ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

COLLEGE OF NATURAL SCIENCE

DEPARTMENT OF INFORMATION SCIENCE

INTEGRATING CLOSED DOMAIN AMHARIC INTO MULTI-LINGUAL TANSLATION GRAMMATICAL FRAMEWORK

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIRMENT

FOR THE DEGREE OF MASTERS OF SCIENCE IN INFORMATION SCIENCE

BY

NEBYOU WOUBSHET WOREDE

JUNE, 2014

ADDIS ABABA

ACKNOWLEDGEMENT

First and foremost, I would like to thank my Lord the almighty God and his Mother St Mary for leading me the way from the beginning to the end. My huge thanks goes out to my adviser Dr. Solomon Teferra for his invaluable support that makes it this all possible. His gentle way of teaching in the class room was the source of my inspiration to work my thesis on NLP.

I also would like to thank my best friend Leggesse Nega for providing me the books that helped me to expand my knowledge on NLP.

ABSTRACT

One alternative approach to multi-lingual translation systems is Grammar-based approach in Grammatical Framework (GF). GF is a grammar formalism which is based on constructive type theory (2). GF mainly targets quality rather than coverage, when applied to build closed domain languages multi-lingual translation applications, it works well and with better quality. The absence of Amharic language from the 72 languages in Google-Translate™, the world number one widely used translation system is one indicator to reveal a broad research has to be made on Amharic NLP. In this study, we have made simple and a bit complex closed domain Amharic multi-lingual translation experiment using a rule based approach in GF. The methodology applied in the experiment involves designing English and Amharic foods grammars, generating active voice sentences as a data source and evaluating the translation accuracy based on sample data from the data source. Our experiment on food domain had passed two levels of experimentation involving simple and a bit complex design, the first covered only sentences of simple active voice singular present tense forms, while in the second level we have introduced paradigms and case handling functions onto the food grammar that expanded the foods grammar translation system capacity to grow by threefold. The advanced experiment involved designing an Amharic and English closed domain foods grammar that generates a set of 32,460 unique plural and singular active voice sentences for each language. Performing a manual translation accuracy test on a randomly chosen 3,000 records accounting 10% of the entire data source, our system achieved a 81% accuracy. The result of our experiment demonstrated the possibility of developing an effective small scale closed domain Amharic multi-lingual translation. However, our research have also discovered developing a robust Amharic multi-lingual translation application in GF requires the full Amharic resource grammar library and its API, thus we recommended further research to be made on improving and completing the partially implemented Amharic resource grammar(1) that currently exists in GF.

ACRONYMS

AL	Applied Linguistic
API	Application Program Interface
BNF	Backus-Naur Format
CL	Computational Linguistics
GF	Grammatical Framework
GNU	General Public License
KeY	The KeY Project
LGNU	Lesser General Public License
MOLTO	Multi Lingual Translation Open Source Project
NLP	Natural Language Processing
SLM	Statistical Language Model
SOV	Subject-Object-Verb
TALK	Multilingual and Multimodal Spoken Dialogue Systems
WebALT	Web Advanced Learning Technologies (www.webalt.net)

TABLE OF CONTENETS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

Introduction

1.1 Background

Language is one of the human capacity, a vital component for acquiring and using complex systems of communication. For a population of 6,800,596,862 an estimated 7,106 living languages exits in the world (1). As humans progress in knowledge through time, the accumulated knowledge of humans mostly recorded using these natural languages. To access, understand and make use of these bulk of accumulated knowledge is a challenge of the present time. In dealing with these challenges only the idea of giving computers the ability to understand and process natural language can come to the intervention.

The main fields of discipline directly involved in language engineering are NLP and Computational Linguistics (CL) where Applied Linguistic (AL) also plays a significant role (2). Natural Language Processing (NLP) is a major filed of computer science, artificial intelligence, and linguistics. NLP linguistics deals with generating and interpreting natural language. Even though computer technologies are still around the corner to produce a system that can deliver accurate, instantaneous translation, or responses. A human-like language processing capability has not yet been achieved. However, the full realization of NLP potential are projected to deliver smart applications that would be able to understand and process natural language to translate languages accurately in real time(3). In NLP there are two types of machine translation approaches a rule based and statistical (4), Both having their own advantages and disadvantages

Language engineering can be defined both in terms of processing languages and in producing languages (5). linguists and computer scientists involved in NLP brought an idea of language engineering that render a computation based representation of human languages formalization of linguistic knowledge into some form of grammatical rules (grammar formalism) (6). Grammatical Framework (GF) is one such grammar formalism which is based on constructive

type theory to express the semantics of natural languages for multilingual grammar applications (7).This framework (GF) has a language library known as the "GF Resource Grammar Library" which is constituted of resource grammars implemented using the GF programming language (8) for various languages. A complete set of morphological paradigms and syntax fragments composed of a common representation , called abstract syntax and concrete syntaxes packed together in the resource grammar(8).

The GF resource grammar project provides resource grammars for thirty five languages. The Amharic version being the eighth has been included in the Grammatical Framework website [1'] implemented by Markos Kassa in 2010. However, the Amharic resource grammar developed by Markos Kassa requires further development to make it a complete Amharic resource grammar. The Amharic resource grammar lacks modules such as Dict module with large-scale morphological lexicon and Irreg module with irregular verbs (6). To make use of the resource grammar in real applications it needs an application program interface (API).

The application grammarians make use of the resource grammars through application programming interfaces (API's) included as abstract syntaxes in the GF library. Therefore, the overall aim of such a library based design is to make it possible for linguistically untrained application programmers to write linguistically correct application grammars encoding the semantics of special domain (8)

Using GF resource grammar numerous applications are available which includes the verification tool KeY(9), the dialogue system research project TALK and the educational project WebALT(10) are major ones. The GF inspired EU project, MOLTO is another ambitious initiative that develops a set of tools for translating texts between multiple languages in real time with high quality. Moreover, the availability of such a library as open-source software under the GNU LGPL helps the aspirations being made to bring less recognized and under resourced languages, such as Amharic, to the world of computation(6).

Amharic (አማርኛ amarəññа) a Semitic language is an official language of Ethiopia, spoken by the vast majority of the ethnic population as a second language[2'], while it

1'www.grammaticalframework.org

2'http://www.csa.gov.et/images/documents/pdf_files/regional/CountryLevel.pdf

has been a mother tongue for the (አማራ amarə) ethnic group constituting one fourth (1/4) of the entire population. It is also the second Semitic language spoken in the world, next to Arabic.

Amharic script is driven from the early script of South Arabia, The script has no capital letters, The changes in the order of the Geez letters from "አበገደ..." to "ሀለሐመ.." and the practice of writing from left to right (unlike Arabic and Hebrew and other Semitic scripts) where believed to be introduced by the clergies that came to Ethiopia accompanying the first Patriarch of Ethiopia from Egypt in the 4th century(11).

Amharic letters consists 33 basic characters , shown in the alphabet on the right in which the vowel "a" is implied. Each of this letters has seven forms indicating different vowels thus ከkɐ ኩku ኪki ካka ኬkɐ ክk ኮko. In addition, there are 45 characters representing certain consonants followed by the sound "w", thus ኰ_"kwa"_, ኲ_"kwi"_ ኳ_"kwa"_ ኴ_"kwɐ"_ (see **Appendeix A**)

At present, despite the ongoing efforts made by the private software industry, the progress on implementing Amharic language specific features such as spell checker, grammar and translation in the form of software and web-services are almost not available. As of writing this paper we observed the free online translator Google-Translate™ included 72 languages, which Amharic is not in the list. Thus, we can conclude that much research work is still awaiting to be done for Amharic Language in the area of computational resources and their usage for parsing and generation in applications. Hence, learning this gap, the purpose of this research is to study the possibility of developing multi-lingual Amharic translation applications using the rule based approach in Grammatical Framework for specific language domains such as greetings, foods, sports and an Amharic phrasebooks.

1.2 Statement of the Problem

Since Language Engineering attracted linguistics and computer specialists involved in NLP, there has been a considerable achievement in producing a computation based representation of human language through formalization of linguistic knowledge into some form of grammatical rule called grammar formalism. Grammatical Framework (GF) is one such open source grammar formalism which is based on constructive type theory to express the semantics of

natural languages for multilingual grammar applications (7). Compared to the statistical machine translation approach that requires large size quality corpora to a successful development of machine translation systems, GF provides application grammarians the ability to write linguistically correct application grammars by encoding the semantics of special domain. So far for over thirty languages the Resource Grammar implementation has been done and included in the GF resource grammar library. Amharic being the eighth in the list is also partially implemented by (6). However , while for most of the languages that have a complete resource grammar library (RGL) and API, different multi-lingual translation applications has been developed for different platforms, such projects include the open source Multi Lingual Translation (MOLTO Project), to be released as open-source libraries which can be plugged in to standard translation tools and web pages and thereby fit into standard workflows. Amharic language having the partially implemented resource grammar library in GF still needs a researchers attention to make use of the opportunity GF provides to the development of Amharic multi-lingual translation systems. This research is initiated to work on a research on Amharic NLP that contributes to the development of Amharic language translation systems in a rule based GF system. Therefore, the purpose of this research is to study the possibility of the development of multi-lingual closed domain Amharic translation applications in Grammatical Framework for specific language domains such as greetings, foods, sports in the absence of a full fledged complete Amharic resource grammar and its API.

1.3 Research Questions:

The study attempts to answer the following questions:

- Is it possible to design closed domain Amharic grammar in GF?
- Is it possible to conduct an experiment on a closed domain Amharic multi-lingual translation system in GF?
- Is it cost and time effective to develop Amharic closed domain translation systems using the rule based GF?

1.4 Objective of the Study

1.4.1 General Objective

The general objective of this research is to investigate the possibility of developing a closed domain multi-lingual translation for Amharic language in GF.

1.4.2 Specific Objectives

The specific objectives of this study are:

- Adapt and design an abstract syntax for food domain.
- Design food domain concrete syntax for Amharic and English languages.
- Perform an experiment on multi-lingual simple food sentence translation between English, Amharic and Italian based on the food grammar.
- Adapt and design an abstract syntax for foods domain.
- Design foods domain concrete syntax for Amharic and English languages.
- Perform an experiment on multi-lingual foods sentence translation between English, Amharic and Italian based on the foods grammar.
- Analyze the translation results in terms of performance.

1.5 Methodology of the Study

To achieve the general and specific objectives of this thesis work, the following methods and tools have been used.

1.5.1 Literature Review

For better understanding of NLP approaches and GF system, researches and background theories related to NLP and GF have been revised through literature review. An extensive review , test and analysis on a previous work on Amharic NLP that partially implemented Amharic Resource grammar in GF have been performed.

1.5.2 Experimentation

The experimentation process passed through the following steps,

- Design Amharic and English food and foods grammar in GF.

- Generate food and foods comment data in GF using the Amharic and English food and foods grammar and export the data it to a text file preserving the Amharic Unicode characters.

- Prepare a portion of the food and foods comments from the data source for a manual translation accuracy test. This involves sorting and removing duplicated data from a randomly selected foods comments to get a 10% sample data..

- Perform a manual English to Amharic food and foods comments translation accuracy test using the 10% randomly selected data.

1.5.3 Evaluation and Analysis

Evaluation and Analysis of the Amharic food and foods multi-lingual translation system performed in terms of performance of the system.

1.5.4 Tools Used

The multi-lingual closed domain translation experiment conducted in this study used the latest GF 3.5 system tool compiled to work on windows environment. For data manipulation ,arrangement and document preparation Notepad text editor, MS Excel and MS Word applications have been used.

1.6 Scope and Limitation of the Study

This study is delimited to investigate through experiment the possibility of developing a closed domain multi-lingual translation for Amharic language in GF based on foods domain. the translation system experiment is limited only to work on sentence level translation and the sentences are of only present active voice forms that involves four pronouns “These,Those,That” and “This”. The manual verification of translation accuracy didn’t involve linguists. If we hadn’t been encountered with time and cost constraints our research could have gone further to experiment on passive voice forms of statements involving articles and more pronouns like “A, The” and “I,We,They,He,She”. Checking the possibility of integrating Amharic close domain multi-lingual translation applications with open source projects such as MOLTO phrasebook project were also in our thoughts.

1.7 Significance of the Study

Concerning Amharic language translation system researches and development, this study will have a major significance of promoting a rule based approach Amharic NLP research and development as an alternative to the much less progressed Amharic NLP researches that have been conducted based on statistical approaches. Our Amharic closed domain multi-lingual translation experimental research process and results will also encourages application grammarian to attempt to develop useful closed domain Amharic multi-lingual translation systems in GF, such multi-lingual application could be a tourist phrasebook which could be used by tourist coming to visit Ethiopia from different parts of the world speaking different languages.

1.8 Organization of the Research Report

The next chapter reviews the general important aspects of grammatical framework (GF), the third chapter review and analyses a previous research that partially implemented an Amharic Resource Grammar in GF. The fourth chapter covers the varies GF inspired multi-lingual translation project developments and application areas using GF platforms. Chapter five presents the main part of the research which involves the design, generating and evaluation experimentation of Amharic foods grammar in GF. The final chapter states the research conclusion and recommendations.

CHAPTER TWO

Grammatical Framework

2.1 Grammatical Framework

Grammatical Framework (GF) is a grammar formalism and a notation for writing grammars based on Martin-L of type theory (12). It is in general a programming language for multilingual grammar applications. It is a special-purpose language for grammars, a functional programming language, like Haskell and Lisp , a development platform for natural language grammars, like LKB and XLE ,a categorical grammar formalism, like ACG and CCG, a logical framework, like Agda and Coq and a platform for machine translation, like Moses and Apertium(13). GF can be used for building translation systems, multilingual web gadgets, natural-language interfaces, dialogue systems, natural language resources (13).In this chapter we present an introduction to GF; a more detail information can be refereed from (7).

The key feature of GF is the representation of a grammar as a pair consisting of an abstract syntax acting as a semantic Interlingua which represents the structure or meaning of values in the language and a concrete syntax which describes the appearance of the abstract syntax. It is this important distinction between abstract syntax and concrete syntax that enabled translation between languages for which a concrete syntax is defined. One or more concrete syntax can exist corresponding to target languages. In addition to this, GF is a functional programming language equipped with a runtime system featuring parsing and verbalization capabilities (14).“GF programs are called grammars, a grammar is a declarative program that defines parsing, generation, and translation” (15). GF is a functional programming language with types and modules (15).

To better explain GF grammars, we consider a very small grammar that describes simple “hello world” grammar, which we adapted from (16), implemented in GF system. a GF system is an

open-source free software, can be downloaded via the GF Home page'[1] grammaticalframework.org.

A GF program, in general, is a **multilingual grammar**. Its main parts are

- an **abstract syntax**
- one or more **concrete syntaxes**

The abstract syntax defines what **meanings** can be expressed in the grammar

- *Greetings*, where we greet a *Recipient*, which can be *World* or *Mum* or *Friends*

A typical GF code for the abstract syntax looks like the following:

```
-- a "Hello World" grammar
abstract Hello = {

  flags startcat = Greeting ;

  cat Greeting ; Recipient ;

  fun
    Hello : Recipient -> Greeting ;
    World, Mum, Friends : Recipient ;
}
```

The code has the following parts:

- a **comment** section starting with" - - "
- a **module header** indicating that it is an abstract syntax module named **Hello**
- a **module body** in braces, consisting of
    - a **startcat flag declaration** stating that Greeting is the default start category for parsing and generation
    - **category declarations** introducing two categories, i.e. types of meanings
    - **function declarations** introducing three meaning-building functions

The English concrete syntax (mapping from meanings to strings) for the above abstract syntax looks:

```
concrete HelloEng of Hello = {
lincat Greeting, Recipient = {s : Str} ;
     lin
        Hello recip = {s = "hello" ++ recip.s} ;
        World = {s = "world"} ;
        Mum = {s = "mum"} ;
        Friends = {s = "friends"} ;
    }
```

The major parts of the above concrete syntax for English language are:

- a **module header** indicating that it is a concrete syntax of the abstract **`syntax Hello`**, itself named **`HelloEng`**
- a **module body** in curly brackets, consisting of
    - **linearization type definitions** stating that **`Greeting`** and **`Recipient`** are **records** with a **string `s`**
    - **linearization definitions** telling what records are assigned to each of the meanings defined in the abstract syntax

Amharic, Finnish and an Italian concrete syntaxes:

```
-- Amharic concrete syntax

concrete HelloAmh of Hello = {
flag code= utf8; -- for representation of Unicode characters set.
      lincat Greeting, Recipient = {s : Str} ;
      lin
        Hello recip = {s = " ሠላም" ++ recip.s} ;
        World = {s = " ለሁሉም"} ;
        Mum = {s = " እምዬ"} ;
        Friends = {s = " ባልንጀሮቼ"} ;
    }


-- Finish concrete syntax

concrete HelloFin of Hello = {
      lincat Greeting, Recipient = {s : Str} ;
      lin
        Hello recip = {s = "terve" ++ recip.s} ;
        World = {s = "maailma"} ;
        Mum = {s = "äiti"} ;
        Friends = {s = "ystävät"} ;
    }
```

```
-- Italian concrete syntax

concrete HelloIta of Hello = {
      lincat Greeting, Recipient = {s : Str} ;
      lin
        Hello recip = {s = "ciao" ++ recip.s} ;
        World = {s = "mondo"} ;
        Mum = {s = "mamma"} ;
        Friends = {s = "amici"} ;
   }
```

Using a GF system in order to compile the grammar in GF, we create four files, one for each module, named ModuleName.gf:

```
Hello.gf  HelloEng.gf   HelloAmh.gf  HelloFin.gf  HelloIta.gf
```

In the GF system, using the command **"import"** the grammars are compiled into an internal representation.

```
> import HelloEng.gf
    - compiling Hello.gf...   wrote file Hello.gfo 8 msec
    - compiling HelloEng.gf...   wrote file HelloEng.gfo 12 msec

>
```

To do a translation between two languages, first we import the concrete syntax of the two languages , in the above example to do a translation from English language to Italian, we use import and linearize commands in the GF system as shown below.

Translation: **pipe** linearization to parsing:

```
> import HelloEng.gf
> import HelloIta.gf
> parse -lang=HelloEng "hello mum" | linearize -lang=HelloIta
ciao mamma
```

The above command in the GF system outputs the translation of **"hello mum**" of English phrase in Italian equivalent **"ciao mamma"**

Translation for all the languages the concrete syntax defined can be achieved by typing the following command into the GF system.

Multilingual generation:

```
> parse -lang=HelloEng "hello friends" | linearize
   ሠላም ባልንጀሮቹ
    terve ystävät
     ciao amici
     hello friends
```

2.2 Abstract and Concrete Syntax

The main distinction between the abstract syntax and concrete syntax is the abstract syntax can be written without taking certain language specific details such as inflection and word order into consideration. Abstracting away smaller details gives an advantage of making the grammars easy to create and maintain as well as to read and understand them(17). .

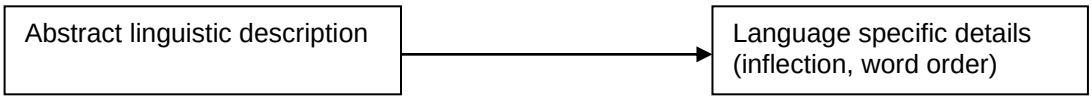

Fig 2.1 : The difference between Abstract and Concrete syntax

The grammar writers, therefore, only focuses on building the structure of the common features to easily analyze and synthesize in a program. The abstract syntax might also be shared between grammars for different languages. Languages that originates from the same roots have more advantage to share a lot in common in the abstract syntax. A set of concrete syntaxes, on the other hand, maps the abstract syntax to different languages through coding linearization rules that handles the details of language specific decisions for the abstract terms. It is possible to define several different concrete syntax mappings for one particular abstract syntax , Usually a number of concrete syntaxes are designed for a single language depending on the application domain.

2.3 Application Grammar and Resource Grammar

Resource Grammars consists syntactic constructs, morphology and general purpose vocabulary of a language which are general grammars to some languages, where the writer of the resource grammar has a working knowledge in the grammar of the target language (18). The possibility of separate compilation of grammar compositions, opens up for writing resource grammars (19). The distinction between Application Grammar and Resource Grammar is described best as “an application grammar has an abstract syntax expressing the semantics of an application domain. A

resource grammar has an abstract syntax expressing linguistic structures. The concrete syntax of an application grammar can be defined as a mapping to the abstract syntax of the resource grammar: it tells what structures are used for expressing semantic object, instead of telling what strings are used."(8)

2.4 Resource Grammar Library

The GF Resource Grammar Library (RGL) freely available and is distributed under the GNU General Public License (GPL). It is a set of natural language grammars built upon a common abstract syntax, that is a common tree structure implemented in GF (16). RGL, designed to be usable by domain experts without linguistic training(16), avoids re-implementing the basic machinery of the target language by domain-specific application grammarians. The application grammarians accesses the abstract syntax in the resource grammar through RGL API. The API allows the grammarians to implement grammars for the languages included in the API. The API also provides tools to extend the resource grammars, using these tools it is possible for grammarians to add more new words to the lexicon(15). The possibility of transferring the experience gained in building one resource grammar in GF to some other resources of similar language has been a big factor for the fast growing interest to build resource grammar library for more languages. RGL have been developed for a number of languages as of august 2013 it has reached 27 languages(14). A range of applications has been developed using RGL such as dialogue systems, tourist phrasebooks, museum object descriptions, pharmaceutical patents and also extended for large-coverage translation with some languages(16).

CHAPTER THREE

Related Works

Despite the existence of a number of researches that dealt with language engineering for over a decade in GF, we only came across one research publication conducted for Amharic under the title "IMPLEMENTING AN OPEN SOURCE AMHARIC RESOURCE GRAMMAR IN GF" by Marko's Kassa in 2010 at the university of Gothenburg. The research performed by Markos can be considered as the first attempt ever made in elevating Amharic language into the GF language engineering research avenue. The research partially achieved to implement Amharic Resource Grammar in GF.

A Resource Grammar in language engineering is the basic component needed for implementing application grammars such as domain based multi-lingual translation applications in GF which can be integrated in MOLTO open source multi lingual translation tools. The primary objective of this thesis is to study the possibility of the development of multi-lingual Amharic translation applications in Grammatical Framework for specific language domains such as greetings, foods, sports, Museum art descriptions and an Amharic phrasebooks. Thus this chapter is dedicated to review the research (6) as to assess and understand how the main product of the research, the Amharic Resource Grammar, can be used to implement domain based Amharic multi-lingual translation applications.

.

3.1 Amharic Grammar

Prior to proceeding to show how the Amharic Language features are implemented in GF formalism, Marko's (6) research went on to discuss the main features of the Amharic language by adapting the major concepts of the Amharic Grammar based on the book written in Amharic (22).

The research extended to give a thumbnail summary of the morphology of Amharic words. The variable features of Nouns in Amharic and their inherent behavior towards gender. an Amharic noun is either masculine or feminine. These declensions are usually achieved through affixation. Suffixal affixations are predominant while there are also a good number of prefixing. an example of affixation of Amharic noun phrase "the houses" (definite, plural noun phrase used as a direct

object) from the sentence “I sold the houses.“ and show the Amharic counter part as follows, .

the houses-Acc - ቤቶቹን - bet-occ-u-n

Here the suffixes **(“-occ“, “-u“ and “-n“)** are added to the head noun **“bet“,** which means house, to mark declensions for number (Pl), definiteness (Def) and case (Acc) to get the required form – the houses - betoccun (ቤቶቹን).

With the exception in second person ‘you’ which may take different agreements when referring to ‘plural’, 'respected (politeness)','singular-female' or 'singular-male' the pronouns in Amharic can be put into three persons as in English.

You (Masc. Sg.) : አንተ - antä

You (Fem. Sg.) : አንቺ - anči

You (Pl.) : እናንተ- ənantä

You (Politeness) : እርስዎ- ərswo

To modify sentences Adjectives in Amharic come before nouns. In a sentence (አንተ በጣም ጐበዝ ተማሪ ነኽ) **“antä bäTam gobäz tamari näh“ – “you are a very clever student“** there are two adjectives **“በጣም/bäTam“** - very and **“ጐበዝ/gobäz“** – clever. **“በጣም/bäTam“-** very modifies **“ጐበዝ/gobäz“** - clever , and **“ጐበዝ/gobäz“** modifies the noun **“ተማሪ/tamari“** – student.

In Amharic verbs are the most complex category of words, they are created generally from consonantal radicals which are inflected by a process of merging with vocalic components based on various patterns. Also there are multi-radical and bi-radical Amharic verbs, the majority of Amharic root words have three radicals. ..

The numerals in Amharic can assume a cardinal or ordinal form. The ordinals are all the times achieved by adding the prefix ኛ/**äňňa** on the Cardinals.

Cardinal Ordinal

ሁለት hulät / twoሁለት + -ኛ ->ሁለተኛ hulät-äňňa/ second

The prepositions appear as simple prepositions that stand alone or as separate entities coming both at pre and post positions.

To – you ለ-አንተ- lä antä (pre)

On - you አንተ-ላይ - antä lay (post)
With – you ከ-አንተ-ጋር- ke antä gar (both pre and post)

Amharic clauses order generally follows Subject-Object-Verb(SOV). Verbs agree with their subjects in number gender and person and objects precede verbs within the verb phrase. The definite article in Amharic is a morphologically bound element which needs a special treatment studying Amharic grammar. enable the user of the library to obtain Amharic concrete sentences like the above just from descriptions made in the abstract syntax.

The research has provided an example of Amharic statement **“ልጁ ይህች ቆንጆ ልጃገረድን ይወዳል።“** being translated to English using the system implemented in GF. the example demonstrated how the GF library user can obtain the Amharic concrete sentence.

ልጁ	ይህች	ቆንጆ	ልጃገረድን	ይወዳል።
l∂ju	**y∂h∂c**	**qonjo**	**l∂jagäräd∂n**	**y∂wäddal**
boy-DEF	this-FEM	beautiful	girl-FEM	loves

“The boy loves this beautiful girl”

The parsing tree representation of the above sentence is given in the fig.2.

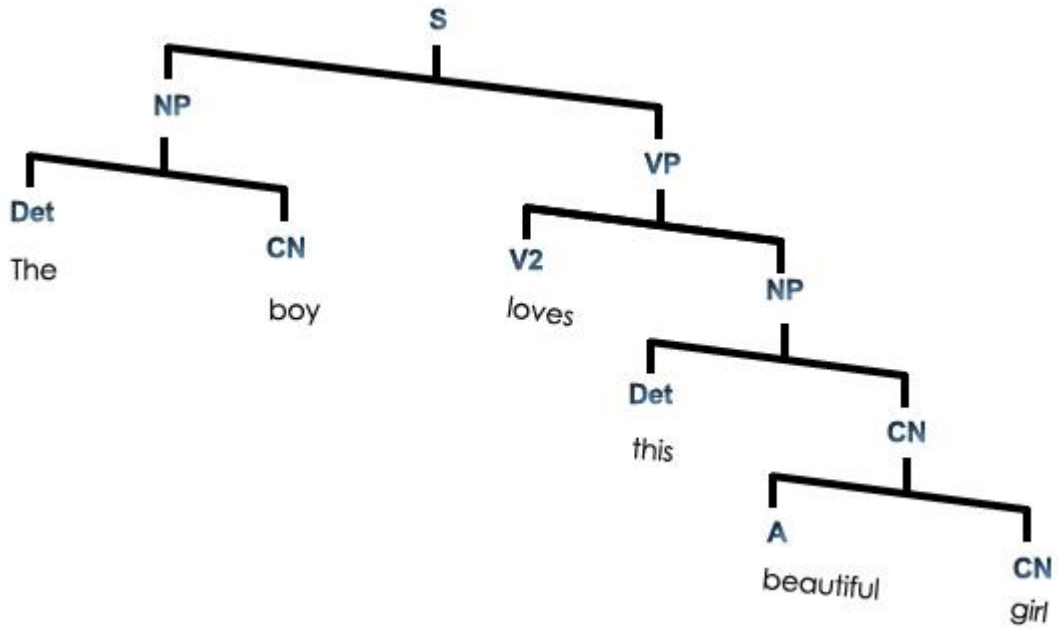

Fig 3.1 : A tree structure representing "The boy loves this beautiful girl"

When examining the rules in the abstract syntax taking the left branch tree that takes a determiner and noun to give a noun phrase has the following type signature.

fun

DetCN: Det -> CN -> NP; (3.1)

Where

DetCN: --the name of the syntax rule

Det: --the types for the determiner and

CN: --the noun component.

For the case of the above Amharic sentence "ልጁ ይህች ቆንጆ ልጃገረድን ይወዳል።/l∂ju y∂h∂c qonjo l∂jagärād∂n y∂wäddal "as the definite marker in Amharic is a morphologically bound element, in the above type signature Det is -ኡ / -u and the CN is l∂j /ልጅ -boy. But these separate entities l∂j and -u are combined to form l∂ju / ልጁ - the boy. In Amharic the gender of the determiner is derived from the noun. In this case in the example ልጃገረድ l∂jagärād/ -girl is a feminine noun and

this is the reason why ይህች/ y∂h∂c- this is selected instead of ይህ/y∂h – (masculine counterpart for this). In the same fashion the number and definiteness features of the noun come from the determiner while case remains usually nominative.

The research implementation further covered the linguistic features such as the morphology, syntax and lexicon. It has also introduced the transliteration technique adopted to handle the need for Unicode infrastructure by Amharic orthography.

3.2 Amharic Orthography

Amharic alphabets composed of thirty three consonants and seven vowels, it is written from left to right. The consonant vowel (CV) combination forms the seven order.

በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ
bä	bu	bi	ba	be	b∂	bo
ተ	ቱ	ቲ	ታ	ቴ	ት	ቶ
tä	tu	ti	ta	te	t∂	to

Vowels stand alone and form unique symbol.

ኧ	ኡ	ኢ	ኣ	ኤ	እ	ኦ
ä	u	I	a	e	∂	o

The implementation of Amharic orthography where demonstrated in the research, an example of a noun declension for a number shows how an orthographically sound word can be formed, adding –occ / -ኦች at the end of the noun ቤት will form

ቤት + -ኦች= ቤት ኦች which the combination of ት and ኦ results in ቶ to form the right written word
=-ቤቶች

bet + -occ = betocc

House Houses

These conversion where achieved by using the regular expression methods in GF, the methods for Amharic are modified to follow the pattern that changes the six order to the seventh order in the case of noun declension for a number or pluralization of a noun.

```
-- replaceLastLet6_7 is a function

oper replaceLastLet6_7 : Str -> Str = word ->

let y = last word in

case y of { "ህ" =>"ሆች" ; "ሌ" => "ልች" ; "ሔ" => "ሕች" ; "ሜ" => "ምች" ; "ሤ" => "ሥች" ; ... "ት" => "ቶች"; ... _ => word+"ዎች" } ;
```

3.3 Amharic Verbal Morphology

In dealing with Verbal Morphology, the research has implemented Amharic verbal morphology using templates (vowel and affix patterns) to modify the root consonant at different positions, doubling the middle consonants, inserting vowels between consonants, adding consonantal affixes, etc. The verbal morphology applied considering those radical roots between two and four. the majority of Amharic verbs are triradicals and marked for person, number, and gender. For example, when the root sbr is applied to a pattern C1aC2aC3a, where C1,C2 and C3 stand for the consonantal radicals forms the perfect stem ሰበረ/säbärä (he broke) is formed, while applying another pattern meC1C2aC3 on the same root word sbr, it forms an Amharic infinitive መስበር/ mäsbär (to break).

The Tense /Aspect/ Mood (TAM) is signaled mainly by the prefixes and suffixes that indicate the subject of the verb but is also reflected in the stem template. For example non-geminated each of the main TAM‟s when considered for the third person singular masculine (Per3 Sg Masc), active (Act) voice. The root of the verb is sbr – to break. .

- Perfective: ሠበረ säbärä / he broke
- Imperfective: ይሠብራል yə-säbr-al / he breaks /he will break
- Jussive: ይሥበር yə-sbär / let him break
- Imperative: ሥበር sbär / break!
- Gerundive: ሠብሮ säbr-o/ having broken

Using the voice signal taking tä as a prefix on the stem säbärä (to break) Passive: ተሠበረ tä-säbärä (he is broken) can be formed, in another case taking 'ä' in between the second and third roots (bear in mind that sbr is the root form). ይሠበራል yə-säbär-al (he will be broken) is formed.

The research has defined three types of verb types: types A, B and C. where the 2nd radical is always geminated regardless of the types. The verb derivations are adapted from.Leslau (1969). In implementing the verbal morphology a triradical verb-inflecting-operation is defined as mkV3gdl for regular verb paradigms for each form and agreement features :

```
mkV3gdl : Str -> Verb = \v ->

let root = getRoot3 v

in {

s = table {

Perf => table {

Per1 Sg => appPattern3 root C1aC2aC3ku;

Per1 Pl => appPattern3 root C1aC2aC3n;

Per2 Sg Masc => appPattern3 root C1aC2aC3k;

Per2 Sg Fem => appPattern3 root C1aC2aC3sh;

Per2 Pl _ => appPattern3 root C1aC2aC3achehu ; ... };

Imperf => table { ... }

...

}

};
```

Using the above conjugating operation mkV3gdl implementation against the root form ‘sbr’ generates the perfective forms as shown below.

Perf Act (Per1 Sg) : ሠበርኩ säbbärku – I broke

Perf Act (Per1 Pl) : ሠበርን säbbärn – We broke

Perf Act (Per2 Sg Masc): ሠበርክ säbbärk – You broke (for Masc)

Perf Act (Per2 Sg Fem) : ሠበርሽ säbbärx –You brike (for Fem)

Perf Act (Per2 Pl Masc): ሠበራችሁ säbbärachu Your broke (for plurals of any gender)-same for : s Perf- Act (Per2 Pl Fem)

Perf Act (Per3 Sg Masc): ሠበረ säbbärä – He broke

Perf Act (Per3 Sg Fem) : ሠበረች säbbäräc – She broke

Perf Act (Per3 Pl Masc): ሠበሩ säbbäru – They broke - same for : Perf Act (Per3 Pl Fem)

...

3.4 Amharic Morphology of Nouns

Numbers of the Amharic nouns plural and singular nouns are implemented in the research by adding plural suffix /-ኦች/-occ or /-ዎች/-wocc, the former for nouns ending in a consonant and the latter for nouns ending in a vowel.

The species/definiteness of the noun is determined by the suffix element attached to the noun is in a consonant or a vowel, singular or plural, and masculine or feminine. The research implemented the three basic suffix added to mark definiteness are /ኡ/- u , /-ው/-w and /-ዋ/-wa . the three suffixes are implemented for nouns to which attached (Masc Sg and ends with a consonant),(Masc Sg and ends with vowel) and (Fem Sg and ends with consonants) respectively.

The definite suffix added to plural nouns ,regardless of gender of the noun, is -u. For example, in the masculine, as in /ንጉሶች/ n∂gusocc "kings" (the plural of n∂gus "king"), the definite form becomes /ንጉሶቹ/n∂gusocc-u „the kings'. In the feminine, as in /ንግስቶች/n∂g∂stocc "queens" (the plural of n∂g∂st "queen"), the definite form becomes /ንግስት -ኦች - ኡ = ንግስቶቹ/ n∂g∂stocc-u "the queens".

3.5 Morphology of the Adjectives

In Amharic Adjectives takes position before the nouns they modify,

እርሱ ሰነፍ ተማሪ ነው ።

ɘrsu sänäf tämari näw
He lazy student is
----- "He is a lazy student."

In this example, the adjective **sänäf "lazy"** precedes the noun **tämari "student"** which it modifies.

In the resource grammar implementation, Adjectives (A) are represented as

A = {s : Gender => Number => Species => Case => Str} ;

Unlike nouns gender is not treated as an inherent feature in the adjectives, one form of adjective can be used to describe both masculine and feminine nouns in some of the cases. Nouns and adjectives inflection have great deal of similarity, the above Adjective (A) implementation builds adjectives tables consisting of 32 (2x2x2x4) forms.

more than 50 regular adjectives in the Amharic Swedish Lexicon list built in a uniform manner using a single paradigm **mkA**.

A lexicographer can, for example, enter a new adjective in this list as shown below:

beautiful_A = mkA "ቆንጆ" ;

where **mkA** is an operation that transfers a given string to an adjective table **(mkA : Str -> A)** inflecting it along the way to give a tabular output such as:

Masc Sg Def Nom : ቆንጆዉ

Masc Sg Def Acc : ቆንጆዉን

Masc Pl Def Acc : ቆንጆዎቹን

Masc Pl Indef Nom : ቆንጆዎች

Fem Sg Def Dat : ለቆንጆዋ

……

So far in this section we have reviewed how the major Amharic Resource Grammar was implemented in GF, for the full understanding of the implementation and further to look in to the remaining parts the Amharic resource grammar implementation, We recommend readers to refer (6). In the next section we provide some usability test on the Amharic Resource Grammar Library.

3.6 The Amharic Grammar Usability Test

The Amharic resource grammar implementation we have shown in section 3.3 to 3.5 has accomplished a lot of works, however it is not yet fully completed to be utilized by Application grammarians. developing multilingual translation application in controlled language applications projects such as Phrasebook for different domains areas requires an API of the resource grammar to which the Phrasebook is targeting. The Amharic resource grammar in its current status can be evaluated using the GF system. For the purpose of testing the Amharic Resource Grammar in this research we have compiled the GF system making sure to include the Amharic Resource Grammar implementation files available in the core system. We obtained the full source package from the grammatical framework main website [3'].

The first step to perform English to Amharic translation test in GF system using the Amharic Resource Library is to import the English and Amharic Language concrete syntax (LangEng.gfo and LangAmh.gfo). This concrete syntaxes inherits the Lexicon (LexiconEng.gfo,LexiconAmh.gfo) and the Grammar (GrammarEng.gfo,GrammarAmh.gfo). Therefore by importing LangEng.gfo and LangAmh.gfo we also imports the lexicon and grammar files.

The test conducted on the Amharic Resource Grammar covers Amharic sentence translation and morphology of nouns and verbs.

3.7 Amharic Sentence Translation Test

Test 1)

```
 >  i LangEng.gfo LangAmh.gfo
```

```
Lang> P –lang=Eng “this beautiful girl is so clever” | l –lang=Amh –to_amharic
```

Lang > ይህች ቆንጆ ልጃገረድ በጣም ብልህ ነች።

[3'] www.grammaticalframework.org/download/index.html

Test 2)

Lang> P –lang=Eng "this girl is so clever" | l –lang=Amh –to_amharic
Lang > ይህች ልጃገረድ በጣም ብልህ ነች።

Test 3)

Lang> P –lang=Eng "she loves wine" | l –lang=Amh –to_amharic

Lang > እርሷ ወይን ትወዳለች።

Test 1 -3 shows an accurate translation using the Amharic resource library, for more accurate translation results refer to **Appendix I** .

Test 4)

Lang> P –lang=Eng "These beautiful girls are so clever" | l –lang=Amh –to_amharic

Lang>እነዚህ ቆንጆዎች ልጃገረዶች በጣም ብልህ ናቸው

Test 5)

Lang> P –lang=Eng "These good girls are so beautiful" | l –lang=Amh –to_amharic

Lang> እነዚህ ጥሩዎች ልጃገረዶች በጣም ቆንጆ ናቸው

Test 4 & 5 confirms the existence of some semantic imperfect translation of English to Amharic using the Amharic resource library, In these examples the erroneous translation obviously resulted from the paradigms module that handles modification of the Amharic adjectives. the Adjective **"ቆንጆ"** gets its number from the noun **"ልጃገረዶች"** and wrongly modified itself into its plural form **"ቆንጆዎች"** , the same way in the second example the Adjective **"ጥሩ"** modified itself into a plural form **"ጥሩዎች"** which resulted in inaccurate translation. To avoid such inaccurate translation the Amharic paradigm that modifies the Amharic adjective needs some revision. for more inaccurate translation results refer to **Appendix F**.

Test 6)

```
Lang> P –lang=Eng "This nice girl is so clever" | l –lang=Amh –
to_amharic

Lang >
```

Test 7)

```
Lang> P –lang=Eng "This girls are so good" | l –lang=Amh –
to_amharic

Lang >
```

Test 6 & 7 outputted an empty result because the input sentence involved the adjective word "nice" which is not found in the Swedish list. the Amharic Resource lexicon was built based on the Swedish list consisting of one hundred eighty nouns, eighty verbs, forty eight adjectives, four adverbs and their corresponding word inflections. In such circumstances sentences that includes words outside the Swedish list wouldn't return a result. However, a lexicographer can expand the Swedish list by adding new words. Because of limitations on the number of words, usually multi-lingual translation applications are not designed only using the Amharic Resource Library, rather the main purpose of the resource library is to make domain based multi-lingual translation applications robust. In the next chapter we will see how the Amharic Resource Grammar Library can be used as a resource by domain based applications. The second test shows a number disagreement of the subject **"Girls"** and the determinant **"this"** which resulted an empty output. for more similar examples refer to **Appendix G**.

3.8 Testing the Amharic Morphology

We started the Amharic Verbal Morphology test by importing the LangAmh.gfo file the same way we did to conduct a test on the English to Amharic sentence translation in section 3.7. The LangAmh.gfo file inherits the important Amharic morphology function MorphoAmh.gfo under ParadigmsAmh.gfo module. The transliteration rule applied in the test is the one implemented by Markos(6) as referred in **Appendix B.**

3.8.1 Verbal Morphology

Verbal morphology is the most challenging task to grammarians to create a robust morphological formula for any language, Amharic being morphologically rich language largely shares the difficulties.

Test 1)

```
> i LangAmh.gfo

Lang> –retain ParadigmsAmh.gfo

Lang> compute_concrete –table mkV "sdb"

 >
```

Perf Act (Per1 Sg) : s'd'bk&'k&/ሠደብኩ
Perf Act (Per1 Pl) : s'd'bn/ሠደብን
Perf Act (Per2 Sg Masc) :s'd'bk/ ሠደብክ
Perf Act (Per2 Sg Fem) : s'd'bx/ ሠደብሽ
Perf Act (Per2 Pl Masc) : s'd'b!ch&/ ሠደባችሁ

....

A total of one hundred forty inflected words produced for the tri-radical root verb **"sdb'/ሠደበ"**, for the full list of the inflection table refer to **Appendix H**. in some cases the Amharic verbs inflection produces words that bears no meaning or none related meaning to the root word, for instance if we take the inflection of the tri-radical verb "nbr**/ነበረ"** , we find a result such as:

Perf Act (Per1 Sg) : n'b'rk&'k&/ነበርኩ
Perf Act (Per1 Pl) : n'b'rn/ነበርን
Perf Act (Per2 Sg Masc) :n'b'rk/ ነበርክ
Perf Act (Per2 Sg Fem) : n'b'rx/ ነበርሽ
Perf Act (Per2 Pl Masc) : n'b'r!ch&/ ነበራችሁ

⋮

Act (Per1 Sg/Pl) : m'nb'r/መንበር
Act (Per2 Sg/Pl Masc/Fem) : m'nb'r/መንበር
Act (Per3 Sg/Pl Masc/Fem) : m'nb'r/መንበር
Pas(Per1 Sg/Pl) : m'nb'r/መንበር
Pas (Per2 Sg/Pl Masc/Fem) : m'nb'r/መንበር
Pas (Per3 Sg/Pl Masc/Fem) : m'nb'r/መንበር

From the above inflection example , we observe that, the inflected word "m'nb'r/**መንበር"** holds none related meaning to the root word "nbr**/ነበረ".**

3.8.2 Amharic Noun

Relatively speaking, compared to verbal morphology formulating the Amharic Noun morphology is an easy task to the grammarians, this is because unlike the verbal morphology the majority of the Amharic Nouns falls under few declension rules.

```
 > i LangAmh.gfo

Lang> –retain ParadigmsAmh.gfo

Lang> compute_concrete –table mkN "ሰው"

 >
```

Sg Def Nom : **ሰው**
Sg Def Acc : **ሰውን**
Sg Def Gen : **የሰው**
Sg Def Dat : **ለሰው**
⋮
Pl Indef Acc : ሰዎችን
Pl Indef Gen : የሰዎች
Pl Indef Dat : ለሰዎች

A total of sixteen declension of words produced for the Noun "sew/**ሰው**", for the full list of the inflection see **Appendix I**. similar to verbs inflection, for exceptional nouns the noun declension produces words that bears irrelevant words, for instance if we take the declension of the uncountable noun **"ዱቄት"** , we find a result such as:

Lang> compute_concrete –table mkN **"ዱቄት"** feminine

>

Sg Def Nom : **ዱቄቱ**
Sg Def Acc : **ዱቄቱን**
Sg Def Gen : **የዱቄቱ**
Sg Def Dat : **ለዱቄቱ**.
.
Pl Indef Acc : **ዱቄቶችን**
Pl Indef Gen : **የዱቄቶች**
Pl Indef Dat : **ለዱቄቶች**

In the above noun declension example, we observe that the uncountable noun **"ዱቄት"** being pluralized by affixing the letter **"ቶች"** but the noun **"ዱቄት"** is plural by its own.

3.8.3 Amharic Adjectives

Except the gender not treated as an inherent feature in adjectives , inflection of Amharic nouns and adjectives have a great deal of similarity(6), a single paradigm **mkA** was enough to build a representation of more than fifty regular adjectives in the Swedish Lexicon list adapted for Amharic language(6).

```
> i LangAmh.gfo

Lang> –retain ParadigmsAmh.gfo

Lang> compute_concrete –table mkA “ቆንጆ”

 >
```

Masc Sg Def Nom : **ቆንጆው**
Masc Sg Def Acc : **ቆንጆውን**
Masc Pl Def Acc : **ቆንጆዎቹን**
⋮
Masc Pl Indef Nom : **ቆንጆዎች**
Fem Sg Def Dat : **ለቆንጆዋ**

Using the **mkA** paradigm function the adjective representation tables will consist thirty two adjective forms.

3.9 Setbacks of the Amharic Resource Grammar

The major setback on the Amharic Resource grammar is lack of an API of the resource grammar, which is the main method of Application software’s to import and use the paradigm functions and categories implemented in the resource grammar. In GF multi-lingual grammar formalism to support different writing systems, a different character sets is required. The simplest set, ASCI, works for the very few languages, while languages like Amharic, Arabic, Hebrew, Hindi and Russian need Unicode Character encoding. The GF system internally supports a 32-bit Unicode characters(15). The default encoding of .gf files is iso-latin-1. To make a 32-bit Unicode characters possible to read and write, a different characters encodings are used that overrides iso-latin-1. All files flagged with UTF-8 character encoding are encoded in UTF-8.

flags coding=utif8;

Many text editors , terminals and web browsers support UTF-8. However for some languages including Amharic, Arabic, Hebrew even though they are flagged with UTF-8 encoding in their

implementation and compiled internally in GF system with a 32-bit Unicode character, the actual characters are not displayed properly or simply the characters are replaced with "?" symbol in GF terminal. To deal with the none ASCII characters display problem a transliteration method that applies a one-to-one mappings between Unicode characters and ASCII characters(possibly strings to two ASCII characters) were used. Using this technique the research just discussed in this chapter Amharic transliteration, see **Appendix B** for full reference of "A- Transliteration Table for the Fidäl as implemented and used by Markos(6) ". While performing an experiment and discussing on domain based Amharic multi-lingual translation in GF in the next chapter, we used the pre-compiled standard transliteration rule that we can view the file in GF by the following GF command.

```
>unicode_table -amharic | wf –file="AmhTransliterationRule.txt"
```

This command extract and exports the Amharic transliteration table into a text file. See **Appendix C** for full reference.

To experiment the Amharic word **"አበባ/Abebä**" in GF's system, using the default standard Amharic transliteration and Markos(1) transliteration file, the word **"አበባ/Abebä**" should be represented as "**ab.b(" and "!b'b!"** respectively.

Therefore for the above reasons we used the Amharic transliteration module to see the actual Fidäl characters during the test, The commands used for this purpose are *put_string*, *-to_amharic* and *from_amharic*.

--The GF command below creates a file name ababa.txt and writes the actual transliterated Fidäl string **"አበባ"** into the file. this is according to the transliteration rule developed by Markos(6).

```
ps –to_amharic "!b'b!" | wf –file="abeba.txt"
```

አበባ

--Using the in-built Amharic transliteration rule in GF, the same result can be achieved.

```
ps –to_amharic "ab.b(" | wf –file="abeba.txt"
```

አበባ

In this chapter we have reviewed and tested extensively the research (6) that partially implemented Amharic Resource grammar in GF, The next chapter reviews GF inspired translation projects which are currently actively undergoing and progressing.

CHAPTER FOUR

GF Inspired Translation Projects

GF has inspired a range of language engineering projects since its inception. Translation systems, multi-lingual web gadgets, natural-language interfaces, dialogue systems, natural language resources are among the many more undergoing development. From these projects, our research work is more interested in the translation system projects, in this regard Molto project gets our big attention.

4.1 Molto Project

MOLTO project is an open source project funded by the European Union since 2008, the main goal of the project is to develop a set of tools that automatically translates documents on the web, with a high quality and between multiple languages in real time[4']. MOLTO's aspiration for high quality translation is not with out a cost, in order to guarantee the quality the coverage is usually restricted to specific domains which can be covered by a grammar, As its main technique MOLTO project uses domain-specific semantic grammars and ontology-based interlinguas(8). MOLTO translation is based on GF and its resource grammar library[4'].

So far MOLTO tools supported more than twelve European languages that includes English, German, and Italian(14).There are also undergoing projects to add more languages in to MOLTO, among them are Arabic and Hibru languages which has the same Semitic root as Amharic language.

MOLTO tool is not the only one in the domain of online translation, there are a number of standards widely used translation tools on the web such as Systran (Babelfish) and Google Translate, these translation tools are based on Statistical Language Model (SLM). The major difference that distinguishes GF MOLTO project from other translating tools is its preferred precision oriented design rather than aiming at large coverage, for these reason MOLTO system focused on translating a domain based translation, MOLTO tools are predestined for translating what it has already customized. A domain can be e-commerce sites, Wikipedia articles, contracts, business letters, user manuals, and Phrasebooks. Domains such translating newspapers and novels are beyond the scope of MOLTO tools.

MOLTO hosted a number of multi-lingual online GF grammar translation applications, the most popular among the applications is the web based Phrasebook application that supports a multi-lingual translation between twenty languages as of writing this report. The Phrasebook consists 46 categories that includes Country, Phrase, Place ,Transport , Question and Words. Fig.3

shows Molto Phrasebook being used to translate an English greeting message "nice to meet you" in twenty languages.

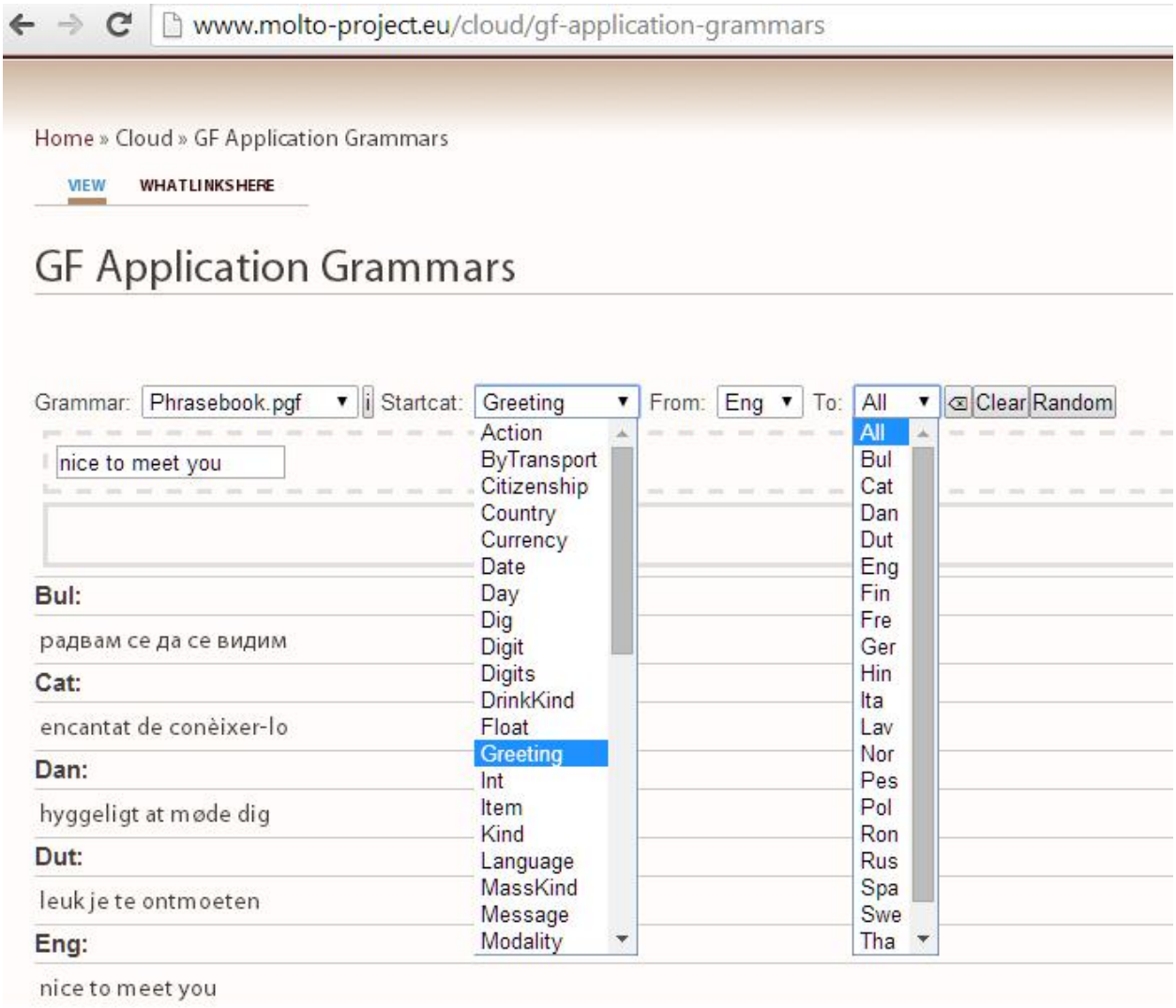

Fig 4.1 : Molto Phrasebook "nice to meet you" greeting translated in to 25 languages.

The main idea behind Molto project is to build an open source domain based multi-lingual translation plug-in tools that can be integrated in to other web applications such as e-commerce sites, tourist websites and smart mobile phone browser. The GF grammar object files plug-in can be compiled in GF to suit the intended platform that hosts the application. The availability of embedded PGF interpreter in Java(the most popular programming language) enhanced the usability of GF in variety of platforms. JavaScript is one of the most widely supported languages in different platforms, almost all browsers supported it including mobile platforms, JavaScript

runnable in web browsers(16), it is usually the preferable scripting language by application programmers to create portable applications. Unlike java, javaScript embedded grammar interpreter doesn't read PGF files thus for GF grammar applications targeting web browsers as the application host needs to compile the GF grammar in to JavaScript code. Fig.4 shows below a Molto project JavaScript-based numeral translator developed for iPhone(16).

Fig 4.2 : A JavaScript-Based numeral translator running on iPhone

4.2 Building Amharic Phrasebook in Molto Project

Adding a language in to MOLTO online translations projects requires utilizing an API of the target language resource grammar library, The Amharic resource grammar we have reviewed in chapter three (6) had been partially implemented, thus to make a progress in Amharic application grammar development in Molto projects, the full implementation of the Amharic language resource grammar is demanding. The full resource grammar library, as of summer 2010 has around 80 categories and 200 abstract syntax functions. Out of these categories and syntax functions the Amharic resource grammar implemented by Markos left features such as case of object suffixes, relative clauses and aspects such as reciprocals and iterative for the various TAM's and voices as a future work(8).

A mini-resource grammar is an alternative to the full implementation of the Amharic resource grammar that can be developed by only implementing the core phrasal categories and functions that helps to produce a working resource grammar. Using the mini-resource grammar application grammarians can get access to the resources through an API. Building the Amharic mini-resource library can be started using resources from the existing Amharic resource grammar implementations.

The next chapter discusses the core part of the research, the experimentation of Amharic Grammar in GF and its results. .

CHAPTER FIVE

Experimentation

This chapter discusses experimentation on Amharic Grammar in GF.The experiment aimed at evaluating the applicability of developing closed domain Amharic Grammar for the purpose of a multi-lingual translation. The Experiment covers two different approaches of building Amharic Grammar in GF. The first one is designed to test the limited translation of food comments which only able to deal with singular present active voice form, Its design requirement is also relatively simple and its data resources generating capacity is bounded by its design. Further enhancing the design by adding additional pronouns and paradigm functions, the experiment widened the translation resource capacity by threefold to accommodate translation on foods comments that include singular and plural present active voice form.

5.1 Domain Specific Grammar Selection

For the purpose of this research experiment, comments about foods has been used. We decided to use comments about food "foods grammar" mainly because a multi-lingual translation implementation of this domain already exists for about 15 languages. The "foods grammar" resource can be found at GF resource library[5'].

Experimenting Amharic Foods grammar on the existing Foods grammar allows us to sidestep the required language experts for the manual multi-lingual translation of the Amharic words used in the foods grammar. Evaluating the accuracy of the Amharic translation against many languages can be performed with a good knowledge of the target language Amharic and one other language from the multi-lingual implementation of foods grammar which in our case the knowledge of English.

5.2 The GF System Preparation

To conduct the GF grammars experiment, we need first to install the GF system. The GF system can be installed in different manners on different platforms more information can be found at the GF home page[6'] on how to install the GF system. The version of GF system we have installed on our windows system for the experimentation is the latest version of GF 3.5 which

[5'] www.gramaticalframework.org/src/

[6'] www.grammaticalframework.org/download/

was released on August 2013. This version is also available for the major computer platforms , Linux (including Ubuntu 32/64bit), Mac OS, and Windows. Our best experience and intense familiarity with windows systems fated our choice of the windows version. A typical properly installed GF command shell interactive window looks as shown in fig.5.1

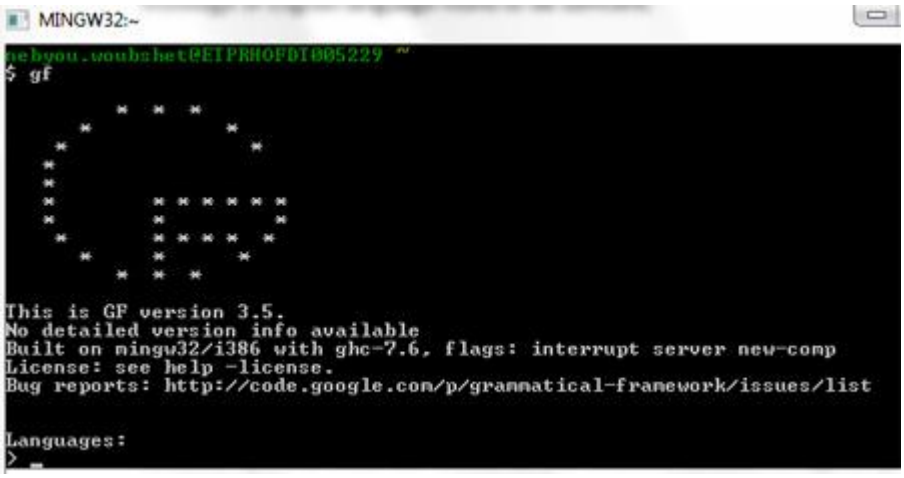

Fig 5.1 : A GF system command line interface

5.3 Food Grammar

The English Foods grammar in the GF resource has been adapted and modified to construct the Amharic foods grammar. The Amharic foods grammar construction achieved by replacing the English words with their corresponding Amharic language dictionary equivalents. fig.5.2a shows English GF-readable BNF grammar for comments about food. The main difference between the Amharic grammar in fig.5.2b constructed for this experiment and the English grammar is the choice of the tokens, We translated the tokens in the English Foods Grammar to its corresponding Amharic tokens.

Pred. Comment ::= Item “is” Quality	Pred. Comment ::= Item "ነው" Quality
This. Item ::= “this” Kind	This. Item ::= "ይህ" Kind
That. Item ::= “that” Kind	That. Item ::= "ያ" Kind
Mod. Kind::= Quality Kind	Mod. Kind ::= Quality Kind
Wine. Kind ::=”wine”	Wine. Kind ::= "ወይን"
Cheese. Kind::= “cheese”	Cheese. Kind ::= "አይብ"
Fish. Kind. ::= “fish”	Fish. Kind ::= "አሣ"
Very. Quality ::= “very” Quality	Very. Quality ::= "በጣም" Quality
Fresh. Quality ::=”fresh”	Fresh. Quality ::= "ትኩስ"
Warm. Quality::= “warm”	Warm. Quality ::= "የሞቀ"
Italian. Quality ::=”Italian”	Italian. Quality ::= "የጣሊያን"
Expensive. Quality ::= “expensive”	Expensive. Quality ::= "ውድ"
Delicious. Quality::=”delicious”	Delicious. Quality ::= "ጥዑም"
Boring. Quality ::=”boring”	Boring. Quality ::= "ዘጊ"
(a)	**(b)**

Fig 5.2 GF-readable BNF food grammar, (a) English grammar , (b) Amharic grammar

In the foods grammar description in fig.5.2 the terms “This.,That.,Mod.,Wine.…”at the left corner are just labels for the constants. Item, Kind and Quality are type identifiers of the their corresponding values. The step leading from BNF to GF is to separate the rules defining abstract syntax trees from the rules telling how trees are linearized (4). The next sub section just does the separation and build the abstract syntax and the concrete syntax for Amharic grammar.

5.4 Abstract and Concrete Food Grammar for Amharic and English

The abstract syntax for Food Food.gf in fig.5.3 is common for all languages implemented the

foods grammar. It defines the categories and the functions telling how the trees are linearized. For example the predication rule

```
Pred. Comment ::= Item "is" Quality
```

Becomes

```
 fuc
Pred :Item -> Quality -> Comment;
```

and the common value types are grouped in to categories

```
Cat
Comment ; Item ; Kind ; Quality ;
```

In the abstract syntax, defining the types of the function specifies the category of arguments to the function, in the above example the function rules defined to expect values from the Item and Quality categories. This fun definition in the abstract syntax will have its corresponding linearization implementation for each targeted languages in the concrete syntax. The concrete syntax for each language is different and can be more than one if need be.

Crafting the Amharic concrete syntax can be achieved simply by copying and making simple modification to the structure of the English food concrete syntax FoodEng.gf shown in fig.5.4. ".gf" is the file extension for the abstract and concrete syntax files. Because of the difference in the word order in Amharic and English construction of clauses a structural modification is required for the copied Amharic abstract FoodAmh.gf grammar. Amharic clauses takes Subject Object Verb order while in English clause construction follows Subject Verb Object word order(6). The following lucid example show the difference.

The boy loves a ball.

Subject Verb Object

ልጁ ኳስ ይወዳል።

Subject Object Verb

```
abstract Food = {
  flags startcat = Comment ;
  cat
    Comment ; Item ; Kind ; Quality ;
  fun
    Pred : Item -> Quality -> Comment ;
    This, That : Kind -> Item ;
    Mod : Quality -> Kind -> Kind ;
    Wine, Cheese, Fish : Kind ;
    Very : Quality -> Quality ;
    Fresh, Warm, Italian,
    Expensive, Delicious, Boring : Quality ;
}
```

Fig 5.3 : Food Abstract syntax

In the foods abstract syntax, The flags tag "**startcat=Comment**" defines "**Comment**" as the beginning of the tree node.

```
concrete FoodEng of Food = {
  lincat
    Comment, Item, Kind, Quality = Str ;
  lin
    Pred item quality = item ++ "is" ++ quality ;
    This kind = "this" ++ kind ;
    That kind = "that" ++ kind ;
    Mod quality kind = quality ++ kind ;
    Wine = "wine" ;
    Cheese = "cheese" ;
    Fish = "fish" ;
    Very quality = "very" ++ quality ;
    Fresh = "fresh" ;
    Warm = "warm" ;
    Italian = "Italian" ;
    Expensive = "expensive" ;
    Delicious = "delicious" ;
    Boring = "boring" ;
}
```

Fig 5.4 : English concrete syntax, FoodEng.gf

```
concrete FoodAmh of Food = {
  flags coding = utf8 ;
  lincat Comment, Item, Kind, Quality = Str ;
  lin
   Pred item quality = item ++  quality ++ "ነው";
   This kind = "ይህ" ++ kind ;
   That kind = "ያ" ++ kind ;
   Mod quality kind = quality ++ kind ;
   Wine = "ወይን" ;
   Cheese = "አይብ" ;
   Fish = "as(" ;
   Very quality = "በጣም" ++ quality ;
   Fresh = "ትኩስ" ;
   Warm = "የሞቀ" ;
   Italian = "የጣሊያን"  ;
   Expensive = "ውድ"  ;
   Delicious = "ጥሑም" ;
   Boring = "ዘጊ" ;
 }
```

Fig 5.6: Amharic concrete syntax FoodAmh.gf.

```
concrete FoodAmh of Food = {
  flags coding = utf8 ;
  lincat Comment, Item, Kind, Quality = Str ;
  lin
   Pred item quality = item ++  quality ++  "n.w";
   This kind = "yh" ++ kind ;
   That kind = "y(" ++ kind ;
   Mod quality kind = quality ++ kind ;
   Wine = "w.yn" ;
   Cheese = "ayb" ;
   Fish = " ayb" ;
   Very quality = "b.T(m" ++ quality ;
   Fresh = "tk-s" ;
   Warm = "y.m?q." ;
   Italian = "y.T(l'y(n"  ;
   Expensive = "wd"  ;
   Delicious = "Tum" ;
   Boring = "z.g'" ;
 }
```

Fig 5.7: Amharic concrete syntax, FoodAmh.gf with transliteration.

5.5 Translating Experiment on Food Grammar in GF

Since we have now built a multi-lingual food grammar for Amharic and English language as shown in section 5.4. we can start the translation experiment between the two languages in the GF shell. The first thing we do is import the abstract and the concrete synatxFoodEng.gf and FoodAmh.gf. The GF system automatically imports the Food.gf as it knows the imported concrete syntaxes are dependent on the abstract syntax Food.gf. The concrete and the abstract syntax then automatically compiled to an object file with a file extension ".gfo". The command used to accomplish this task is as follows

>import FoodAmh.gf FoodEng.gf

food>

GF is now ready to test the translation, let us see how the GF system translates the English sentence "this wine is expensive" in to Amharic

Food> parse –lang=Eng "this wine is expensive" | linearize

ይህ ወይን ውድ ነው

This wine is expensive

food>

The above command **"parse"** parses the English Food grammar tree of the given sentence "this wine is expensive "and transfers the result by piping "|" to be linearized using the command "l" in to all the languages implemented in foods grammar. In this case the linearization result showing the translation of the input sentence in to the Amharic and English. Let us import an Italian Food concrete grammar and repeat the above multi-lingual translation experiment.

food>import FoodIta.gf

Food> p –lang=Eng "this wine is expensive" | l

ይህ ወይን ውድ ነው

This wine is expensive

questo vino è caro

Food>

The linearization process in the above example involved the concrete syntax FoodIta.gf which is used to linearization of the target language parsed tree into Italian grammar.

As a multi-lingual grammar formalism, GF needs to support the different characters set in different writing systems(4). ASCII system works for few languages, iso-latin-1 supports many of the European languages. But for many languages Unicode is required(4). To support the majority of the languages GF is designed to work internally with a 32 bits Unicode characters. With this Unicode facility GF makes it possible to give supports for the Semitic, Slavic,Sino-Tibetian and Japonic languages family members. Amharic being a member of Semitic language family is well established to work in GF under the 32 bit Unicode characters systems. many terminals , text editors and web browsers supports UTF-8 , which encodes Unicode characters by means of variable-size sequences of 8-bits characters(4). Also all files generated by GF such as instances of the ".gfo" object files are encoded in UTF-8 purportedly to set the default inputs and outputs to be in UTF-8 format. The windows version of GF shell has failed to display the Amharic character as viewed in the above translation result "ይህ ወይን ውድ ነው".This is because windows command shell input output system doesn't support UTF-8, non-ASCII characters sets. However if the input is supplied using the transliteration ASCII-to-Unicode characters mapping system, the result can be exported to a text editor by the help of transliteration and data export GF commands. The command below shows Amharic sentence "ይህ ወይን ውድ ነው" argument as a sequence of ASCII characters "yh w.yn wd n.w" as defined in the default GF transliteration file for Amharic.

```
Food> p –lang=Amh "yh w.yn wd n.w" | l

"yh w.yn wd n.w"

This wine is expensive

questo vino è caro

food>
```

It is observed that the multi-lingual translation of the Amharic sentence "yh w.yn wd n.w/ይህ ወይን ውድ ነው" output the same Italian translation result "questo vino è caro" as the English sentence "This wine is expensive" translated. In GF a translation of a sentence that bears similar meaning written in different languages results the same output.

In the above windows environment experiment the Amharic sentence supplied as an argument in a form of transliteration string "yh w.yn wd n.w" output itself in the same format. Transliterating and displaying the result in Amharic characters strings can be achieved by adding an optional command "–to_amharic" after the linearization command "l". The full command looks as follows

Food> p–lang=Eng "this wine is expensive" | l –lang=Amh –to_amhric

ይህ ወይን ውድ ነው

food>

The optional command –lang=Amh inserted next to the linearization command "l" dictates the GF system to parse and linearize only the Amharic grammar. If this option is left blank, the result will be generated to all the languages the concrete syntax implemented and imported in the running GF system.

Another approach in windows GF shell is to export the output into a text file where the Amharic translated outputs can be represented in the actual Amharic alphabets ፊደል/ fidäl.

Food> p–lang=Eng "this wine is expensive" | l –lang=Amh –to_amharic | wf –file="wine.txt"

food>

The translation result "**ይህ ወይን ውድ ነው**" saved in the text file named wine .txt.

In GF system, the multilingual food grammar implementation generates twelve thousand two hundred forty separate food comments for each language.

Food> gt | l

Practically, the above figure constitutes the set of sentences that holds the total number of valid sentences in the food grammars. It is from only these valid sentences in the set that can be supplied as an argument to the translation system. Otherwise, GF responds blank sentence for any arguments which doesn't exactly map to one of the instances in the sentences set. However, it is important to note that valid inputs doesn't always bear a meaningful sentences.

Out of the total 12,460 generated sentences, manually evaluating a sample of 1000 randomly generated unique sentences, we found out that there are as many as 274 records that doesn't give any meaningful senses. This number accounts close to 22% of the total number of sentences in the sentence set. Some sentences such as "**that boring fish is very delicious/ያ ዘጊ አሣ በጣም ጥሑም ነው** " are grammatically correct but conveys a contradictory message. such sentences are discovered in significant amount and identified as the

most difficult problems to solve. If we take an English sentence **"this warm warm wine is very very very Italian"** is one of the grammatically incorrect sentences generated by GF but still is a valid sentence as an argument. It is also observable that the corresponding literal translation of this meaningless English sentence to different languages are also effects of grammatically incorrect meaningless sentences. The above translation of the awkward English sentence in to Amharic results a meaningless Amharic sentence **"ይህ የሞቀ የሞቀ ወይን በጣም በጣም የጣሊያን ነው"**. This is a lesson we have learned in the experiment that in GF translation system a garbage input always produce a garbage output. **Table 5.1** shows some of the valid arguments but the grammatically incorrect statements generated by GF.

	English	**Amharic**
Sentence	this wine is very very very Italian	ይህ ወይን በጣም በጣም በጣም የጣሊያን ነው
	this warm warm wine is very very expensive	ይህ የሞቀ የሞቀ ወይን በጣም በጣም የጣሊያን ነው
	that boring fish is very very delicious	ያ ዘጊ አሣ በጣም በጣም ጥኡም ነው
	that Italian fish is Italian	ያ የጣሊያን አሣ የጣሊያን ነው
	that very boring cheese is delicious	ያ በጣም ዘጊ ዘጊ አይብ ጥኡም ነው

Table 5.1: Grammaticaly incorrect sentences generated by GF for food grammar acounting 10% of the total set of sentences.

	English	**Amharic**
Sentence	this wine is Italian	ይህ ወይን የጣሊያን ነው
	that expensive cheese is boring	ያ ውድ አይብ ዘጊ ነው
	this wine is expensive	ይህ ወይን ውድ ነው
	this wine is very expensive	ይህ ወይን በጣም ውድ ነው
	that fish is fresh	ያ አሣ ትኩሥ ነው
	this fish is delicious	ይህ አሣ ጥኡም ነው
	that fresh fish is delicious	ያ ትኩሥ አሣ ጥኡም ነው

Table 5.2 : Meaningful sentences and their translations generated by GF for food.

To save time on typing sentences, we performed a random test on the accuracy of the translation using the GF command "gr", the command generates random sentence for linearization, **Appendix D** lists some of the sentences randomly generated and linearized sentences.

Food> gr | l

ያ አይብ በጣም ውድ ነው
That cheese is very expensive
quel formaggio è caro

In GF translation some of these awkward sentences are resulted because the design of the food grammar doesn't handle cases of different scenarios or doesn't employ smart paradigm functions. The grammatically incorrect awkward sentences in some cases are generated because of generic handling of adjectives. The adjective **"very"** modifies itself as "very very" or modifying another Adjective such as warm as **"very warm"** forms a correct grammar, while the same pattern applied to the adjective **"warm"** it produces an incorrect meaningless grammar **"warm warm"** and **"warm very"**. Such design issues can be solved by introducing paradigm functions in the concrete syntax. in most cases these paradigm functions have a specific implementation to each language. All words belonging to closed category with in the lexical classification can be completely enumerated. The word "very" is part of this category where a special case handling paradigm function are devised for this category.

```
food> gt -number=20 | l -lang=Amh -to_amharic
```

ያ አይብ ዘገ ነው

ያ አይብ ጣፋጭ ነው

ያ አይብ ውድ ነው

ያ አይብ ትኩሥ ነው

ያ አይብ የጣሊያን ነው

ያ አይብ በጣም ዘገ ነው

ያ አይብ በጣም ጣፋጭ ነው

ያ አይብ በጣም ውድ ነው

ያ አይብ በጣም ትኩሥ ነው

ያ አይብ በጣም የጣሊያን ነው

ያ አይብ በጣም በጣም ዘገ ነው

ያ አይብ በጣም በጣም ጣፋጭ ነው
ያ አይብ በጣም በጣም ውድ ነው
ያ አይብ በጣም በጣም ትኩሥ ነው
ያ አይብ በጣም በጣም የጣሊያን ነው
ያ አይብ በጣም በጣም በጣም ዘገ ነው
ያ አይብ በጣም በጣም በጣም ጣፋጭ ነው
ያ አይብ በጣም በጣም በጣም ውድ ነው
ያ አይብ በጣም በጣም በጣም ትኩሥ ነው
ያ አይብ በጣም በጣም በጣም የጣሊያን ነው
Food>

Table 5.3: The first 20 sentences generated

5.6 Foods Grammar

With the closed domain food grammar design and implementation we have discussed under section 5.4, we where only able to do translations of singular active voice sentences involving only three food Items, Six expressive words and one linking verb that generates 12,460 unique food comments. In this section by increasing the food grammar items by one adding the food item **"Pizza"**, and adding additional pronouns "these" and "those" we will show how plural forms of the foods sentence translation become possible. The modifications on the food abstract syntax (Food.gf) provides us the new foods abstract syntax (Foods.gf) as described in fig.5.8. The addition of new item and new pronouns in the abstract foods grammar increased the total number of sentences generated from 12,640 sentences in food grammar to 32,640 sentences in foods grammar, for the later more than 20,000 of them are projected meaningful sentences.

The structural changes we have made on the abstract syntax are relatively minimal, but on the concrete syntax of each language FoodsEng.gf and FoodsAmh.gf shown in fig.5.9 respectively, the changes are a bit complex that involves adding paradigm functions, morphology of Nouns and case handling of nouns/number agreements.

```
abstract Foods = {
  flags startcat=Phrase ;
  cat
    Phrase ; Item ; Kind ; Quality ;
  fun
    Is : Item -> Quality -> Phrase ;
    This, That, These, Those : Kind -> Item ;
    Kind : Quality -> Kind -> Kind ;
    Wine, Cheese, Fish, Pizza : Kind ;
    Very : Quality -> Quality ;
    Fresh,Warm, Italian, Expensive, Delicious, Boring :Quality ;
}
```

Fig 5.8: Foods abstract syntax (Foods.gf) upgraded from Food abstract syntax Food.gf.

```
--# -path=.:prelude

concrete FoodsEng of Foods = open Prelude in {

  lincat

    Phrase, Quality = SS ;

    Kind = {s : Number => Str} ;

    Item = {s : Str ; n : Number} ;



  lin

    Is item quality = ss (item.s ++ copula ! item.n ++ quality.s) ;

    This  = det Sg "this" ;

    That  = det Sg "that" ;

    These = det Pl "these" ;

    Those = det Pl "those" ;

    QKind quality kind = {s = \\n => quality.s ++ kind.s ! n} ;

    Wine = regNoun "wine" ;

    Cheese = regNoun "cheese" ;

    Fish = noun "fish" "fish" ;

    Pizza = regNoun "pizza" ;

    Very = prefixSS "very" ;

    Fresh = ss "fresh" ;

    Warm = ss "warm" ;

    Italian = ss "Italian" ;

    Expensive = ss "expensive" ;

    Delicious = ss "delicious" ;

    Boring = ss "boring" ;



  param

    Number = Sg | Pl ;



  oper

    det : Number -> Str -> {s : Number
```

```
--# -path=.:prelude

concrete FoodsAmh of Foods = open Prelude in {

  lincat

    Phrase, Quality = SS ;

    Kind = {s : Number => Str} ;

    Item = {s : Str ; n : Number} ;



  lin

    Is item quality = ss (item.s ++ quality.s ++ copula ! item.n) ;

    This  = det Sg "yh" ;

    That  = det Sg "y(" ;

    These = det Pl "en.z'h" ;

    Those = det Pl "en.z'y(" ;

    QKind quality kind = {s = \\n => quality.s ++ kind.s ! n} ;

    Wine = regNoun "w.yn" ;

    Cheese = regNoun "ayb" ;

    Fish = noun "as(" "as(w?c";

    Pizza = regNoun "p'z(" ;

    Very = prefixSS "b.T(m" ;

    Fresh = ss "tk-s" ;

    Warm = ss "m-q";

    Italian = ss "y.T(l'y(n" ;

    Expensive = ss "wd" ;

    Delicious = ss "T(f(C" ;

    Boring = ss "z.g'" ;



  param

    Number = Sg | Pl ;



  oper

    det : Number -> Str -> {s : Number => Str} -> {s : Str ; n : Number} =
```

```
=> Str} -> {s : Str ; n : Number} =
      \n,d,cn -> {
        s = d ++ cn.s ! n ;
        n = n
      } ;
    noun : Str -> Str -> {s : Number
=> Str} =
      \man,men -> {s = table {
        Sg => man ;
        Pl => men
        }
      } ;
    regNoun : Str -> {s : Number =>
Str} =
      \car -> noun car (car + "s") ;
    copula : Number => Str =
      table {
        Sg => "is" ;
        Pl => "are"
        } ;
}
```

(a)

```
      \n,d,cn -> {
        s = d ++ cn.s ! n ;
        n = n
      } ;

    noun : Str -> Str -> {s : Number => Str} =
      \man,men -> {s = table {
        Sg => man ;
        Pl => men
        }
      } ;
    regNoun : Str -> {s : Number => Str} = \w ->
 let
  ws:Str =case w of {
x + "n"=> x + "n?c";
x + "t"=> x + "t?c";
x + "b"=> x + "b?c";
x + "s"=> x + "s?c";
_ => w + "w?c"
}
in
noun w ws;
    copula : Number => Str =
      table {
        Sg => "n.w" ;
        Pl => "n(c.w"
        } ;
}
```

(b)

Fig 5.9 Foods concrete syntax, (a) in English , (b) in Amharic

The new modified Foods grammar allows the parsing and linearization of plural forms of comments on foods. The translation of sentences like "These wines are delicious/**እነዚህ ወይኖች ጣፋጭ ናቸው**" , "Those cheeses are expensive/**እነዚያ አይቦች ውድ ናቸው**" become possible. Below is some examples of foods grammar system, for more examples refer to **Appendix E.**

Test 1)

```
Food> p–lang=Eng "these wines are delicious" | l –lang=Amh –to_amhric
```

እነዚህ ወይኖች ጣፋጭ ናቸው

Test 2)

```
Food> p–lang=Eng "those cheeses are expensive" | l –lang=Amh –to_amhric
```

እነዚያ አይቦች ውድ ናቸው

Test 3)

```
Food> p–lang=Eng "these boring wines are expensive" | l –lang=Amh –to_amhric
```

እነዚህ ዘጊ ወይኖች ውድ ናቸው

Test 4)

```
Food> p–lang=Eng "those Italian pizzas are fresh" | l –lang=Amh –to_amhric
```

እነዚያ ትኩሥ የጣሊያን ፒዛዎች ውድ ናቸው

5.7 Analysis of the Multi-lingual Food and Foods Grammar Experiment

5.7.1 Food Grammar Data Preparation

The food comment data is generated in GF using the food grammar by importing the abstract and concrete syntaxes of the food grammar and using the **`geretae_tree`** and **`linerization`** command. 12,460 unique food comments were generated. A sample of 1,000 English sentences with their corresponding Amharic translation records again randomly generated and reserved for the manual translation accuracy evaluation. Ms Access is used for sorting the sentences and removing duplicated sentences.

5.7.2 Food Grammar Experiment Analysis

In our first experiment on food grammar, our grammar formalism design consists the most common ten food items and seven common terms used to express the quality of a food item. With these closed domain food items and terms GF builds more than twelve thousand present tense singular active voice form Amharic food comments. Manually evaluating a sample of one thousand randomly generated and linearized sentences approximately 78% of the sentences are meaningful, while the rest 22% are either grammatically or semantically incorrect sentences. **"that fish is very Italian/ያ አሣ በጣም የጣሊያን ነው"** and **"that boring fish is very delicious/ያ ዘጊ አሣ በጣም ጥሑም ነው "** is an example of grammatically and semantically incorrect sentences. Since the food comments multi-lingual translation system is assumed to be operated by human interactions, inputting grammatically incorrect sentences are very minimal.

In spite of the food translation system built in GF based on closed food domain have limitations on its coverage. The experiment showed a encouraging high translation quality. For a sample of 1,000 valid sentences a 84% accurate multi-lingual translations between English and Amharic language achieved.

5.7.3 Foods Grammar Data Preparation

By following a similar GF procedure that generated food comments, we generated 32,460 unique foods comments. A sample of 3,000 English sentences with their corresponding Amharic translation records randomly generated and reserved for a manual translation accuracy evaluation. Ms Access is used for sorting the sentences and removing duplicated food sentences.

5.7.4 Foods Grammar Experiment

The second experiment was made by modifying the food abstract syntax to add more items and pronouns. A paradigm function has also been added on the English and Amharic food concrete syntax to expand the food grammar formalism into foods grammar that handle more foods comments including present tense plural active voice forms. With the newly added items and paradigm functions in the foods grammar GF builds more than thirty two thousand foods comments that includes present tense plural active voice forms. The addition of the paradigm function in the concrete syntax allows the foods translation system to be able to translate more foods comments such as **"these pizzas are Italian/እነዚህ ፒዛዎች የጣሊያን ናቸው"** and **"those Italian wines are very expensive/ እነዚያ የጣሊያን ወይኖች በጣም ውድ ናቸው"**. The evaluation of the translation of a sample of 3,000 randomly selected valid English sentences generated by foods grammar achieved a 81% accurate translation into Amharic. Table 5.4 shows the statistical data between the food grammar and foods grammar.

Table 5.4 Food and Foods grammar data analysis.

	Type of Sentences	**Sentence Generated**	**Randomly Selected Sentences for Manual Evaluation**	**Translation Success Rate, Based on the Sample Evaluation.**
Food grammar	-Present tense singular active voice form.	12,460 unique records	1,000	84%
Foods grammar	-Present tense singular active voice form. -Present tense plural active voice form.	32,640 unique records	3,000	81%

Although the food and foods multi-lingual translation system experiment in GF demonstrated the feasibility of developing a closed domain Amharic translation systems. We don't dare to declare these systems are meant to fully satisfy the foods domain environment. We understand maximizing the utilization of close domain translations systems in GF demands to consider many aspects of sentence translation such as gender, person, number and polarity. To see how Amharic multi-lingual translation can go further in a robust foods domain environment, It needs further modifying and redesigning the foods grammar to handle different cases that in particular concerns Amharic language.

Thus, in order to engage issues concerning Amharic language in a broader way either the full implementation of the Amharic resource grammar has to be achieved or an alternative Amharic mini-resource grammar must be produced, which both tasks are beyond the scope of this paper. However, our experiment on foods grammar can show a case for how these tasks can be achieved by using and modifying an existing resource grammars which are already implemented for other languages. Both the partially implemented Amharic resource grammar and mini-resource grammars implemented for many other languages are available in the GF main website http://www.gramaticalframework.org.

CHAPTER SIX

Conclusion and Recommendation

6.1 Conclusion

The emergence of NLP, a computerized approach to analyzing text that is based on both a set of theories and a set of technologies(24) still remains a very active area of research and development among linguist and computer science professionals. related to processing and linguistic analysis of text a set of reusable language components and resources has been developed for a number of languages that includes tokenizers, stemmers, POS taggers, lemmatizers, named entity recognizers, term extractors, surface syntactic analysers, parsers and computational lexica. Likewise, Amharic Language lined itself among the few world languages that have their own unique writing system, attracted many researches from both linguistic and computer science professionals. For Amharic NLP many of the researches were conducted based on statistical approaches which as a result produced a number of research outputs and sill attracting new researchers. Despite the growing number of research out puts a real progress was much slower mainly because of lack of large size Amharic corpora. The success of researches based on statistical approach is basically depends on the size and quality of the corpus involved in the research, this problem has been reported by many researchers involved in Amharic NLP. Developing a large size quality corpora requires a pragmatic knowledge in linguistics as well as a coordinated hard work and effort of professionals from different disciplines. To make a real progress in Amharic NLP researches developing a large size quality corpora is an essential task.

Mean while, another alternate to statistical NLP approach is a Grammar-based approach such as GF. GF specifically designed targeting quality rather than coverage sets restriction to provide large coverage in an open text translation as compared to the statistical approach. But when applied for a closed domain languages multi-lingual translation it works well and with good quality.

Our experiment on string-type Amharic grammar formalism for a food domain demonstrated our initial hypothesis that developing grammar formalism for closed domain Amharic language in GF is quite possible. Where application grammarians can follow a similar pattern further dealing with the core aspects of sentence translation such as gender, person, number and polarity agreements that in particular concerns Amharic language. Our experiment shows how GF makes it simple to describe complicated linguistic issues such as sentence structure and word inflection.

However, we also understand developing a robust closed domain Amharic multi-lingual translation applications can only be easily achieved when a fully implemented Amharic resource grammars libraries and its API is available.

Languages that have the full resource grammar in GF have reached over 28 as of August 2013. These languages provide application grammarians with a fully fledged readily available API. Because of these API's writing a robust multi-lingual translation applications for these languages in major platforms become possible. The open source application grammar implemented in MOLTO online translations project is one example. With regard to Amharic resource grammar, the partially implemented Amharic resource grammar by Markos(1) has already laid the foundation for the fulfillment of the goal towards engineering fully fledged Amharic resource grammar library, further research and development is yet needs to come to implement more key categories in to the library and improve some of the existing Amharic morphological paradigms and case handling functions up to the limits where countering full features of the Amharic language becomes possible.

Although the paramount task ahead by Amharic GF researchers should be conducting continuous research on producing a fully fledged complete Amharic Resource grammar, it is understood that the realization process requires a lot of time and resources. However, in some cases developing language applications doesn't require the utilization of all the categories and functions implemented in the resource grammar. Out of the existing total 80 categories and 200 abstract syntax functions of the GF grammar resource library, a mini-resource grammar for a language can be created by only implementing some of the most relevant categories and functions of the target language.

Our conclusion is that, the Amharic foods grammar experiment we had performed with the absence of Amharic resource grammar and its API had exhibited the cost and time effectiveness of Amharic multi-lingual translation development in the rule based GF system, this is proved by the relatively simplicity of designing Amharic grammar in GF. The performance of the Amharic foods grammar multi-lingual sentence translation system had also showed there is a lot of potential for Amharic language in GF to develop a variety of simple closed multi-lingual translation systems that could be useful in our everyday life.

6.2 Recommendation

We recommend Amharic grammarians to get engaged in the following research areas.

- Conduct research on top of this research by introducing more food Items, personal pronouns "I,We,He,She,They" , definite articles "The,A", linking verbs "was,were" and regular verbs "eat,want,hate,like".
- Conduct a research towards improving and completing the existing partially implemented Amharic resource grammar by Markos (1).
- Conduct a research for the implementation of Amharic-mini resource grammar and its API. Availability of a mini-Amharic resource grammar could allow application grammarians to develop a variety of medium size domain specific Amharic language multi-lingual translation applications . such applications can be museum object descriptions , greetings, foods comments, sport commentary, product descriptions and many more diverse multi-lingual translation applications for different platforms.
- Conduct a research on the possibility of integrating Amharic close domain multi-lingual translation applications with open source GF projects such as MOLTO phrasebook project.
- Conduct a research on the possibility of developing a closed domain multi-lingual translation in GF involving the major local languages that exists in Ethiopia.

- Conduct a research to identify a closed domain language areas where developing a multi-lingual GF translation application is applicable and useful.

REFERENCES

1. Lewis, M. Paul, Gary F. Simons, and Charles D. Fennig (eds.).: Ethnologue: Languages of the World, Seventeenth edition Dallas, Texas: SIL International. Online version: http://www.ethnologue.com. (2014).

2. Sue Ellen Wright.: Trends In Language Engineering. University Of The Saarland (1998)

3. Grishman, Ralph.: Computational Linguistics: An Introduction. New York: Cambridge University Press. (1994)

4. Ranta, Aarne .: Gf Tutorial For Resource Grammar Writers. http://www.grammaticalframework.org/doc/resource-tutorial.pdf. (2011)

5. Sager, J. C.: Language Engineering And Translation: Consequences Of Automation. Amsterdam ; Philadelphia : J. Benjamins Pub. Co. (1994)

6. Markos Kassa Gobena. Implementing An Open Source Amharic In Gf (Master Of Science Thesis In Intelligent Systems Design). Chalmers University of Technology University of Gothenburg. (2010)

7. Ranta, Aarne.: Grammatical Framework: A Type Theoretical Grammar Formalism. The Journal Of Functional Programming 14 (2). (2004)

8. Ranta, Aarne.: The Gf Resource Grammar Library: Linguistic Issues In Language Technology. CSLI Publications . (2009)

9. Bernhard Beckert , Daniel Bruns , Ralf Küsters , Christoph Scheben , Peter H. Schmitt , Tomasz Truderung , Bernhard Beckert , Daniel Bruns , Ralf Küsters , Christoph Scheben , Peter H. Schmitt , Tomasz Truderung.: The KeY Approach for the Cryptographic Verification of JAVA Programs. Karlsruhe Institute of Technology. (2006)

10. Olga Caprotti.: Society for Information Technology & Teacher Education International Conference Volume: 2006, Issue: 1. (2006)

11. Aleka Kidanewolde Kifle." Mezgebe Fidäl /መዝገበ ፊደል. Artistic Publishing House. (1934)

12. P. Martin-L¨Of.: Intuitionistic Type Theory. Naples: Bibliopolis . (1984)

13. Thomas Hallgren, Krasimir Angelov,Aarne Ranta.: Grammatical Framework Tutorial. (2012)

14. Ramona Enache.:. Frontiers Of Multilingual Grammar Development: Thesis For The Degree Of Doctor Of Philosophy at University of Gothenburg. (2013)

15. Thomas Hallgren, Krasimir Angelov,Aarne Ranta.: Grammatical Framework Tutorial. (2012)

16. Aarne Ranta.: Grammatical Framework Tutorial. http://www.grammaticalframework.org/doc/tutorial/gf-tutorial.html#toc10 . (2013)

17. Peter Ljungl.: Expressivity And Complexity Of The Grammatical Framework: Thesis For The Degree Of Doctor Of Philosophy At University of Gothenburg. (2004)

18. J. Khegai.: Language Engineering In Grammatical Framework Licentiate Thesis, Chalmers University Of Technology, Gothenburg, Sweden. (2003)

19. Ranta, Aarne.: Modular Grammar Engineering In GF. Chalmers University of Technology and Göteborg University. (2004)

20. Peng Chen.: Implementation Of Chinese Resource Grammar In Grammatical Framework .International Journal Of Knowledge And Language Processing . (2013)

21. Mois_Es Salvador Meza Moreno and Bjorn Bringert.: Interactive Multilingual Web Applications With Grammatical Framework. Chalmers University Of Technology And Göteborg University. (2008)

22. YIMAN, BAYE.: የአማርኛስዋሰው· ADDIS ABABA. (e.m.p.d.a press.). (1987)

23. http://www.molto-project.eu/content/faq/what-languages-are-there-molto

24. Liddy, E.D.: Natural Language Processing. In Encyclopedia of Library and Information Science, 2nd Ed. NY. Marcel Decker, Inc. (2001)

APPENDICES

Appendix - **A** · Amharic alphabets *fidäl* :

	ä [ə]	u	i	a	e	ə [ɨ]	o	ʷä [ʷə]	ʷi	ʷa	ʷe	ʷə [ʷɨ]
h	ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ					
l	ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ			ሏ		
h	ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ			ሗ		
m	መ	ሙ	ሚ	ማ	ሜ	ም	ሞ			ሟ		
s	ሠ	ሡ	ሢ	ሣ	ሤ	ሥ	ሦ			ሧ		
r	ረ	ሩ	ሪ	ራ	ሬ	ር	ሮ			ሯ		
s	ሰ	ሱ	ሲ	ሳ	ሴ	ስ	ሶ			ሷ		
š	ሸ	ሹ	ሺ	ሻ	ሼ	ሽ	ሾ			ሿ		
q	ቀ	ቁ	ቂ	ቃ	ቄ	ቅ	ቆ	ቈ	ቍ	ቋ	ቌ	ቍ
b	በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ			ቧ		
v	ቨ	ቩ	ቪ	ቫ	ቬ	ቭ	ቮ			ቯ		
t	ተ	ቱ	ቲ	ታ	ቴ	ት	ቶ			ቷ		
č	ቸ	ቹ	ቺ	ቻ	ቼ	ች	ቾ			ቿ		
h	ኀ	ኁ	ኂ	ኃ	ኄ	ኅ	ኆ	ኈ	ኍ	ኋ	ኌ	ኍ
n	ነ	ኑ	ኒ	ና	ኔ	ን	ኖ			ኗ		
ñ	ኘ	ኙ	ኚ	ኛ	ኜ	ኝ	ኞ			ኟ		
'	አ	ኡ	ኢ	ኣ	ኤ	እ	ኦ			ኧ		
k	ከ	ኩ	ኪ	ካ	ኬ	ክ	ኮ	ኰ	ኵ	ኳ	ኴ	ኵ
h	ኸ	ኹ	ኺ	ኻ	ኼ	ኽ	ኾ					
w	ወ	ዉ	ዊ	ዋ	ዌ	ው	ዎ					
'	ዐ	ዑ	ዒ	ዓ	ዔ	ዕ	ዖ					
z	ዘ	ዙ	ዚ	ዛ	ዜ	ዝ	ዞ			ዟ		
ž	ዠ	ዡ	ዢ	ዣ	ዤ	ዥ	ዦ			ዧ		
y	የ	ዩ	ዪ	ያ	ዬ	ይ	ዮ					
d	ደ	ዱ	ዲ	ዳ	ዴ	ድ	ዶ			ዷ		
ǧ	ጀ	ጁ	ጂ	ጃ	ጄ	ጅ	ጆ			ጇ		
g	ገ	ጉ	ጊ	ጋ	ጌ	ግ	ጎ	ጐ	ጕ	ጓ	ጔ	ጕ
t	ጠ	ጡ	ጢ	ጣ	ጤ	ጥ	ጦ			ጧ		
č	ጨ	ጩ	ጪ	ጫ	ጬ	ጭ	ጮ			ጯ		
p	ጰ	ጱ	ጲ	ጳ	ጴ	ጵ	ጶ			ጷ		
ṣ	ጸ	ጹ	ጺ	ጻ	ጼ	ጽ	ጾ			ጿ		
s	ፀ	ፁ	ፂ	ፃ	ፄ	ፅ	ፆ					
f	ፈ	ፉ	ፊ	ፋ	ፌ	ፍ	ፎ			ፏ		
p	ፐ	ፑ	ፒ	ፓ	ፔ	ፕ	ፖ			ፗ		
	ä [ə]	u	i	a	e	ə [ɨ],	o	ʷä [ʷə]	ʷi	ʷa	ʷe	ʷə [ʷɨ]

Appendix - B - Transliteration Table for *fidäl* adapted from "Markos Kasa 2010: IMPLEMENTING AN OPEN SOURCE AMHARIC"
Format: | *UTF code no* | *fidäl character* | *transliteration in the system* |

| 1200 | ሀ | h' |	1225 | ጱ| s |	| 126a | ቚ| v# |	| 1297 | ኗ | n7 |	|12d4 | ዔ | ä% |	| 1302 | ፚ| j# |	| 1337 | ጁ| P7 |
|1201 | ሁ | h& |	|1226 | ጶ| s/ |	| 126b | ቛ | v! |	| 1298 | ኘ | N' |	| 12d5 | ዕ| ä |	|1303 | ፛ | j! |	| 1338 | ጰ | S' |
|1202 | ሂ | h# |	|1227 | ጷ | s7 |	|126c | ቜ| v% |	| 1299 | ኙ | N& |	|12d6 | ዖ | ä/ |	|1304 | ፜ | j% |	| 1339 | ጱ| S& |
|1203 | ሃ | h! |	|1228 | ጺ | r' |	|126d | ቝ | v |	| 129a | ኚ | N# |	|12d8 | ዏ | z' |	|1305 | ፝ | j |	| 133a | ጲ | S# |
|1204 | ሄ | h% |	|1229 | ፈ | r& |	|126e | ቞ | v/ |	| 129b | ኛ | N! |	|12d9 | ዐ | z& |	|1306 | ፞ | j/ |	| 133b | ጳ | S! |
|1205 | ህ | h |	|122a | ፋ | r# |	|126f | ቟ | v7 |	| 129c | ኜ | N% |	|12da | ዑ | z# |	|1307 | ፟ | j7 |	|133c | ጴ| S% |
|1206 | ሆ | h/ |	|122b | ፌ | r! |	|1270 | ተ | t' |	| 129d | ኝ | N |	|12db | ዙ | z! |	|1308 | ጎ | g' |	|133d | ጵ | S |
|1207 | ሇ| h7 |	|122c | ፍ| r% |	|1271 | ቱ | t& |	|129e | ኞ | N/	| 12dc | ዚ | z% |	|1309 | ጏ | g& |	|133e | ጶ | S/ |
|1208 | ለ| l' |	|122d | ፎ | r |	|1272 | ቲ | t# |	||129f | ኟ | N7	| 12dd | ዘ | z |	|130a | ጊ | g# |	|133f | ጷ | S7 |
|1209 | ሉ | l& |	|122e | ፏ | r/ |	|1273 | ታ | t! |	|| 12a0 | ኡ | ! |	| 12de | ዞ | z/ |	|130b | ጋ | g! |	|1348 | ፌ | f' |
|120a | ሊ | l# |	|122f | ፐ | r7 |	|1274 | ቴ | t% |	|12a1 | ኡ | & |	| 12df | ዟ | z7 |	|130c | ጌ | g% |	|1349 | ፋ | f& |
|120b | ሌ | l! |	|1238 | ፘ | x' |	|1275 | ት | t |	|12a2 | ኢ | # |	|12e0 | ዠ | Z' |	|130d | ግ | g |	|134a | ፌ | f# |
|120c | ላ | l% |	|1239 | ፙ | x& |	|1276 | ቶ | t/ |	|12a3 | ኣ | A |	|12e1 | ዡ | Z& |	|130e | ጎ | g/ |	|134b | ፋ | f! |
|120d | ሌ | l |	|123a | ፚ | x# |	|1277 | ቷ | t7 |	|12a4 | ኤ | % |	| 12e2 | ዣ | Z# |	|1313 | ጓ | g7 |	|134c | ፌ | f% |
|120e | ል | l/ |	|123b | ፛ | x! |	|1278 | ቸ | c' |	|12a5 | ኣ | (|	| 12e3 | ዤ | Z! |	|1320 | ጠ | T' |	|134d | ፍ | f |
|120f | ሎ| l7 |	|123c | ፜ | x% |	|1279 | ቹ | c& |	|12a6 | ኦ | / |	| 12e4 | ዥ | Z% |	|1321 | ጠ| T& |	|134e | ፎ | f/ |
|1210 | ሏ | H' |	|123d | ፝ | x |	|127a | ቺ | c# |	|12a7 | ኧ | ' |	12e5 | ዣ| Z |	|1322 | ጡ| T# |	|134f | ፏ| f7 |
|1211 | ሐ | H& |	|123e | ፞ | x/ |	|127b | ቻ | c! |	|12a8 | ከ | k' |	| 12e6 | ዧ| Z/ |	|1323 | ጣ | T! |	|1350 | ጥ | p' |
|1212 | ሑ| H# |	|123f | ፟ | x7 |	|127c | ቼ | c% |	|12a9 | ኩ | k& |	| 12e7 | ዤ| Z7 |	|1324 | ጤ| T% |	|1351 | ጥ | p& |
|1213 | ሒ| H! |	|1240 | ቀ | q' |	|127d | ች | c |	|12aa | ኪ | k# |	| 12e8 | ዠ | y' |	|1325 | ጥ | T |	| 1352 | ጢ | p# |
|1214 | ሓ | H% |	|1241 | ቁ | q& |	|127e | ቾ | c/ |	|12ab | ካ | k! |	| 12e9 | ዧ| y& |	|1326 | ጦ | T/ |	| 1353 | ጧ | p! |
|1215 | ሔ | H |	|1242 | ቂ | q# |	|127f | ቿ | c7 |	|12ac | ኬ | k% |	|12ea | ዪ | y# |	|1327 | ጧ| T7 |	| 1354 | ጨ | p% |
|1216 | ሕ | H/ |	|1243 | ቃ | q! |	|1280 | ኀ | X' |	|12ad | ክ | k |	|12eb | ዬ | y! |	|1328 | ጨ| C' |	| 1355 | ጣ | p |
|1217 | ሖ | H7 |	|1244 | ቄ | q% |	|1281 | ኁ | X& |	|12ae | ኮ | k/ |	|12ec | ይ | y% |	|1329 | ጩ | C&	| 1356 | ጧ| p/ |
|1218 | ሗ | m' |	|1245 | ቅ | q |	1282 | ኂ | X# |	|12b3 | ኳ | k7 |	|12ed | ያ | y |	|132a | ጩ | C# |	| 1357 | ጪ | p7 |
|1219 | መ | m& |	|1246 | ቆ | q/ |	|1283 | ኃ | X! |	|12c8 | ዐ | w' |	|12ee | ዮ | y/ |	|132b | ጫ | C! |		
121a | ሙ | m# |	|1247 | ቇ | q7 |	|1284 | ኄ | X% |	|12c9 | ዑ | w&	|12ef | ዯ | y7 |	|132c | ጬ C%		
|121b | ሚ | m! |	|1260 | በ | b' |	1285 | ኅ | X |	|12ca | ዒ | w# |	| 12f0 | ደ | d' |	|132d | ጭ | C |		
|121c | ማ | m% |	|1261 | ቡ | b& |	|1286 | ኆ | X/ |	|12cb | ዋ | w! |	| 12f1 | ዱ | d& |	|132e | ጮ | C/ |		
121d | ሜ | m |	|1262 | ቢ | b# |	|128b | ኋ | X7 |	|12cc | ዌ | w% |	| 12f2 | ዲ | d# |	|132f | ጯ | C7 |		
|121e | ም | m/ |	|1263 | ባ | b! |	|1290 | ነ | n' |	|12cd | ው | w |	| 12f3 | ዳ | d! |	|1330 | ጸ | P' |		
|121f | ሞ | m7 |	|1264 | ቤ | b% |	|1291 | ኑ | n&	|12ce | ዎ | w/ |	| 12f4 | ዴ | d% |	| 1331 | ጹ | P& |		
|1220 | ሟ | s' |	|1265 | ብ | b |	|1292 | ኒ | n# |	|12cf | ዏ | w7 |	| 12f5 | ድ | d |	| 1332 | ጺ | P# |		
|1221 | ሡ | s& |	|1266 | ቦ | b/ |	|1293 | ና | n! |	|12d0 | ዐ | ä' |	| 12f6 | ዶ | d/ |	|1333 | ጻ | P! |		
|1222 | ሡ | s# |	|1267 | ቧ | b7 |	|1294 | ኔ | n% |	|12d1 | ዑ | ä& |	| 12f7 | ዷ| d7 |	|1334 | ጼ | P% |		
|1223 | ሢ | s! |	|1268 | ቭ | v' |	| 1295 | ን | n |	|12d2 | ዒ | ä# |	| 1300 | ጀ | j' |	| 1335 | ጽ | P |		
| 1224 | ሣ | s% |	| 1269 | ቮ| v& |	| 1296 | ኘ | n/ |	|12d3 | ዓ | ä! |	1301 | ጁ| j& |	| 1336 | ጾ | P/ |		

Appendix - C - the default pre-compiled Amharic Fidäl Transliteration Table > `ut -amharic`

Format: | *UTF code no* | *fidäl character* | *transliteration in the system* |

| 1200 | ሀ | h. |	| 1225 | ፉ | s |	| 126a | ቚ | v' |	| 1297 | ኗ | n* |	| 12dc | ፚ | z) |	| 1309 | ጉ | g- |	| 133e | ፘ | S? |
| 1201 | ሁ | h- |	| 1226 | ፑ | s? |	| 126b | ቛ | v(|	| 1298 | ኘ | N. |	| 12dd | ፝ | z |	| 130a | ጊ | g' |	| 133f | ፙ | S* |
| 1202 | ሂ | h' |	| 1227 | ፒ | s* |	| 126c | ቜ | v) |	| 1299 | ኙ | N- |	| 12de | ፞ | z? |	| 130b | ጋ | g(|	| 1348 | ፈ | f. |
| 1203 | ሃ | h(|	| 1228 | ፈ | r. |	| 126d | ቝ | v |	| 129a | ኚ | N' |	| 12df | ፟ | z* |	| 130c | ጌ | g) |	| 1349 | ፉ | f- |
| 1204 | ሄ | h) |	| 1229 | ፉ | r- |	| 126e | ቞ | v? |	| 129b | ኛ | N(|	| 12e0 | ፠ | Z. |	| 130d | ግ | g |	| 134a | ፊ | f' |
| 1205 | ህ | h |	| 122a | ፊ | r' |	| 126f | ቟ | v* |	| 129c | ኜ | N) |	| 12e1 | ፡ | Z- |	| 130e | ጎ | g? |	| 134b | ፋ | f(|
| 1206 | ሆ | h? |	| 122b | ፋ | r(|	| 1270 | ተ | t. |	| 129d | ኝ | N |	| 12e2 | ። | Z' |	| 1313 | ጓ | g* |	| 134c | ፌ | f) |
| 1207 | ሇ | h* |	| 122c | ፌ | r) |	| 1271 | ቱ | t- |	| 129e | ኞ | N? |	| 12e3 | ፣ | Z(|	| 1320 | ጠ | T. |	| 134d | ፍ | f |
| 1208 | ለ | l. |	| 122d | ፍ | r |	| 1272 | ቲ | t' |	| 129f | ኟ | N* |	| 12e4 | ፤ | Z) |	| 1321 | ጡ | T- |	| 134e | ፎ | f? |
| 1209 | ሉ | l- |	| 122e | ፎ | r? |	| 1273 | ታ | t(|	| 12a0 | አ | a |	| 12e5 | ፥ | Z |	| 1322 | ጢ | T' |	| 134f | ፏ | f* |
| 120a | ሊ | l' |	| 122f | ፏ | r* |	| 1274 | ቴ | t) |	| 12a1 | ኡ | u |	| 12e6 | ፦ | Z? |	| 1323 | ጣ | T(|	| 1350 | ፐ | p. |
| 120b | ላ | l(|	| 1238 | ፐ | x. |	| 1275 | ት | t |	| 12a2 | ኢ | i |	| 12e7 | ፧ | Z* |	| 1324 | ጤ | T) |	| 1351 | ፑ | p- |
| 120c | ሌ | l) |	| 1239 | ፑ | x- |	| 1276 | ቶ | t? |	| 12a3 | አ | A |	| 12e8 | ፨ | y. |	| 1325 | ጥ | T |	| 1352 | ፒ | p' |
| 120d | ል | l |	| 123a | ፒ | x' |	| 1277 | ቷ | t* |	| 12a4 | ኤ | E |	| 12e9 | ፩ | y- |	| 1326 | ጦ | T? |	| 1353 | ፓ | p(|
| 120e | ሎ | l? |	| 123b | ፓ | x(|	| 1278 | ቸ | c. |	| 12a5 | እ | e |	| 12ea | ፪ | y' |	| 1327 | ጧ | T* |	| 1354 | ፔ | p) |
| 120f | ሏ | l* |	| 123c | ፔ | x) |	| 1279 | ቹ | c- |	| 12a6 | ኦ | o |	| 12eb | ፫ | y(|	| 1328 | ጨ | C. |	| 1355 | ፕ | p |
| 1210 | ሐ | H. |	| 123d | ፕ | x |	| 127a | ች | c' |	| 12a7 | ኧ | e* |	| 12ec | ፬ | y) |	| 1329 | ጩ | C- |	| 1356 | ፖ | p? |
| 1210 | ሐ | H. |	| 123d | ፕ | x |	| 127a | ች | c' |	| 12a7 | ኧ | e* |	| 12ec | ፬ | y) |	| 1329 | ጩ | C- |	| 1356 | ፖ | p? |
| 1211 | ሑ | H- |	| 123e | ፖ | x? |	| 127b | ቻ | c(|	| 12a8 | ከ | k. |	| 12ed | ፭ | y |	| 132a | ጪ | C' |	| 1357 | ፗ | p* |
| 1212 | ሒ | H' |	| 123f | ፗ | x* |	| 127c | ቼ | c) |	| 12a9 | ኩ | k- |	| 12ee | ፮ | y? |	| 132b | ጫ | C(|		
| 1213 | ሓ | H(|	| 1240 | ቀ | q. |	| 127d | ች | c |	| 12aa | ኪ | k' |	| 12ef | ፯ | y* |	| 132c | ጬ | C) |		
| 1214 | ሔ | H) |	| 1241 | ቁ | q- |	| 127e | ች | c? |	| 12ab | ካ | k(|	| 12f0 | ፰ | d. |	| 132d | ጭ | C |		
| 1215 | ሕ | H |	| 1242 | ቂ | q' |	| 127f | ቿ | c* |	| 12ac | ኬ | k) |	| 12f1 | ፱ | d- |	| 132e | ጮ | C? |		
| 1216 | ሖ | H? |	| 1243 | ቃ | q(|	| 1280 | ኀ | X. |	| 12ad | ክ | k |	| 12f2 | ፲ | d' |	| 132f | ጯ | C* |		
| 1217 | ሗ | H* |	| 1244 | ቄ | q) |	| 1281 | ኁ | X- |	| 12ae | ኮ | k? |	| 12f3 | ፳ | d(|	| 1330 | ጰ | P. |		
| 1218 | መ | m. |	| 1245 | ቅ | q |	| 1282 | ኂ | X' |	| 12b3 | ኳ | k* |	| 12f4 | ፴ | d) |	| 1331 | ጱ | P- |		
| 1219 | ሙ | m- |	| 1246 | ቆ | q? |	| 1283 | ኃ | X(|	| 12c8 | ኰ | w. |	| 12f5 | ፵ | d |	| 1332 | ጲ | P' |		
| 121a | ሚ | m' |	| 1247 | ቇ | q* |	| 1284 | ኄ | X) |	| 12c9 | ኲ | w- |	| 12f6 | ፶ | d? |	| 1333 | ጳ | P(|		
| 121b | ማ | m(|	| 1260 | በ | b. |	| 1285 | ኅ | X |	| 12ca | ኴ | w' |	| 12f7 | ፷ | d* |	| 1334 | ጴ | P) |		
| 121c | ሜ | m) |	| 1261 | ቡ | b- |	| 1286 | ኆ | X? |	| 12cb | ኵ | w(|	| 1300 | ጀ | j. |	| 1335 | ጵ | P |		
| 121d | ም | m |	| 1262 | ቢ | b' |	| 128b | ኋ | X* |	| 12cc | ኸ | w) |	| 1301 | ጁ | j- |	| 1336 | ጶ | P? |		
| 121e | ሞ | m? |	| 1263 | ባ | b(|	| 1290 | ነ | n. |	| 12cd | ኺ | w |	| 1302 | ጂ | j' |	| 1337 | ጷ | P* |		
| 121f | ሟ | m* |	| 1264 | ቤ | b) |	| 1291 | ኑ | n- |	| 12ce | ኾ | w? |	| 1303 | ጃ | j(|	| 1338 | ጸ | S. |		
| 1220 | ሠ | s. |	| 1265 | ብ | b |	| 1292 | ኒ | n' |	| 12cf | ዀ | w* |	| 1304 | ጄ | j) |	| 1339 | ጹ | S- |		
| 1221 | ሡ | s- |	| 1266 | በ | b? |	| 1293 | ና | n(|	| 12d8 | ዘ | z. |	| 1305 | ጅ | j |	| 133a | ጺ | S' |		
| 1222 | ሢ | s' |	| 1267 | ቧ | b* |	| 1294 | ኔ | n) |	| 12d9 | ዙ | z- |	| 1306 | ጆ | j? |	| 133b | ጻ | S(|		
| 1223 | ሣ | s(|	| 1268 | ቨ | v. |	| 1295 | ን | n |	| 12da | ዚ | z' |	| 1307 | ጇ | j* |	| 133c | ጼ | S) |		
| 1224 | ሤ | s) |	| 1269 | ቩ | v- |	| 1296 | ኖ | n? |	| 12db | ዛ | z(|	| 1308 | ገ | g. |	| 133d | ጽ | S |		

Appendix D: list of sentences randomly generated and linearized from food grammar.

	English	**Amharic**
Sentence	that fish is boring	ያ አሣ ዘጊ ነው·
	that fish is delicious	ያ አሣ ጥኡም ነው·
	that wine is boring	ያ ወይን ዘጊ ነው·
	that Italian wine is very warm	ያ የጣሊያን ወይን በጣም የሞቀ ነው·
	that wine is very Italian	ያ ወይን በጣም የጣሊያን ነው·
	This fish is expensive	ይህ አሣ ውድ ነው·
	This wine is fresh	ይህ ወይን ትኩሥ ነው·
	This fresh wine is Italian	ይህ ትኩሥ ወይን የጣሊያን ነው·
	That boring expensive አሣ is boring	ያ ዘጊ ውድ አሣ ዘጊ ነው·
	This Italian fish is warm	ይህ የጣሊያን አሣ ሙቅ ነው·
	This delicious wine is Italian	ይህ ጣፋጭ ወይን የጣሊያን ነው·
	this very boring wines is fresh	ይህ በጣም ዘጊ ወይን ትኩስ ነው·
	This cheese is very boring	ይህ አይብ በጣም ዘጊ ነው·

Appendix E: list of sentences randomly generated and linearized from foods grammar.

	English	Amharic
Sentence	This fresh wines are delicious	እነዚህ ትኩስ ወይኖች ጣፋጭ ናቸው
	That pizza is Italian	ያ ፒዛ የጣሊያን ነው
	This fish is fresh	ይህ አሳ ትኩስ ነው
	That warm fish is fresh	ያ ሙቅ አይብ ትኩስ ነው
	Those cheeses are fresh	እነዚያ አይቦች ትኩስ ናቸው
	Those cheeses are fresh	እነዚያ አይቦች ዘጊ ናቸው
	These fish are very expensive	እነዚህ አሳዎች በጣም ውድ ናቸው
	This fish is very Italian	ይህ አሳ በጣም የጣሊያን ነው
	Those boring fish are very delicious	እነዚያ ዘጊ አሳዎች በጣም ጣፋጭ ናቸው
	Those wines are very expensive	እነዚያ አይቦች በጣም ውድ ናቸው
	That pizza is warm	ያ ፒዛ ሙቅ ነው
	These fresh cheeses are expensive	እነዚህ ትኩስ አይቦች ውድ ናቸው
	These expensive expensive cheese is fresh	ይህ ውድ ውድ አይብ ትኩስ ነው
	Those warm pizzas are boring	እነዚያ ሙቅ ፒዛዎች ዘጊ ናቸው
	That boring expensive wine is boring	ያ ዘጊ ውድ ወይን ዘጊ ነው
	This Italian cheese is warm	ይህ የጣሊያን አይብ ሙቅ ነው
	This delicious pizza is Italian	ይህ ጣፋጭ ፒዛ የጣሊያን ነው
	These very boring wines are fresh	እነዚህ በጣም ዘጊ አይቦች ትኩስ ናቸው
	This cheese is very boring	ይህ አይብ በጣም ዘጊ ነው

Appendix F: list of sentences inaccurately linearized using the Amharic Resource Grammar.

	English	**Amharic**
Sentence	These clever boys love beer	እነዚህ ብልሆች ልጆች ፍቅር ቢራ እነዚህ ብልሆች ልጆች ቢራ ይወድዳሉ
	Those bad boys drink beer	እነዚያ መጥፎዎች ልጆች ቢራ ይጠጣሉ
	The young boys and girls are very clever	ወጣቶቹ ልጆች እና ልጃገረዶች በጣም ብልህ ናቸው
	The youngest boys and girls are very clever	ከሁሉ ወጣቶቹ ልጆች እና ልጃገረዶች በጣም ብልህ ናቸው
	Those dirty young boys are bad	እነዚያ ቆሻሻዎች ወጣቶች ልጆች መጥፎዎች ናቸው
	They broke the rules	ህጎቹን ሠበሩ እነርሱ ህጎቹን ሠበሩ

Appendix G: list of empty outputs using the Amharic Resource Grammar.

	English	**Amharic**
Sentence	this boys love beer	Empty result due to the number disagreement
	they are bad boy	>>
	these boys and girls are nice	Empty result due to missing the word “nice in the Amharic Lexicon.
	the car is brand new	Empty result due to missing the word “brand” in the Amharic Lexicon.

Appendix H: A total of one hundred forty inflected words produced for the tri-radical root verb "sdb'/□□□" using the Amharic resource grammar .

Perf Act (Per1 Sg) : ሠደብኩ
Perf Act (Per1 Pl) : ሠደብን
Perf Act (Per2 Sg Masc) : ሠደብክ
Perf Act (Per2 Sg Fem) : ሠደብሽ
Perf Act (Per2 Pl Masc) : ሠደባችሁ
Perf Act (Per2 Pl Fem) : ሠደባችሁ
Perf Act (Per3 Sg Masc) : ሠደበ
Perf Act (Per3 Sg Fem) : ሠደበች
Perf Act (Per3 Pl Masc) : ሠደቡ
Perf Act (Per3 Pl Fem) : ሠደቡ
Perf Pas(Per1 Sg) : ተሠደብኩ
Perf Pas(Per1 Pl) : ተሠደብን
Perf Pas(Per2 Sg Masc) : ተሠደብክ
Perf Pas(Per2 Sg Fem) : ተሠደብሽ
Perf Pas(Per2 Pl Masc) : ተሠደባችሁ
Perf Pas(Per2 Pl Fem) : ተሠደባችሁ
Perf Pas(Per3 Sg Masc) : ተሠደበ
Perf Pas(Per3 Sg Fem) : ተሠደበች
Perf Pas(Per3 Pl Masc) : ተሠደቡ
Perf Pas(Per3 Pl Fem) : ተሠደቡ
Imperf Act (Per1 Sg) : ዕሠድባለሁ
Imperf Act (Per1 Pl) : ዕንሰድባለን
Imperf Act (Per2 Sg Masc) : ትሠድባለህ
Imperf Act (Per2 Sg Fem) : ትሰድብያለሽ
Imperf Act (Per2 Pl Masc) : ትሠድባላችሁ
Imperf Act (Per2 Pl Fem) : ትሠድባላችሁ
Imperf Act (Per3 Sg Masc) : ይሠድባል
Imperf Act (Per3 Sg Fem) : ትሠድባለች
Imperf Act (Per3 Pl Masc) : ይሰድባሉ
Imperf Act (Per3 Pl Fem) : ይሰድባሉ
Imperf Pas(Per1 Sg) : ዕሠደባለሁ
Imperf Pas(Per1 Pl) : ዕንሠደባለን
Imperf Pas(Per2 Sg Masc) : ትሠደባለህ
Imperf Pas(Per2 Sg Fem) : ትሠደቢያለሽ
Imperf Pas(Per2 Pl Masc) : ትሠደባላችሁ
Imperf Pas(Per2 Pl Fem) : ትሠደባላችሁ
Imperf Pas(Per3 Sg Masc) : ይሠበራል
Imperf Pas(Per3 Sg Fem) : ትሠደባለች
Imperf Pas(Per3 Pl Masc) : ይሠደባሉ
Imperf Pas(Per3 Pl Fem) : ይሠደባሉ
Jus_Imperat Act (Per1 Sg) : ልሥደብ
Jus_Imperat Act (Per1 Pl) : ዕንሥደብ
Jus_Imperat Act (Per2 Sg Masc) : ሥደብ
Jus_Imperat Act (Per2 Sg Fem) : ሥደቢ
Jus_Imperat Act (Per2 Pl Masc) : ሥደቡ
Jus_Imperat Act (Per2 Pl Fem) : ሥደቡ
Jus_Imperat Act (Per3 Sg Masc) : ይሥደብ
Jus_Imperat Act (Per3 Sg Fem) : ትሥደብ
Jus_Imperat Act (Per3 Pl Masc) : ይሥደቡ
Jus_Imperat Act (Per3 Pl Fem) : ይሥደቡ
Jus_Imperat Pas(Per1 Sg) : ልሠደብ
Jus_Imperat Pas(Per1 Pl) : ዕንሠደብ
Jus_Imperat Pas(Per2 Sg Masc) : ተሠደብ
Jus_Imperat Pas(Per2 Sg Fem) : ተሠደቢ
Jus_Imperat Pas(Per2 Pl Masc) : ተሠደቡ
Jus_Imperat Pas(Per2 Pl Fem) : ተሠደቡ
Jus_Imperat Pas(Per3 Sg Masc) : ይሰደብ
Jus_Imperat Pas(Per3 Sg Fem) : ትሰደብ
Jus_Imperat Pas(Per3 Pl Masc) : ይሠደቡ
Jus_Imperat Pas(Per3 Pl Fem) : ይሠደቡ
Gerund Act (Per1 Sg) : ሠድቤ
Gerund Act (Per1 Pl) : ሠድበን
Gerund Act (Per2 Sg Masc) : ሠድበክ
Gerund Act (Per2 Sg Fem) : ሠድበሽ
Gerund Act (Per2 Pl Masc) : ሠድባችሁ
Gerund Act (Per2 Pl Fem) : ሠድባችሁ
Gerund Act (Per3 Sg Masc) : ሠድቦ
Gerund Act (Per3 Sg Fem) : ሠድባ
Gerund Act (Per3 Pl Masc) : ሠድበው
Gerund Act (Per3 Pl Fem) : ሠድበው
Gerund Pas(Per1 Sg) : ተሠድቤ
Gerund Pas(Per1 Pl) : ተሠድበን
Gerund Pas(Per2 Sg Masc) : ተሠድበክ
Gerund Pas(Per2 Sg Fem) : ተሠድበሽ
Gerund Pas(Per2 Pl Masc) : ተሠድባችሁ
Gerund Pas(Per2 Pl Fem) : ተሠድባችሁ
Gerund Pas(Per3 Sg Masc) : ተሠድቦ
Gerund Pas(Per3 Sg Fem) : ተሠድባ
Gerund Pas(Per3 Pl Masc) : ተሠድበው
Gerund Pas(Per3 Pl Fem) : ተሠድበው
Infinitive Act (Per1 Sg) : መሥደብ
Infinitive Act (Per1 Pl) : መሥደብ
Infinitive Act (Per2 Sg Masc) : መሥደብ
Infinitive Act (Per2 Sg Fem) : መሥደብ
Infinitive Act (Per2 Pl Masc) : መሥደብ
Infinitive Act (Per2 Pl Fem) : መሥደብ
Infinitive Act (Per3 Sg Masc) : መሥደብ
Infinitive Act (Per3 Sg Fem) : መሥደብ
Infinitive Act (Per3 Pl Masc) : መሥደብ
Infinitive Act (Per3 Pl Fem) : መሥደብ
Infinitive Pas(Per1 Sg) : መሠደብ
Infinitive Pas(Per1 Pl) : መሠደብ
Infinitive Pas(Per2 Sg Masc) : መሠደብ
Infinitive Pas(Per2 Sg Fem) : መሠደብ
Infinitive Pas(Per2 Pl Masc) : መሠደብ
Infinitive Pas(Per2 Pl Fem) : መሠደብ
Infinitive Pas(Per3 Sg Masc) : መሠደብ
Infinitive Pas(Per3 Sg Fem) : መሠደብ
Infinitive Pas(Per3 Pl Masc) : መሠደብ
Infinitive Pas(Per3 Pl Fem) : መሠደብ
Parti Act (Per1 Sg) : ሠዳቢ
Parti Act (Per1 Pl) : ሠዳቢ
Parti Act (Per2 Sg Masc) : ሠዳቢ
Parti Act (Per2 Sg Fem) : ሠዳቢ
Parti Act (Per2 Pl Masc) : ሠዳቢ
Parti Act (Per2 Pl Fem) : ሠዳቢ
Parti Act (Per3 Sg Masc) : ሠዳቢ
Parti Act (Per3 Sg Fem) : ሠዳቢ
Parti Act (Per3 Pl Masc) : ሠዳቢ
Parti Act (Per3 Pl Fem) : ሠዳቢ
Parti Pas(Per1 Sg) : ተሠዳቢ
Parti Pas(Per1 Pl) : ተሠዳቢ
Parti Pas(Per2 Sg Masc) : ተሠዳቢ
Parti Pas(Per2 Sg Fem) : ተሠዳቢ
Parti Pas(Per2 Pl Masc) : ተሠዳቢ
Parti Pas(Per2 Pl Fem) : ተሠዳቢ
Parti Pas(Per3 Sg Masc) : ተሠዳቢ
Parti Pas(Per3 Sg Fem) : ተሠዳቢ
Parti Pas(Per3 Pl Masc) : ተሠዳቢ
Parti Pas(Per3 Pl Fem) : ተሠዳቢ
CompPerf Act (Per1 Sg) : ሠድቢያለሁ
CompPerf Act (Per1 Pl) : ሠድበናል
CompPerf Act (Per2 Sg Masc) : ሠድበካል
CompPerf Act (Per2 Sg Fem) : ሠድበሻል
CompPerf Act (Per2 Pl Masc) : ሠድባችኋል
CompPerf Act (Per2 Pl Fem) : ሠድባችኋል
CompPerf Act (Per3 Sg Masc) : ሠድቧል
CompPerf Act (Per3 Sg Fem) : ሠድባለች
CompPerf Act (Per3 Pl Masc) : ሠድበዋል
CompPerf Act (Per3 Pl Fem) : ሠድበዋል
CompPerf Pas(Per1 Sg) : ተሠድቢያለሁ
CompPerf Pas(Per1 Pl) : ተሠድበናል
CompPerf Pas(Per2 Sg Masc) : ተሠድበካል
CompPerf Pas(Per2 Sg Fem) : ተሠድበሻል
CompPerf Pas(Per2 Pl Masc) : ተሠድባችኋል
CompPerf Pas(Per2 Pl Fem) : ተሠድባችኋል
CompPerf Pas(Per3 Sg Masc) : ተሠድቧል
CompPerf Pas(Per3 Sg Fem) : ተሠድባለች
CompPerf Pas(Per3 Pl Masc) : ተሠድበዋል
CompPerf Pas(Per3 Pl Fem) : ተሠድበዋል
Cont Act (Per1 Sg) : ዕሠድብ
Cont Act (Per1 Pl) : ዕንሠድብ
Cont Act (Per2 Sg Masc) : ትሠድብ
Cont Act (Per2 Sg Fem) : ትሠድብ
Cont Act (Per2 Pl Masc) : ትሠድቡ
Cont Act (Per2 Pl Fem) : ትሠድቡ
Cont Act (Per3 Sg Masc) : ይሠድቡ
Cont Act (Per3 Sg Fem) : ትሠድብ
Cont Act (Per3 Pl Masc) : ይሠድቡ
Cont Act (Per3 Pl Fem) : ይሠድቡ
Cont Pas(Per1 Sg) : ዕሠደብ
Cont Pas(Per1 Pl) : ዕሠደብ
Cont Pas(Per2 Sg Masc) : ትሠደብ
Cont Pas(Per2 Sg Fem) : ትሠደብ
Cont Pas(Per2 Pl Masc) : ትሠደቡ
Cont Pas(Per2 Pl Fem) : ትሠደቡ
Cont Pas(Per3 Sg Masc) : ይሠደብ
Cont Pas(Per3 Sg Fem) : ትሠደብ
Cont Pas(Per3 Pl Masc) : ይሠደቡ
Cont Pas(Per3 Pl Fem) : ይሠደቡ

Appendix I: list of sentences accurately linearized using the Amharic Resource Grammar.

	English	**Amharic**
Sentence	I will come to you	እኔ ወደ እናንተ እመጣለሁ እኔ ወደ እርስዎ እመጣለሁ እኔ ወደ አንተ እመጣለሁ
	I love you	እኔ እናንተን እወድዳለሁ እኔ እርስዎን እወድዳለሁ እኔ አንተን እወድዳለሁ
	these girls are bad	እነዚህ ልጃገረዶች መጥፎዎች ናቸው
	these boys love beer	እነዚህ ልጆች ቢራ ይወድዳሉ
	Those very bad boys drink beer	እነዚያ በጣም መጥፎ ልጆች ቢራ ይጠጣሉ
	I eat bread	እኔ ዳቦ እበላለሁ
	this car is new	ይህ መኪና አዲስ ነው
	these cars are old	እነዚህ መኪናዎች አሮጌዎች ናቸው
	she breaks the window	እርሷ መስኮቱን ትሰብራለች
	those windows are clean	እነዚያ መስኮቶች ንጹሆች ናቸው

DECLARATION

I undersigned declare that this thesis is my original work and has not been presented for a degree in any other university, and that all materials used for the thesis have been duly acknowledged

_____

Nebyou Woubshet Worede

June, 2014

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

_____

Dr. Solomon Teferra

June , 2014