinite loop.
Completeness: a parser is complete if for any given grammar and sentence it is sound producing every valid parse, and terminates.

CHAPTER 5

DATA PREPARATION

5.1. OVERVIEW

Once the theoretical perspectives of parser is understood and known; it is a time to think about its practical aspects. But before talking about the experiments it is a must to know how data should be prepared for selected tools, and implemented as per the sciences of the algorithm. Thus, this chapter has unique property of dealing and highlighting pervious chapters; upon recalling chapter 2 syntactic parsing and its relation with syntax of the given sentence and its grammar was understood. Knowing syntax and grammars of case study in turn recall chapter 3; everything about Afaan Oromo (complex sentences, word classes and phrase categories).

Methods and tools for implementation related issues were discussed under methodology chapter. Having the notions of aforementioned chapters, this chapter simplifies its procedure and it starts by identifying minimum things needed to be known.

5.2. THINGS TO BE KNOWN

In order to parse sentence syntactically, knowledge of following points are required:

- **Datasets**

In chapter 3 we have seen the form of complex sentence: one or more than one dependent clause, and only one independent clause. This study focused on the minimum one, i.e., Afaan Oromo complex sentence constructed from one dependent and independent clauses. It is possible to develop parser for any generic sentences, but due time and resource constraints i.e., this work had no prerequisite resources that could have facilitated it.

Thus, the only solution was starting the work from scratch including sentence construction and formation. Additionally, unlike independent clause, the number of dependent clause that Afaan Oromo complex sentence has is not limited so imagining and incorporating them to the study is difficult and takes long time. Due to these reasons researcher constructed 250 complex sentences, as (Chomsky, 1957) argued the possibility of generating infinite set of utterances using a limited set of grammatical rules and infinite terms. So grammars required for the study were reviewed from literatures and books of Afaan Oromo by researcher using guidance and corrections of linguistic experts. Each sentence of this corpus is composed of at least three and at most seven words having either closed or disclosed single subject.

- **Preprocessing data**

The raw Afaan Oromo complex sentence needs to be organized and obtained according to the language's sentence structure. The raw sentence should be preprocessed to fit criteria that a tool requires. Features of NLTK are compatible for Afaan Oromo just as in English so preprocessing raw sentence starts from stripping or removing out punctuation marks, and putting each stripped sentence on a newline, finally save the data in .txt file format.

Note here punctuation is not considered for parsing sentences because they are neither terminal nor non-terminal in CFG rule.

- **Part-of-speech tagging**

In CFG rules the terminal parts are more or less lexical category of a given sentence which implies PoS tag is used as an input for parser even though the structure differs. Part-of-speech tagger represents word with its corresponding PoS like ("word", "PoS") or ("word"/"PoS"), this description is not directly used in CFG parsing but the idea behind it is PoS tagger is useful for parser to define lexicons.

The researcher tried to assign each PoS to every word which proportionally needed constructing grammar rule hence the approach is rule based. But it consumed more time, and assigning the matching tag-set to every word of a sentence made the rule stricter which minimized the accuracy.

Moreover, since the selected corpus is small in number, the researcher assumed to use nine tag-sets, listed in Table 5.1. These tag-sets are used in constructing lexical rules to hold lexicons of each sentence as in figure 5.1.

Table 5.1: Lexical categories

No	Tag-sets	Assigned to
1.	NN	Nouns
2.	PRN	Pronouns
3.	JJ	Adjectives
4.	Num	Numbers
5.	Prep	Prepositions
6.	CC	Conjunctions
7.	VV	Verbs
8.	IN	Initials
9.	NEG	Negative indicator

- **Constituency**

Table 5.2 shows five constituencies selected for this study. The first four are type of phrases in Afaan Oromo, discussed in chapter 3. The nature of constituency is not limited to phrase so as long as lexicons are grammatically united it can form constituency. For example DC is not a phrase, but it is a constituent for a part of sentence that used as dependent clause.

Table 5.2: Constituencies

No.	**Symbol**	**Name**
1	NP	Noun phrases
2	PP	Prepositional phrases
3	ADJP	Adjective phrases
4	VP	Verb phrases
5	DC	Dependent clause

- **Tokenization**

It is not possible to parse raw text directly so each sentence must be broken into its tokens which are used in defining lexical rules.

- **Syntax of grammar (CFG)**

The expected output of syntactic parsing is parse tree of a sentence constructed from grammar rules. This tree can be described with the same analogy of real world tree but inverted structure: the root of parse tree is always the sentence(S symbol), the leaves of the trees are terminal symbols (actual words), and the branches of the tree are phrasal, and lexical constituents (PoS tags). The branches of the tree are actually the nodes in between the root and the leaves representing different grammar rules which are applied to obtain that particular tree.

- **Construction CFG**

In NLTK, CFG is defined in ***nltk.grammar*** or anywhere in the system as long as it is saved with .cfg file extension. It contains four parameters (N, Σ, R, S), and two rules: grammar rules and lexical entries; constructed in figure 5.1.

Grammar rules

S -> NP VP
S -> VP
VP -> DC VP
DC -> NP VV | ADJP VV | CC VV | CC NEG VV | Prep NP VV | NP
DC -> Prep VP PRN | Prep VV | NP Prep VV | CC NEG VV VV | ADJP VV
NP -> CC NN | CC NN PRN | CC PRN NN | CC NN NEG | CC PRN
NP -> Prep NN | CC IN NN | IN NN | IN NN NN | CC Num | NN NN
NP -> CC NN PRN NEG | PRN NEG | PRN | CC PRN NEG
VP -> NP VV | VV
NP -> NN | NN PRN | PRN NN | NN NEG | NN PRN NEG | NN Num
ADJP -> CC JJ | CC NEG JJ | JJ
VP -> ADJP VV | NEG VV | PP VV | NP ADJP NEG VV | PP ADJP VV
VP -> NP PP VV | NP ADJP VV | NP ADJP PRN VV | PP NP VV
PP -> Prep | CC Prep

Sample of lexical rules

PRN -> "ani" | "na" | "nan" | "si" | "inni" | "ati" | "Ati" | "sin" | "hunda"
NEG -> "hin"
Num -> "12ffaa" | "20"
IN -> "Aadde" | "Jeneraal" | "obbo" | "Doctor"
CC -> "yoommuu" | "osoo" | "yoo" | "yenna" | "Erga" | "yeroo" | "eenyulle"
Prep -> "akka" | "ni" | "waan" | "of" | "wajiin" | "giddu" | "haa" | "irra"
JJ -> "beeloftetta" | "aarre" | "baredduu" | "gaarii" | "haaraa" | "gudda"
NN -> "Mosisaan" | "Caalaan" | "Olyaad" | "Gammadaan" | "Soyyamee" | "Tola"
VV -> "dhaqxu" | "waami" | "nyaadhu" | "hojjette" | "geessu" | "Dubbise" | "dubbise"

Figure 5.1: Grammar and lexicons construction of datasets

After defining CFG rules, we have to **unit test** them to find out error happened grammar constructions.

'Unit test' tests algorithms, strategies, and productions selecting for the parser. Chart parser and its strategies were already unit tested based on their theoretical and practical qualities. So bottom-up chart parser passed unit testing and selected to the life time of this study. But technical unit testing was made on context-free grammar: non-terminals and productions.

Non-terminals were: S, NP, VP, PP, DC, ADJP; each of them are defined in grammar rule and none them is used as lexicons of terminal symbols.

Also productions were unit tested: productions' LHS symbol is S as expected and RHSs are NP, VP, PP, DC, ADJP, each of this RHS productions have their own counterpart productions which were correctly tested. Demonstration of unit testing is described in appendix A.

Once the grammar succeeded unit testing we can enter into the detail operations and discussions of figure 5.1.

Grammar rules and lexical rules of every sentence supposed to be parsed must be constructed. Non-terminal associated with each word is its lexical category or part-of-speech as defined under lexical rules. It is case sensitive so it must be similar with tokens of a sentence, unless the grammar coverage problem will be displayed, i.e., the verbs "dubbise" (read) and "Dubbise" are different lexicons represented uniquely as VV -> "dubbise" and VV -> "Dubbise" respectively.

It is not necessary to define grammar rules twice or more for similar phrasal category; for example if grammar has rules: NP -> NN and NP -> CC NN, these could be re-written into disjunction form using '|' symbol just beside previous rule as: NP -> NN | CC NN.

The same is true for lexical constituents: no need of constructing similar lexicons twice and more unless case is different. So if same word of same PoS is used in different sentence just lexicalize them only once.

Similarly, sentences of the same phrasal constituents of different lexicons, the corresponding CFG rule is defined only once using disjunction form. Thus, in aforementioned figure there are two start symbols: **S → NP VP** and **S → VP** that handle start symbol of grammar or root node of parse tree. Each production is equally required as a root for sentences of different structure and constituents. The first production is used for sentences whose subject is disclosed and visible, which leads sentence to have larger noun phrase (NP) branched directly from S. But the latter is used for covering sentences whose subject is hidden that do not generate NP from start symbol S, or it is used when the whole sentence is represented by VP.

For example: “**Ati** yoommuu gabaa dhaqxu na waami” (when **you** go to market call me). This sentence has visible or disclosed subject “**Ati**”(**You**) which caused a sentence to have NP as a result of pronouns can produce NP in CFG rules.

Hiding or closing the subject of this particular sentence is just removing “Ati”, which becomes: “yoommuu gabaa dhaqxu na waami”.

Note: hiding subject in counter English translation of this sentence is not matching with what is supposed in Afaan Oromo version because removing the subject ’you’ would be ungrammatical. Hiding subject is not always grammatical even in Afaan Oromo, but the form of sentence determines it. Both sentences are transferring the same message but they are structurally different resulting different parse tree, shown figure 5.2 and figure 5.3. Therefore, a parser analyzes sentences as per given grammar and produces parse tree.

```
(S (NP (NN Ati))
   (VP (DC (NP (CC yoommuu) (NN gabaa)) (VV dhaqxu))
       (VP (NP (NN na)) (VV waami))))
```

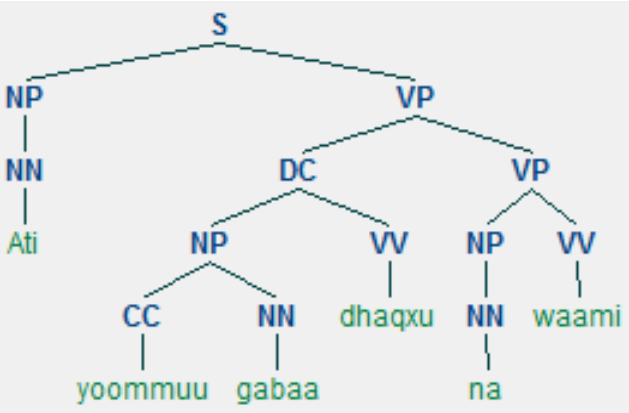

Figure 5.2: Example Parse tree for disclosed subject

(S (VP (DC (NP (CC yoommuu) (NN gabaa)) (VV dhaqxu))
(VP (NP (NN na)) (VV waami))))

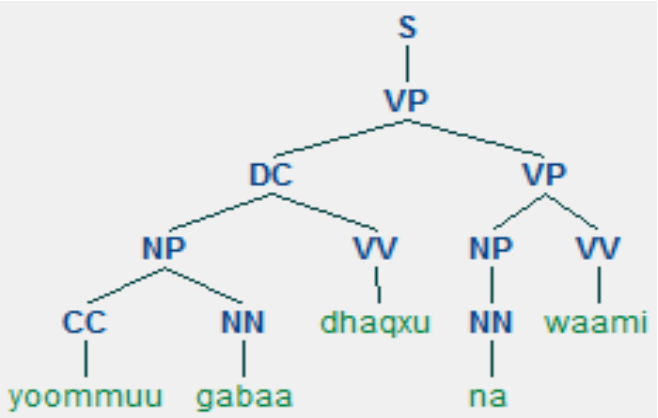

Figure 5.3: Example Parse tree for closed subject

It is not mandatory to display parses in diagrammatic notation so bracket notation is also preferable.

Before generating parse tree, bottom-up chart parser builds initialization, predict and fundamental rules using chart, edges, agenda, and vertices. In order to incorporate these as part of our output, we have to have trace parameter defined in separate method.

Upon assigning trace value to different positive integers, operations of the algorithm could be determined. If trace value is zero it has no tracing output so it only display the bracketed tree, and if the trace value is equal to one, parser displays components of chart parse including rules. But if trace parameter is two, it displays all operations of bottom-up chart parser. These operations of bottom-up chart parsing are described in appendix B on example of Afaan Oromo complex sentence: "**yoo fixxe deemi**" (if you finish go). We start by defining grammar rules of a sentence as usual. Just to specify them for this specific sentence from figure 5.1, the following CFG need to be defined.

S -> VP

VP -> DC VP | VV

DC -> CC VV

CC -> "yoo"

VV -> "fixxe" | "deemi"

CHAPTER 6

ALGORITHM & EXPERIMENTATIONS

6.1. OVERVIEW

This chapter is a stage where theories and assumptions supposed in the study is going to be experimented and the findings are discussed. Rule based approach for syntactic parsing is better to bypass corpus (i.e. Treebank) based stochastic parsing especially for under-resourced languages like Afaan Oromo. Although defining rule for every sentence was troublesome, rule based approach is yet preferable as a result of better accuracy scored upon experimenting on small dataset.

Chart parser algorithm is selected due to its efficiency for managing parsing ambiguities resulted in parsing Afaan Oromo complex sentence hence it holds strings in chart so that they may be seen only once and obviated from duplicate entries. This algorithm builds parse tree by searching and spanning CFG rules using bottom-up parsing strategy (philosophy of queue data structure) then generating this tree to every sentence in the corpus. A correct tree is consistent with the grammar and the leaves of the tree that cover all and only the words in the input. Covering all terminal symbols and building parse tree may not be efficient because the correctness of the tree is determined by number of parses (one parse tree per sentence) and the match between constituents of given sentence and grammar rules.

Moreover, the accuracy of parser is measured using evaluation criteria like soundness, terminations and completeness of parse tree.

6.2. EXPERIMENTATIONS

In this work 250 raw sentence construction, PoS tagging, constructing grammar rules, annotating data, error corrections, etc., were done by researcher based on Afaan Oromo literatures, and corrections and guidance of language experts. The selected dataset was manually parsed on paper; to automatize it we have to construct CFG rules in NLTK.

In order to do so, seven different experiments were conducted by categorizing total data (250 Afaan Oromo complex sentences) into equal ratio, i.e., approximately experiments per 50 sentences.

Following this type of ratio is not mandatory; it is just to define grammar rules of 50 sentences once and trying to generate parse tree and then fix error if any otherwise it is possible even many ratios than this.

In fact, it is possible to construct CFG rules of all sentences but controlling ambiguities is feasible on small sentence. These experimental results are discussed in section 6.3 below.

6.3.FINDINGS

In rule based approach, grammar rules are defined for each sentence, upon doing these any generic sentence beyond complex sentence can be parsed correctly. Because as long as correct grammar and lexical rules of that single sentence are defined, parser easily generates parse tree. But parsing two or more sentences simultaneously have a chance of encountering parse errors, as rules defined for one sentence may **contradicts** rules of other(s). Therefore, contradictions among grammar rules occurred when CFG rule defined for one sentence incorrectly generates parse trees of other sentences. Thus, generating ambiguous parse tree is a serious problem in parsing.

Note: In regular grammars rules are constructed by regular expressions (example PoS tagging and chunking), and it is possible to define rules for unknown words. But in CFGs it is not possible to define rules of unknown words because in rule based parsing we are dealing with rules of a sentence so it is difficult to imagine unknown word to constituent it as a phrase.

Contradiction problem among grammar rules is a natural problem of rule based approach; it most probably happens even though reference dataset is still correctly parsed because reference dataset is manually and uniquely prepared for each sentence without considering rules of other sentences.

The grammar covered lexicons of all input sentences correctly (none of the input sentence encountered input coverage error); hence all sentences were defined with their corresponding lexical rules. As a result of minimum number of sentence, **first experiment** had not had error and it scored better accuracy result as shown below.

$$\text{Accuracy} = \left(\frac{50}{50}\right) * 100\% = \mathbf{100}\%.$$

Second experiment was carried out by increasing data size to 100 i.e., by adding 50 new sentences to the previously parsed sentences but parser encountered **grammar contradiction** problem which caused to generate **incorrect parse trees**.

Justifications: all manually (i.e., on the paper) parsed sentences were 100% accurate, but in second experiment twenty sentences were parsed erroneously generating more than one parse trees for each sentence, i.e., grammar rules of these sentences contradicted one another.

The contradiction problem of this experiment is happened because grammar rules had NP as it could be produced from personal pronouns (first, second, third person) and all nouns. Since each pronouns or nouns could be **head** of NP, when at least two of these pronouns came sequentially in a sentence, it certainly generates at least two parses, contradicting rules of other sentences. Therefore, parser in second experiment scored the following accuracy result:

Accuracy $= \left(\frac{80}{100}\right) * 100\% = \mathbf{80}\%$.

Experiment 3 is conducted by correcting errors of experiment 2:

Parse ambiguity is corrected as follow: accuracy can be increased by minimizing the number of lexical constituents or tag-sets defined in lexical rules, which required modifying grammar rules that all sub-categories of so called personal pronouns and all nouns become condensed and combined into just pronoun and noun respectively.

Therefore, the correction of this error is made by lexicalized pronouns of any kind as pronoun (technically PRN), and all nouns as NN. Then, after error corrections accuracy of parser is improved (as shown below).

$$Accuracy = \left(\frac{100}{100}\right) * 100\% = \mathbf{100}\%$$

Experiment four is carried out by maximizing the size of dataset to 200, and then 18 sentences gave ambiguous parse trees due to grammar contradictions, as shown below:

$$\text{Accuracy result} = \left(\frac{182}{200}\right) * 100\% = \mathbf{91}\%.$$

Continually, fifth **experiment** is carried out to correct errors of experiment 4: correction of grammar rules were done manually by modifying grammar rules of contradicted grammars. Then grammar contradiction problems were corrected and scored the following accuracy.

$$\text{Accuracy} = \left(\frac{200}{200}\right) * 100\% = \mathbf{100}\%.$$

Experiment 6 is conducted on last 50 sentences in similar fashion with previous experiments and the same **problem** encountered by generating structurally ambiguous parse trees for 22 sentences which occurred due to contradictions among grammar rules.

Justifications: Two grammar productions: $S \rightarrow NP\ VP$, and $S \rightarrow VP$ were defined so far to handle start symbol of grammar or root node of parse tree as constructed in chapter 5, but it caused parser to parse NP ambiguously. Defining two start symbols were correct but some grammar contradicted and caused ambiguity.

For example in a sentence***: "osoo namni isa ilaaluu hin nyaatu" (while person is seeing him he don't eat)***, two ambiguous parse trees were generated (figure 6.1 and 6.2). The cause of this parse error is grammar rules (figure 5.1), both start symbol **S** and a constituent 'dependent clause' **DC** have NP as their immediate constituents for different sentences, defined as $NP \rightarrow CC\ NN$ and $NP \rightarrow CC\ NN\ PRN$, respectively. Though NP in each case is same in its function, the first is larger than the latter.

```
(S
  (NP (CC osoo) (NN namni))
  (VP (DC (NP (PRN isa)) (VV ilaaluu)) (VP (NEG hin) (VV nyaatu))))
```

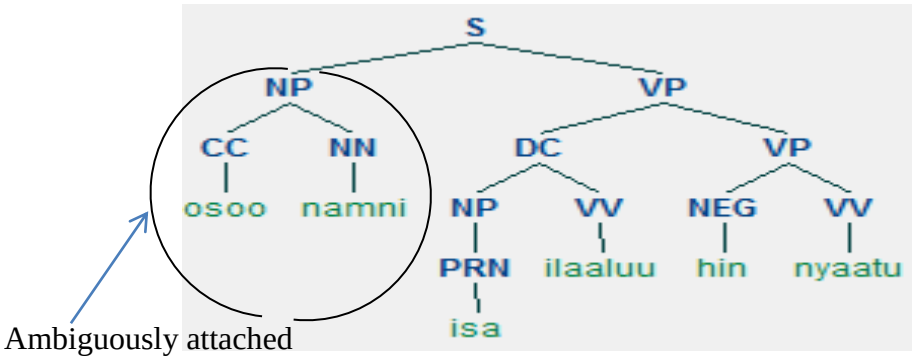

Figure 6.1: Ambiguous parse tree

```
(S
  (VP
    (DC (NP (CC osoo) (NN namni) (PRN isa)) (VV ilaaluu))
    (VP (NEG hin) (VV nyaatu))))
```

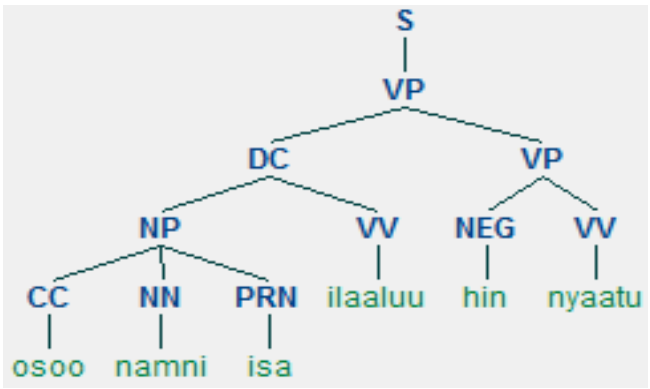

Figure 6.2: Correct parse tree

From these diagrams; the first is ambiguous because the input sentence hasn't had larger phrasal constituent NP because the given sentence is verb phrase but the others (NP, DC, and smaller/inner VP) are its constituents.

The second tree is correct; it was the only parse tree that the parser would have been displayed. But due to the fact that the length of parse was set to one (the parser must output a single parse tree for a sentence as a result of ambiguity management) both trees were rejected.

Therefore, 223 sentences were parsed correctly whose accuracy is calculated below:

$$\textbf{Accuracy} = \left(\frac{223}{250}\right) * 100\% = \mathbf{89.2}\%.$$

Experiment 7: correcting errors of experiment 6, i.e., manually correcting sentence.

The **first** one is modifying the sentence to have similar form because in corpus there are sentences that have two constructions and formation possibilities whose phrasal constituents can either be CC NN or NN CC; both these lexical constituents are noun phrases because their head is noun (NN), such kind of sentence had two parses.

The **second** one is disclosing the subject of sentences: the ambiguous sentence described in experiment 6 has the same meaning with: "namni osoo isa ilaalu hin nyaatu". The difference between the two is only their form (the latter sentence used a common noun "namni" (person) as a subject which made it larger NP and satisfied the rule: S -> NP VP and gave a required parse tree.

So disclosing the subject using third person singular pronouns "Inni" (He) or any proper noun (like "Caalaa") in actual sentence is a solution. These types of subjects were lexicalized as NN in a grammar rule, although they are different in type.

The subject of the original sentence is not visible but it indicates gender singular masculine so it can be described as one of the following two sentences that generated one parse.

- Inni osoo namni isa ilaaluu hin nyaatu
- Caalaan osoo namni isa ilaaluu hin nyaatu

Upon disclosing the subject of sentences of this kind in a corpus, the accuracy of experiment 7 was improved, as calculated below:

$$\text{Accuracy} = \left(\frac{250}{250}\right) * 100\% = \mathbf{100}\%.$$

Total accuracy is the average of all accuracies found in all experiments:

$$\textbf{Average accuracy} = \frac{(100\%+80\%+100\%+91\%+100\%+89.2\%+100\%)}{7} = \underline{\mathbf{94.31\%}}$$

In general, this accuracy of parser succeeded the following evaluation criteria:

Soundness: Even though the parser generated error parse tree, after error corrections the parser returned valid parse tree so none of sentences gave invalid parse tree.

Termination: None of parses went into infinite loop because the maximum parse length was already set to one. Even those unsounded parses had maximum of two parse length which weren't infinite. Moreover, it is the nature of the algorithm (chart parser) that hinders infinite loop.

Completeness: a parser is complete since it was sound and terminates for given grammar, and produced valid parse for a sentence. So the parser was complete except for unsounded parses which latter fixed by manually correcting grammar rules, PoS, and sentence forms. Generally, the samples of final parse tree of a corpus are indicated in appendix C.

CHAPTER 7

CONCLUSION AND RECOMMENDATIONS

7.1. SUMMARY

The purpose of this study was to develop syntactic parser for Afaan Oromo complex sentence by exploring different experiments. Automatic syntactic parsing of natural language sentence have been investigated using ether rule based or statistical approaches. Investigating each approach is more or less dependent on the availability of language's resources. Afaan Oromo hasn't had as such well resources that could have helped for exploring statistical approaches. Consequently, the rule based approach was applied in this study because it is better for dealing with under-resourced languages and small dataset with better accuracy.

Upon investigating context-free grammar parser on complex sentence, computational level of Afaan Oromo hopefully went one step forward. Thus, this model parses Afaan Oromo complex sentence and could in turn be used for incorporating it to other NLP application.
Chart parser was selected in NLTK and implemented using python programming language using bottom-up parsing strategy because it is efficient in managing parsing ambiguities like generating incomplete and infinite erroneous parse trees.

The input sentence for parser was manually preprocessed i.e., striped punctuation marks, putting sentence per new line, and PoS tagging. Every sentence was tokenized so tokens could match lexicons defined in lexical rules. Then, the grammar, and lexical rules for each sentence is defined so that the parser construct parse tree based on set of constituents defined in grammar production for developing model. The CFG rules, algorithm and productions rules were unit tested to find out errors, and each of them succeeded it.

Seven different experiments were made on 250 Afaan Oromo complex sentences by categorizing them into different ratios for defining grammar rules of each sentence. Upon incrementally adding raw data to previously parsed data and then correcting parse errors, the parser scored an average accuracy of **94.31**%, and each parse tree was sound, terminate and complete. This result is promising for exploring further experiments for correcting ambiguities resulted due to rule contradictions so that the accuracy could be increased.
Since length of sentence affects the parse tree, the prototype was developed for Afaan Oromo complex sentence whose minimum word length is three and the maximum is seven.

Also any sentence whose grammar rule is not covering whole structure could not be parsed although its word length is covered in the domain.

Moreover, the accuracy of rule based approach of bottom-up chart parser of this work would welcome those who want to pursue it for further advancing the work as well as managing parsing ambiguities or problems so that effective full-flagged parser would hopefully be developed even for generic Afaan Oromo complex sentence.

7.2.CONCLUSION

- Using CFG to construct rules of Afaan Oromo complex sentence constructed grammar and lexical rules for every sentence. NLP application like stemming, speech recognition, machine translation are encountering problem to deal with Afaan Oromo words having glottal sounds called "hudha" (diacritical) words. But these words were easily defined in CFG since it puts lexicons of a sentence in double quotation marks which became grammatical just like other Afaan Oromo words. Contrasting this, lexical rules for compound words had problem occurred due to the blank space or hyphen between compounded words. Thus, CFG rules covered input sentence adequately.

- In addition to prerequisites resource like NP Chunker and PoS tagger, syntactic parser encountered many technical challenges during experimentations; some of them are: contradictions between CFG rules, grammar coverage problem and incorrect length of parse tree. If sentences are not covered by rules of grammar, it is impossible to generate even ambiguous parse tree. Grammar contradictions occurred among many CFG rules which incorrectly generated parse tree of other sentences.

  So the reason of scoring aforementioned accuracy result was this contradiction among grammar rules especially the lexical constituents defined to form phrasal constituents like NP.

  Additionally, the form and structure of the sentence like disclosing and closing of the subject of sentences affect parse tree even though the contextual meaning of sentences are same. Of course, the algorithm did right because parsing is dealing with syntax so whatever the grammatical structure the sentence have it is the role of parser to generate parse tree irrespective of meaning (i.e., semantics discovery) of sentences.

Modifying the grammar rule and minimizing derived lexical categories into the basic category were the methods for resolving the ambiguity.

- The grammar couldn't cover and generate parse tree for any generic Afaan Oromo complex sentence unless the word length is known and its counter lexicon and phrasal constituent is defined. So sentence length had effect the development of parser because if a lexicon of sentence is not constituted in one of phrasal constituents parser could not construct parse tree. CFG imagine neither unknown lexicon nor undefined constituents if they are not constructed properly in the rule.

- The performance of parser was evaluated and succeeded termination criteria because bottom-up chart parser has a natural characteristics of controlling parsing ambiguities like infinitely generating erroneous and ambiguous parse tree. Through experiments grammar rules were modified to cover phrasal and lexical constituents, so that valid parse tree was generated so soundness and completeness of parser were successful.

  The accuracy of parser was evaluated using the length of parse tree: in the algorithm the parse length is set to one as parameter so that chart parser cannot give parse tree more than its length of parses; which was again one of its qualities for controlling local ambiguities occurred as a result of a lexicon could be assigned to two and more word classes. For those sentences that a parser generated more than one parse, and conceived as parsing error, the cause was the contradictions among grammar rules of sentences.

7.3. STRENGTH OF THE WORK

Parsing complex sentence is better for advancing computational level of Afaan Oromo one step forward though it was more complex than parsing simple sentence.

Most common parsing ambiguities were managed certainly: for example corpus was generated and tagged manually which controlled local ambiguities resulted by having two part-of-speeches for single word. Hopefully, the algorithm (chart parser) reports this ambiguity even if PoS tagging was not manual.

The other special advantage of this study was everything about parser was started from scratch which allowed researcher to know about the language in detail besides syntactic parsing, although more time were consumed.

The diacritical words of glottal sounds have been having ambiguity in many NLP applications like stemming but the algorithm easily recognized sentences containing these words hence their lexicons are automatically put in double quotation marks.

7.4. DISCUSSIONS

Generally**,** the accuracy scored as a result of conducting this study is better compared to parsers explored on Afaan Oromo like Diriba (2002). Even though this study is experimented on complex sentence, the accuracy is hopeful. It is even possible to score better than this if dataset was simple sentence because the grammar contradiction level is easier compared to complex sentence.

7.5.CHALLENGES

Rule based approach of investigating parser has had many challenges that the researcher could not be able to handle them easily. Some of them are listed below:

1. Unavailability of resources:

It is clear that in every computation there should be inputs that must be preconditions for a target output of the study.

- The primary challenge was finding the dataset/corpus of previous researchers: previous researchers only bound and shelved their final documents but there were no chances of finding out what kind of dataset they used, what kind of model they developed and there was no chance of getting even their contact addresses. For example in 2002 Diriba developed an automatic parser for Afaan Oromo simple sentences, I tried to continue his work with the same tittle hence it is research but there was no means of contacting him and getting the corpus he used as well as the prototype he developed.

- PoS tagger: parser must incorporate PoS tagger but it was another major challenge that a researcher encountered.

  In Afaan Oromo there is no standard tagger that could have helped researcher for exploring parsing, but Abraham (2013) tried to develop a brill tagger on 1100 sentences.

Among these sentences, I selected 120 sentences which seemed to be complex sentence and I brought them to language experts to validate whether or not the sentences qualify the criteria of complex sentence: but the experts responses were following:

- The selected sentences were not as such complex sentences because some words needed to be removed; also words must be added to sentences to fulfill the minimum criteria of complex sentence.

  In order not to do so the PoS tagging was under investigation on same corpus by other researcher of this year. Therefore, both researcher and experts saved themselves from modifying the corpus so that the dataset could not be disturbed.
- The PoS of many words was not correct, it had clear problem of valid PoS.

- Chunking or parsing by chunks:
  - Prior to full (Context free) parsing, shallow parsing minimizes time and effort. However, there was no noun phrase Chunker developed so far for this specific language which could have simplified the work.

- Morphological tokenizer: grammar rules of parser are case sensitive; same word of different cases requires defining lexical rule for each lexicon. Defining these rules took large space, and consumed time. So root word and its counter inflected forms were defined uniquely which could have managed by morphological tokenizer or analyzer.

These reasons forced researcher to start the work from scratch right from constructing Afaan Oromo complex sentences.

2. Consultations of experts:

Almost all works of parser require the contribution of linguistics. It is common that other NLP related researches require experts may be on data preparation (especially on training the dataset). But parser requires intervention of experts at any of its phases like:

- Constructing complex sentence and checking its validity
- Tagging of selected complex sentences
- Data preparation: constructing grammar rule and lexicons of every sentence.
- Checking validity of parse tree

The challenges here were: the linguistic experts were not permissible to do all these tasks in time because some of them were students and others (business man) were running their own business. Additionally, some experts refused me to do it even with incentives. The only chance was the researcher first tried every task by himself and brought the result to experts for correction and approval. Doing these consumed more time, effort and cost.

The easier solution for this problem was just assigning officially the language experts as **co-advisor** from linguistics as usual course assignment in regular courses.

3. **Lexical rules for compound words:**

This is a technical challenge encountered as a result of formation of compound words from two different words serving as single unit. In their formation sometimes they include hyphen between words; which were easier to define lexical rules for them. But some compound words can be formed by leaving a space between each other; which were not possible to define their lexical rules.

If we think technically, it is possible to define rules by putting underscore between words, but the parse tree looks strange for language experts and ungrammatical in the language.

4. **Determining ad-position as a head word was a challenge:**

In Afaan Oromo prepositional phrases as a phrasal constituent can be produced either from preposition or ad-position as a head. Prepositions cannot always be a head unlike other lexical constituents instead they could serve as constituents for other phrases.

Moreover, when ad-position became a head, extracting it into head and its LHS constituent was the major and unsolved problem in this study, some of the examples are: "konkolaataa**dhaan**"(by bus),"isaa**f**" (for him), "ciminaa**n**" (strongly), "hattattamaa**n**" (quickly), etc. these are prepositional phrases whose head is bolded and underlined ad-positions. Let us consider "konkolaataadhaan" (by bus), the head is ad-position "-dhaan" (by), and LHS is noun "konkolaataa" (car). This phrase concatenated both noun and ad-position so parsing is difficult to detach it from noun for defining its lexical rules and to make it a head. One might assume stripping ad-position as if it was immediate solution; but it could not be a solution because the numbers of propositional phrases formed in this form are unknown. Note here the examples are neither noun nor adverb, although their corresponding English translation seemed so. Instead they are propositional phrases formed from LHS constituents: noun, noun, adjective and noun respectively, and a head, ad-position.

7.6. RECOMMENDATION & FUTURE WORKS

The computational infancy of Afaan Oromo needs the development of parser from the very beginning which in turn requires the computerization of other natural language features as prerequisites. The above challenges are better research directions that need to be explored, and listed below as immediate recommendations and future works:-

1. **Morphological tokenizer for Afaan Oromo:**

Morphological tokenizer is required for investigating parser in Afaan Oromo to remedy repeatedly defining lexicons of same root with different affixes so future researchers are recommended to develop this feature for Afaan Oromo.

2. **Part-of-Speech tagger:**

Part-of-Speech tagger is also precondition for developing parser in Afaan Oromo; in rule based approach it is useful for determining lexicons of given sentence and in statistical approach it is directly incorporated as one of the modules. This work is also recommended and targeted to coming researchers.

3. **Parsing by chunks:**

Noun phrase chunking is shallower, easier and more efficient than CFG parsing (Abney, 1996). So for this short timed study development of chunker is recommended as future work; and again it will be used as a component for investigating full parsing.
Note: effective PoS tagger is also mandatory for developing both full and shallow parser.

4. **Large syntactic CFG parsing:**

Parsing in this study was explored using CFG or rule based approach on limited data size and word length of Afaan Oromo complex sentence which scored promising accuracy. So it is open and recommended for researchers to increase data size on same title.

5. **Treebank development:**

The manual development of annotated corpus (Treebank) minimizes the effort invested on generating rules (rule based approach) for parsing. Therefore, coming researchers are recommended to develop Afaan Oromo Treebank which requires contributions of third party like Oromia culture and tourism or any concerned body, for investing financial requirements.

6. Probabilistic context free grammar parsing using semi-supervised learning:

Rule based approach i.e., CFG parser sometimes ambiguously outputting more than one tree for single sentence; but PCFG can easily manage and outputs the most probable parse tree. But it requires corpus known as Treebank consisting of sentence and its corresponding parse tree for training. Constructing Treebank for under-resourced language consumes time and requires the expertise of linguists. For example "for 4000 sentences in the Penn Chinese Treebank, experts took two years to manually create the corresponding parse trees" (Zhu & Goldberg, 2009). It is easier to imagine the consumption of proportional year for creating Afaan Oromo Treebank even though both languages are morphologically different.

Luckily, Zhu and Goldberg (2009) suggested semi-supervised learning to minimize problems occurred as a result of paucity of labeled data. Although obtaining labeled data may be expensive but unlabeled data is available in large quantity, semi-supervised algorithms learn from both datasets. So this is also a future work for coming researchers.

7. Syntactic parsing of generic Afaan Oromo complex sentence:

This study is conducted on Afaan Oromo complex sentence formed from single dependent and independent clauses. But this sentence can be constructed from more than one dependent clauses. So continuing parsing complex sentence of any form is also recommended as another research direction.

Note: the sequence of these recommendations minimizes the general challenges encountering in parsing. For example if there is no Treebank for the language, probabilistic parsing is really difficult.

8. REFERENCES

Abeba Ibrahim. (2013). A Hybrid Approach to Amharic Base Phrase Chunking and parsing (Unpublished Master's thesis). Addis Ababa University, Addis Ababa, Ethiopia.

Abiyot Bayu. (2000). Design and Development of Word Parser for Amharic Language (Unpublished Master's thesis). Addis Ababa University, Addis Ababa, Ethiopia.

Abney,S.(1996). *Part-of-Speech Tagging and Partial Parsing*. Kluwer academic publishers.

Abraham Gizaw. (2013). Improving Brill's Tagger Lexical and Transformation Rule for Afaan Oromo Language (Unpublished Master's thesis). Addis Ababa University, Addis Ababa, Ethiopia.

Abraham Tesso. (2014). Challenges of Diacritical Marker or Hudhaa Character in Tokenization of Oromo Text (Master's thesis). *Journal of Software*, 9(7),1818-1825.

Addunyaaa Barkeessaa.(2012). *Natoo*: Yaadrimee Caasluga Afaan Oromoo(Concept of Afaan Oromo Grammar); Finfinne, Ormiyaa.

Addunyaaa Barkeessaa. (2013). *Sanyii*: Jechaafi Caasaa Isaa(Word and its structure). Finfinne, Ormiyaa.

Addunyaaa Barkeessaa. (2014). *Semmoo*: Bu'uura Barnoota Afaaniifi Afoola Oromoo. Finfinne, Ormiyaa, Far East Trading *PLC*.

Aho, A.V., Lam, M.S., Sethi, R., & Ullman, J.D. (2007). Compilers: Principles, Techniques, and Tools(2nd ed.). Pearson Education,Inc.

Arnold, D.(n.d). Chart Parsing. *Dept. of Language & Linguistics, University of Essex*. Retrieved 3 March 2016, from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.131.3502&rep=rep1&type=pdf

Atelach Alemu. (2002). Automatic Sentence Parsing for Amharic Text an Experiment Using Probabilistic Context Free Grammars (Unpublished Master's thesis). Addis Ababa University, Addis Ababa, Ethiopia.

Bender,E.M.(2009). Modern Approaches to Parsing: Symbolic and Statistical.

Bird, S. (2015). Installing NLTK: nltk 3.0 documentations on. Retrieved 10 March 2015, from http://www.nltk.org/install.html

Bird,S., Klein,E., & Loper,E.(2009). Natural language processing with python (1st ed.). USA : O'Reilly Media, Inc.

Chomsky, N. (1957). Syntactic Structures. Mouton publishers: The Hague, Paris.

Chomsky, N. (1959). On certain formal properties of grammars. *Information and control*, 2(2), 137-167.

Chomsky, N. (1965). Aspects of the Theory of Syntax (Vol.11). MIT press.

Collobert,R., Weston,J., Bottou,L., Karlen,M., Kavukcuoglu,k., & Kuksa,P.(2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12, 2493-2537.

Creswell, J. W.(2003). Research design: Qualitative, Quantitative, and Mixed Methods Approaches(2nd ed.). Sage Publications, Inc.

Daniel Gochel. (2003). An Integrated Approach to Automatic Complex Sentence Parsing For Amharic Text (Unpublished Master's thesis). Addis Ababa University, Addis Ababa, Ethiopia.

Diriba Megersa.(2002). An Automatic Sentence Parser for Oromo Language Using Supervised Learning Technique Language (Unpublished Master's thesis). Addis Ababa University, Addis Ababa, Ethiopia.

Earley,J.(1969) . An efficient context-free parsing algorithm. *Communications of the ACM*, 13 (2), 95-102.

Fan, J. W., Yang, E. W., Jiang, M., Prasad, R., Loomis, R. M., Zisook, D. S., ... & Huang, Y. (2013). Syntactic parsing of clinical text: guideline and corpus development with handling ill-formed sentences. *Journal of the American Medical Informatics Association*, *20*(6), 1168-1177.

Getachew Mamo. (2009). Automatic Part of Speech Tagging for Afaan Oromo Language (unpublished master's thesis), Addis Ababa University, Ethiopia.

Grune,D., & Jacobs,C.J.H.(2008). *Parsing Techniques: A Practical Guide* (2nd ed.), Springer: Science and Business Media, LLC.

Hinsene Makuria.(2009). *Elellee conversation*: Afaan Oromo writing system. Addis Abba,Ethiopia,Commercial printing.

Jiang, J. (2009). Chomsky hierarchy.

Jokipii, A. (2003). Grammar-based data extraction language (GDEL).

Jurafsky,D., & Martin,J.H.(2006*). Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition*. Prentice-Hall, Inc.

Loftsson, H. (2007). *Tagging and parsing Icelandic text* (Doctoral dissertation, University of Sheffield, UK).

Mirreessaa Amanu. (2014). *Qalbii*: SeerlugaOromoo. Finfinnee, Oromiyaa, ELLENI P.P.PLC.

Mohammed, M. A., & Omar, N. (2011). Rule based shallow parser for Arabic language. *Journal of Computer Science, 7*(10), 1505-1514.

Mohammed-Hussen Abubeker. (2010). Part of speech tagger for Afaan Oromo language using transformational error driven learning (TEL) approach (Unpublished Master's thesis). Addis Ababa University, Addis Ababa, Ethiopia.

Perkins,J.(2014). *Python3 Text Processing with NLTK3 Cookbook (2nd ed.)*. Over 80 practical recipes on natural language processing techniques using Python's NLTK 3.0. Birmingham ,Mumbai; Livery Place: Packt Publishing Ltd.

Pulman,S.G.(1991). Basic Parsing Techniques: an introductory survey. To*: appear in Encyclopedia of Linguistics*: Pergamon Press and Aberdeen University Press.

Schmidt, A.(2012). Basic Parsing Algorithms–Chart Parsing. Seminar:Recent Advances in Parsing Technology.

Summary and Statistical Report of the 2007 Population and Housing Census. Printed by United Nations Population Fund (UNFPA).

Thompson,H.S.(2015). Accelerated Natural Language Processing.

Tullu Guya. (2003). *Caasluga Afaan Oromoo*:Jildi-1; Gumii qormaata Afaan Oromootiin Komishinii 'Aadaafi Turizimii Oromoyattii, Finfinnee.

Zhu, X., and Goldberg,A.B.(2009). Introduction to semi-supervised learning. Synthesis lectures on artificial intelligence and machine learning, 6. Morgan & Claypool publishers.

9. APPENDICES:

APPENDIX A: UNIT TESTS FOR THE CONTEXT FREE GRAMMAR

```
import nltk
from nltk import Non-terminal, non-terminals, Production, CFG
nt1 = Non-terminal('NP')
nt2 = Non-terminal('VP')
…
#all non-terminals assigned these
nt1.symbol() #'NP'
nt1 == Non-terminal('NP') #True
nt1 == nt2 #False
S, NP, VP, PP, DC, ADJP = non-terminals('S, NP, VP, PP, DC, ADJP')
NN, VV, Prep, JJ, CC, IN, Num, NEG, PRN = non-terminals('NN, VV, Prep, JJ, CC, IN, Num, NEG, PRN')

prod1 = Production(S, [NP, VP])
prod2 = Production(NP, [CC, NN])
prod1.lhs() # S
prod1.rhs() # (NP, VP)
prod1 == Production(S, [NP, VP]) #True
prod1 == prod2 # False
#sample of grammar rules and lexicons defined in Appendix A impoerted here
    # Grammar rules
          …
          …
   #   Lexicons
         …
        …
```

APPENDIX B: BOTTOM-UP CHART PARSER RULES OF TRACE VALUE 2

```
Input sentence: yoo fixxe deemi
Tokens: [ 'yoo', 'fixxe', 'deemi']
|.    yoo    .   fixxe   .   deemi   .|
Leaf Init Rule:
|[----------]           .           .| [0:1] 'yoo'
|.           [----------]           .| [1:2] 'fixxe'
|.           .           [----------]| [2:3] 'deemi'
Bottom Up Predict Rule:
|>           .           .           .| [0:0] CC -> * 'yoo'
Single Edge Fundamental Rule:
|[----------]           .           .| [0:1] CC -> 'yoo' *
Bottom Up Predict Rule:
|>           .           .           .| [0:0] DC -> * CC VV
Single Edge Fundamental Rule:
|[---------->           .           .| [0:1] DC -> CC * VV
Bottom Up Predict Rule:
|.           >           .           .| [1:1] VV -> * 'fixxe'
Single Edge Fundamental Rule:
|.           [----------]           .| [1:2] VV -> 'fixxe' *
Bottom Up Predict Rule:
|.           >           .           .| [1:1] VP -> * VV
Single Edge Fundamental Rule:
|[----------------------]           .| [0:2] DC -> CC VV *
|.           [----------]           .| [1:2] VP -> VV *
Bottom Up Predict Rule:
|.           >           .           .| [1:1] S  -> * VP
Single Edge Fundamental Rule:
|.           [----------]           .| [1:2] S  -> VP *
Bottom Up Predict Rule:
|>           .           .           .| [0:0] VP -> * DC VP
Single Edge Fundamental Rule:
|[---------------------->           .| [0:2] VP -> DC * VP
Bottom Up Predict Rule:
|.           .           >           .| [2:2] VV -> * 'deemi'
Single Edge Fundamental Rule:
|.           .           [----------]| [2:3] VV -> 'deemi' *
Bottom Up Predict Rule:
|.           .           >           .| [2:2] VP -> * VV
Single Edge Fundamental Rule:
|.           .           [----------]| [2:3] VP -> VV *
Bottom Up Predict Rule:
|.           .           >           .| [2:2] S  -> * VP
Single Edge Fundamental Rule:
|[==================================]| [0:3] VP -> DC VP *
|.           .           [----------]| [2:3] S  -> VP *
Bottom Up Predict Rule:
|>           .           .           .| [0:0] S  -> * VP
Single Edge Fundamental Rule:
|[==================================]| [0:3] S  -> VP *
Number of edges in chart: 25
Length of parse tree is 1:
Output: (S (VP (DC (CC yoo) (VV fixxe)) (VP (VV deemi))))
Parsing times of parser: 2.923sec
```

APPENDIX C: SAMPLE OF PARSED SENTENCES

```
=== Parsing sentence 1 : osoo hojii hin jalqabiin Waaqayoon kadhadhu
(S
  (VP
    (DC (NP (CC osoo) (NN hojii) (NEG hin)) (VV jalqabiin))
    (VP (NP (NN Waaqayoon)) (VV kadhadhu))))
Bottom-up parser times:  0.062sec
=== Parsing sentence 2 : Caalaan osoo namni isa ilaaluu hin hamummatu
(S
  (NP (NN Caalaan))
  (VP
    (DC (NP (CC osoo) (NN namni) (PRN isa)) (VV ilaaluu))
    (VP (NEG hin) (VV hamummatu))))
Bottom-up parser times:  0.078sec
=== Parsing sentence 3 : osoo beellama qabdu hin taa'iin
(S
  (VP
    (DC (NP (CC osoo) (NN beellama)) (VV qabdu))
    (VP (NEG hin) (VV taa'iin))))
Bottom-up parser times:  0.078sec
=== Parsing sentence 4 : osoo Ayyaana kabajuu hojii idile dagate
(S
  (VP
    (DC (NP (CC osoo) (NN Ayyaana)) (VV kabajuu))
    (VP (NP (NN hojii)) (ADJP (JJ idile)) (VV dagate))))
Bottom-up parser times:  0.016sec
=== Parsing sentence 5 : Gaaddisaan yoo dirqama isaa xumure Finfinne gala
(S
  (NP (NN Gaaddisaan))
  (VP
    (DC (NP (CC yoo) (NN dirqama) (PRN isaa)) (VV xumure))
    (VP (NP (NN Finfinne)) (VV gala))))
Bottom-up parser times:  0.031sec
=== Parsing sentence 6 : osoo Farda gulufsiisu muge
(S
  (VP (DC (NP (CC osoo) (NN Farda)) (VV gulufsiisu)) (VP (VV muge))))
Bottom-up parser times:  0.000sec
=== Parsing sentence 7 : Buna dhuguun rakkoo hin qabu
(S
  (VP
    (DC (NP (NN Buna)) (VV dhuguun))
    (VP (NP (NN rakkoo) (NEG hin)) (VV qabu))))
Bottom-up parser times:  0.016sec
```