Grammar Engineering for the Ekegusii Language in Grammatical Framework

Benson Kituku, Wanjiku Nganga, and Lawrence Muchemi

Abstract — **The knowledge-driven economy uses technology, thereby increasing the demand for language tools and resources to acquire and distribute the knowledge. Such tools and resources are scarce for the under resourced, spoken Bantu languages. This paper develops a computational grammar for the Ekegusii language in the Grammatical Framework (GF) to bridge the gap. The grammar development uses a bottom-up and modular-driven approach. A machine translation experiment was set up to evaluate the grammar resulting in BLEU and PER of 55.95% and 19.49%, respectively. This work contributes by providing computational grammar for an under-resourced language, thus providing a platform for analysis and synthesis, plus a machine translation within the GF ecosystem.**

Index Terms — **Ekegusii, Grammar Engineering, grammar, Grammatical Framework, Under resourced language.**

I. Introduction

The technology knowledge-driven economies demand natural language processing (NLP) tools and resources to acquire and distribute knowledge and information [1]. However, few NLP tools and resources exist for under resourced languages and more so for the spoken Bantu languages; thus, language technology digital divide. One way to solve the challenge is Grammar engineering (GE) for under resourced languages. GE is creating computational grammar based on formal grammar that machines can parse or generate [2]. This paper performs grammar engineering for the Ekegusii language, an under-resourced language resulting in computational grammar using the grammatical framework (GF). The work is a significant milestone towards creating a standard Basic Language Resource Kit (BLARK) [3] for such a language.

Ekegusii is an agglutinative and tonal Bantu language. Guthrie [4] classify this great lake Bantu language [5] as E42. There are two main dialects of Ekegusii, mainly Maate and Rogoro dialects [6]. This research has used the Rogoro dialect by virtual of being the standard dialect.

GF is a grammar toolkit, a logic framework of syntaxes and a grammar formalism grounded on categorical formalism [7]-[9]. It has a single abstract syntax that defines a set of categories (Cat) of trees, a set of functions (Fun) to implement those trees plus their type and start category [10]. Furthermore, it has many parallel concrete syntaxes, one for each language grammar. These syntaxes define linearization of both the categories (lincat) and the function (lin) stated in the abstract syntax as exemplified using category Noun (N) with string "house" below [7].

Abstract syntax	**Concrete syntax**
Cat: N	lincat N = Str
Fun House: N	lin House = " house"

The parallel concrete syntaxes are parallel multiple context-free grammars (PMCFG) [11] and reside in the GF resource grammar library (RGL), where the syntactic and morphological properties of a specific language are captured and form the multilingual grammars ecosystem [12]. The RGL is subdivided into three primary modules: lexical, morphology and syntax. Morphology is implemented via paradigms (low and smart), which are functions that take lexeme word form(s) and generates the complete word forms (inflection table). Detrez et al. [13] define a smart paradigm *as a Meta paradigm that inspects the given base form of a lexeme and tries to infer which low paradigm applies*. Parsing provides a means of transforming language-specific strings to abstract trees, while linearization is a composition of homomorphic mapping from common abstract tree structure to specific language concrete syntax [8]. Machine translation would then be achieved by first parsing the source language's string to abstract trees then linearizing the abstract trees to strings in the target language.

Features in GF formalism are provided via parameters and are defined using the keyword *param* and mostly used in table types. For example, the noun in Bantu languages has a parameter number with values: singular and plural; therefore, the definition would look like:

```
param
Number = Singular | Plural
```

A category may have more than one parameter. In such a case, a data structure record is used to gather them. For example, the category noun in Ekegusii has additional parameter gender apart from the number; therefore, it is defined as:

```
N = {s: Number => Str; g: gender};
```

The above is a table from number to strings and has an inherent feature of gender (functions over parameters) [14]. GF distinguishes the function *fun* used in abstract syntax and the function operation *oper* used to implement inflection paradigms. Operation is used to implement the regular pattern in grammars to avoid redundancy of repetition. The keyword *oper* is usually of the form:

Submitted on February 25, 2021.
Published on March 18, 2021.
Benson Kituku, Dedan Kimathi University of Technology, Kenya.
(e-mail: benson.kituku@dkut.ac.ke)
Wanjiku Nganga, University of Nairobi, Kenya.
(e-mail: wanjiku.nganga@uonbi.ac.ke)
Lawrence Muchemi, University of Nairobi, Kenya.
(e-mail: lmuchemi@uonbi.ac.ke).

oper function name: function type = function body

One name can be used for different paradigms in the same category through operation overload

Some Bantu language grammars have been developed in GF, mainly: Kikamba [15], Swahili [16] and Runyankore and Rukiga [17]. Furthermore, Ekegusii has other language resources such as an Interlingua rule-based machine translation between Swahili and Ekegusii [18] and morphology analyzer [19], An online dictionary, little parallel corpus (some bible segments) and some monolingual corpus [20]. The above survey demonstrates that little work has been done to develop NLP tools and resources for this language; hence, this computational grammar will be a significant effort.

II. Ekegusii Descriptive Grammar

Four descriptive grammar sources [6], [21]-[23] were used in computational grammar development. Here, we shall explain the descriptive grammar, starting with morphology followed by syntax.

A. Morphology

The morphology uses the nominal[1] class system [24] based on affixes to a noun root [22]. Arguments have been forwarded whether the nominal class system should be referred to as gender or noun class see [15]. In this paper, we adopt the view that a pair of noun classes (singular and plural) are to be regarded as gender, as shown in Table 1.

TABLE I: EKEGUSII GENDERS(NOUN CLASSES)

Genders(classes)	Class number
omo_aba	1,2
omo_eme	3,4
eri_ama	5,6
ege_ebi	7,8
e_ci	9,10
oro_ci	11,10
aka_ebi	12,8
obo_ama	14,6
oko_ama	15,6
ama_ama	6,6
aa	16

The regular noun's structure consists of a prefix that is a noun class number(a subset of gender) and the noun root. The structure is exemplified by Example 1, where c means class and the number represent the class number as stated in Table 1.

EXAMPLE 1: NOUN STRUCTURE

Singular	Plural
Eke-rogo	Ebi-rogo
c7-stem (gender ege_ebi)	c8-stem (gender ege_ebi)
Chair	Chairs

An adjective is a noun modifier inflect for number and class (a subset of gender). The regular adjective structure has an obligatory prefix (concord), which agrees with the nominal class [6], [21]-[23]. Example 2 shows a phrase using gender omo_eme (class 3 and 4). The bold shows the prefix of the gender used. The root of the adjective is represented by adjroot.

EXAMPLE 2: REGULAR ADJECTIVE STRUCTURE

Singular	Plural
Omo -te omo -nke	Eme-te eme-nke
Omo eme –gender nke -Adjroot	Omo **eme** –gender nke –Adjroot
Small tree	Small trees

Verbs have a complex morphology with much prefixing and suffixing plus infixing for extensional morphology. Its declension involves several morphemes (several prefixes, root, extensional suffix, and final vowels representing mood) plus some grammar features such as person, number, gender, tense, polarity, etc. The morphemes of verbs embody all the constituents needed to make a sentence. Hence a verb can act in place of a sentence. Table II [6], [21]-[23] summarizes all the prefixes, suffixes, roots and extensions needed to form verbs in this language. The subject marker represents positive polarity, while the negation morpheme is negative polarity. Both have grammar features of gender, number, and person that form the agreement parameter. It is essential to note that the following fields are usually not obligatory: object marker, infinitive, and extension. The focus morpheme cannot exist with negation [23]. The following notations are used: *Fs* for focus, *Neg* for negation, *Agr* for the subject marker, *root* for the root, *Tns* for tense, *Asp* for aspect, *Fw* for the final vowel and *Aux* for the auxiliary verb.

TABLE II: VERB STRUCTURE

Structure	Morpheme	Ekegusii
Prefixes	Focus	"n"
	Negation	as per gender, person and number
	Subject marker	as per gender, number and person
	Tense/Aspect	As per tense
	Object marker	as per gender and number
	Infinitive	"ko"
Root		Root
Extension Suffix	Applicative	"er"
	Causative	"I"
	Passive	" u"
	Reversive	"or"
	Reciprocal	" an"
	Stative	"ek"
Final vowel		"a/e/i"

The Ekegusii tenses are marked by a tense morpheme or no morpheme at all. Three points are needed to mark different tenses, as argued by Reichenbach [25]. These points are the speech point, the reference point and the event point in relation to time, while time is based on speech point [26]. The coincidence of the three points results in the present tense. When the speech point is after the other two points, then past tense occurs. Future tense occurs when the speech point is before other points. Finally, when the reference time proceeds to event time, the resultant is perfect tense.

The aspect gives a view of the verb's action, such as beginning, continuing, or ended [26]. Most of the time, tense and aspect are combined. Several tenses exist [6], [21]-[23]. This discussion focuses on the present, future and past tenses.

The Future tense is marked by the suffix "e" [27] though Ongorora [23] argues that the morpheme "e" does not necessarily represent tense. The Future tense is exemplified by Example 3.

[1] https://glossary.sil.org/term/noun-class

EXAMPLE 3: FUTURE TENSE

a-	gocha	go- rar -e
Agr Aux infix morpheme		*Root Tns*
He will sleep		

There are four past tenses in Ekegusii. The morpheme "a" marks the immediate and hesternal past tense, while morpheme "ete" marks the distant and hodiernal past tense [23]. Immediate past tense is today past marked by high tone while hesternal is a recent past marked by low tone of the morpheme. hodiernal is yesterday past while distant is remote past Example 4 exemplifies these tenses.

EXAMPLE 4: PAST TENSE

Ba-	a-	minyok- a
Agr Tns		*Root Fw*
Ba-	a-	minyok- ete
Agr Tns		*Root Tns & Fw*
They ran		

Though Ekegusii exhibit dichotomy regarding tenses (past versus no past) [23], they have present tense, exemplified by habitual tense or progressive tense. Example 5 demonstrates the present tense.

EXAMPLE 5: PRESENT TENSE

A-	ko-	raager- a
Agr tns		*Root* Fw
He is eating /He eats		

Possessive pronouns show ownership by modifying a noun and its structure consists of prefix morpheme, which agrees with the noun gender, while the root has the grammar features of number and person. All genders have person P3 except gender omo_aba that has additional P1 and P2. The personal noun is dropped (pro-drop) when it appears as the subject of a sentence, since is encoded in the verb subject marker [22].

Demonstratives are noun modifiers that show how far object(s) is/are from the speaker. Indo–European languages demonstrate strings for near and distant as opposed to Bantu languages that have an extra string for the aforementioned demonstrative [22]. The demonstrative string has variable features of gender and number.

Adverbs, interjections, and prepositions have a string that is independent of gender. However, the preposition "of" string agrees with the gender and number. Numeral, too, modifies the noun. Generally, the cardinal numerals one to five have a prefix based on gender agreement beside the root. There are no numerals six to nine but a repetition similar to that of one to five, based on the gender of nouns that have been modified, as exemplified in Example 6.

EXAMPLE 6: CARDINAL NUMERAL

emete etano ebere	abarwaria emerongo etano babere
trees five two	Doctors tens fifty twenty
seven trees	Seventy doctors

Therefore, it is like adding two numbers between one and five to get a number between six and nine. There is a disjunctive string for ordinal numeral before the cardinal number except for numbers 1 to 3, which have unique ordinal writing and are based on the gender agreement. The ordinal and cardinal are available in digit and numeral form.

B. Syntax

The dominant topology for Ekegusii is subject-verb-object (SVO) [22], [28], whereby the subject is a noun phrase, followed by a verb phrase. The verb phrase is made of a verb and object complement that can be a verb phrase, noun phrase or both. The object's presence is influenced by the verb valence (univalent, divalent, and trivalent). For example, for the univalent verb, the topology becomes SV because the one place verb does not require arguments. The syntactic agreement is via concord agreement within the lexical items mainly influenced by the gender [22].

A noun phrase (NP hereafter) is made of a noun and its modifiers that include adjectives (Adj), Determiners (Det), both possessives (Poss) and demonstratives (Dem) and numbers (Num). Definition 1 how the noun modifies follow each other to form the NP's structure [22], [29].

Definition 1 NP structure

[dem] [Noun] [Det <Pss.> <Dem>] [Num] [Adj]

The structure represents a complex NP, while a simple NP can only be formed either by a noun or a pronoun. It is also possible to form a complex noun using post-modifiers to the noun phrase - mainly interrogative and past participle of a verb. The verb phrase structure is the same as a verb and carries all parameters that are integral to verbs.

III. Implementing Ekegusii Grammar in GF

The computational grammar development employed an experimental research design. The grammar development adopted the GF morphology-driven strategy and modular-driven development, a bottom-up method. It involves first defining the lexicon, smart paradigms based on the regular expression and their respective linearization categories before working on the syntax [8]. The evolutionary prototype model [30] was used because each function developed had to be iteratively tested to ensure its work before moving to the next function. GF provides text output in the command prompt. However, to visualize the parse trees from production rules or paradigms for the grammar, the Graph viz[2] tool was used. It takes simple texts as input and converts them into diagrams.

A. Morphology

Table III represents the coding of genders in Table I. The coding is of GX, where G stands for gender and X is a number starting from one. Each gender combined two nominal classes based on parameter number (singular and plural) and separated by an underscore. The genders are coded in the resource grammar using the parameter Cgender as per Definition II.

[2] https://graphviz.org/

TABLE III: GF CODING OF GENDERS

GF coding	Genders
G1	omo_aba
G2	omo_eme
G3	e_ci
G4	eri_ama
G5	ege_ebi
G6	oro_ci
G7	aka_ebi
G8	obo_ama
G9	oko_ama
G10	aa
G11	ama_ama

Definition II Gender coding in GF

```
param
Cgender = G1|G2|G3|G4|G5|G6|G7|G8|G9|G10| G11;
```

These genders influence concordial agreements with part of speech tags; hence agreement is implemented using parameter *Agr* and its composition consist of gender, number and person as per Definition III. To minimize over generation during inflection, especially for verbs, optimization was done so that Person one (P1) and two (P2) are only applicable to gender one (G1) because animate (human) belongs here. The function *toAgr* translates each person level to the right agreement as exemplified in definition III.

Definition III. Agreement definition

```
param
Agr = AgP1 Number | AgP2 Number | AgP3 Cgender Number ;
oper
toAgr : Cgender -> Number -> Person -> Agr = \g, n, p ->
 case p of {
 P1 => AgP1 n ;
 P2 => AgP2 n ;
 P3 => AgP3 g n } ;
```

Generally, lexeme definition for linearization of each category followed a similar structure and involved the following, as exemplified by Example 6.

- Definition of the linearization category;
- The low-level paradigm;
- The lexeme for the category;
- Parameter for the category (some had others did not have).

EXAMPLE 6: LEXICON DEFINITION

Ekegusii Languages	Gloss
woman_N = regN "omosubati" omo_aba;	Woman
small_A = regA "nke" ;	Small
play_V=regV"chiesa";	Play
we_Pron =mkPron "intwe" "ito" G1 Pl P1 ;	We
very_AdA = mkAdA "mono" ;	Very

Using the example drawn from Example 6. regN is the paradigm for noun where a woman belongs, "omosubati" is the lexeme for a woman in Ekegusii language, while oma_aba is the parameter gender in which the noun belongs

woman_N = regN "omosubati" omo_aba;

a) Noun

The noun inflects for parameter number that has values singular(sg) and plural (pl) and gender. Therefore, its linearization is modeled as a number to string as below.

N = {s : Number => Str ; g : Cgender} ;

Four paradigms(regular expression) are used to model noun inflection and Table IV explains these paradigms for nouns.

TABLE IV: NOUN PARADIGMS

Function	Type	Explanation
mkN	Str ->Cgender -> N	Function *regN* takes in a string and gender and returns regular words forms
mkN	(man,men : N)-> Cgender -> N	Function *compoundN* takes in two strings of nouns and gender and generate compound noun forms
mkN	V -> N	Function *verb2snoun* takes in a verb and generates a noun in the gender G1.
mkN	(man,men : N)-> Cgender -> N	Function *iregN* takes in two strings of nouns and gender, then assign one as singular and the other plural

An extra paradigm *mkNoun* (make noun) is used to assign all forms of inflections to the right parameter number. We illustrate noun paradigms using regN in Fig. 1. The function *PrefixPlNom* represents the gender morpheme.

```
regN : Str ->Cgender -> Noun = \w, g -> let wpl = case g of {
 G1 =>case w of {
                  "omwo" + _ => "aba" + Predef.drop 3 w ;
                  "omw" + _ => "ab" + Predef.drop 3 w ;
                    _ => PrefixPlNom G1 + Predef.drop 3 w};
 G2 =>case w of {
                  "omw" + _ => "emi" + Predef.drop 3 w ;
                    _ => PrefixPlNom G2 + Predef.drop 3 w};
 G3 => "chi" + Predef.drop 1 w;
 G4 => case w of { "ri" + _ => "ama" + Predef.drop 2 w ;
                    _ => PrefixPlNom G4 + Predef.drop 1 w};
 G10=> [];
 G11=> w;
 _ => PrefixPlNom g + Predef.drop 3 w};
 in mkNoun w wpl g ;
```

Fig. 1. regN paradigm.

The noun morphological inflection table consists of a maximum of two-word forms for each number. An example of the noun "tree" is highlighted below

```
lang> linearise -table tree_N
s Sg : omote ---gloss tree
s Pl : emete --- gloss trees
```

b) Adjective

Adjective inflects for number and gender. Therefore, the parameter *AForm* was used to represent the above two-variable features(parameters) of concordial agreements. The parameter is defined below with values *AAdj and Advv* for positive and adverbs adjectives.

```
param
AForm = AAdj Cgender Number | Advv;
```

Three paradigms were used to model adjective inflection. The paradigm *regA t*akes a string and generates an inflection table. Colour adjectives except red, white and black are generated uniquely using the *cregA* paradigm. Its structure consisted of a prefix specific for each number and gender, the same as the prefixes of pronouns; hence the same operation used in pronouns are used here. The prefix is followed by a string *"color of",* then the color lexeme as illustrated below.

The irregA paradigms deal with irregular adjectives and just assign the listed strings in the lexicon definition.

```
cregA : Str-> {s : AForm => Str} = \seo -> {
 s = table {
 AAdj g Sg => ProunSgprefix g ++ "eragi ya" ++ seo;
 AAdj g Pl=> ProunPlprefix g ++ "eragi ya" ++ seo;
 Advv=> []} } ;
```

c) Verb

The Grammatical Framework resource grammar library, by default, provides positive and negative polarities, past, present, future, and conditional tenses, as well as simultaneous and anteriority [8]. The positive polarity is implemented using the subject marker morpheme, while the negation morpheme implements negative polarity. The two morphemes require extra grammar features to allow agreement, mainly gender, number, and person (first, second and third). The tense or sometimes aspect morpheme implemented both anterior and tense. Other morphemes, as presented in Table II, were also used to implement the verbs. The record type implementation for the verb linearization is shown below.

```
Verb = { s :VForm => Str;
         progV:Str;
         imp : Polarity => ImpForm => Str;
         s1 : Polarity => Tense => Anteriority => Agr=> Str };
```

The string *s* represent various verb forms, namely: infinitive, extensional or derivative morphology form and general form with a final vowel "a", anterior, future and negation form. The second string *progV* is for the progressive verb and *imp* for an imperative verb. The imperative verb is implemented in the *sentence modul*e of RGL and inflects for polarity and parameter *impForm* (number and Boolean with true being a polite request while false being a command).

To model derivational/ Extensional morphology, the parameter *VExte* was used to model all forms shown in Table II and its GF definition is shown below

```
Param
 VExte = EPassive | EApplicative |EReciprocal | ECausative |EStative;
```

The paradigm *regV* generated the possible five forms of verbs while paradigm *mkVerb* used the output of regV or irregV (for irregular verbs where all forms were just listed in the lexicon definition) to generate the inflection table of 403-word forms. The forms are many because a string must be generated for each concord agreement, number, tense, polarity and person.

d) Numerals

Numerals[3] are either cardinal or ordinal. Cardinal describes quantity while ordinal shows order and represented in digits or words, for example, 12 and twelve respectively, both formats are supported in GF. The GF numeral implementation is based on Hammarström and Ranta's [31] work. The numeral linearization type implementation is exemplified below and numeral parameters.

Numeral definition and parameter

```
lincat
Numeral = {s : CardOrd => Cgender => Str ; n : Number} ;
Digits = {s : CardOrd => Cgender => Str ;n : Number    } ;

Param
                CardOrd= NCard | Nord;
                DForm = unit | teen | ten |hund ;
```

The numeral and digits categories have gender and *CardOrd* as variable parameters and number as an inherent parameter. The values of parameter *CardOrd* are cardinal *(Ncard)*and ordinal*(Nord)* numerals. The numeral one is the only one that has the value of a number as singular; all others are plural. The values for parameter DForm were *unit, teen, ten* and *hund.* The *unit* is for numerals between 0 to 9. The *teen* is between 11 to 19, while *ten* is for multiples of ten and *hund* for multiples of hundreds. GF[4] implements numbers ranging from 0- 999,999.

There was gender agreement (concord) for the cardinal numeral one to five in building the numeral. For numerals, six to eight and their multiples are constructed by recursion between one and five. For example, eighty would be constructed as (fifty thirty), as exemplified by Fig. 2.

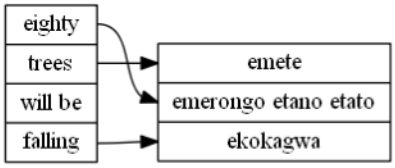

Fig. 2. Word alignment.

Cardinal numerals between 1-9 and their multiples of ones, tens, teen, and hundreds had a specific paradigm to model them. The ordinal numeral was modeled by adding a disjunctive prefix of singular "of" in the specific language, while digits were modeled using a similar function for all languages. The function *IDig,* which took argument digit, returned digits 0 to 9, while function *IIDig* which took argument digit, followed by digits, returned numeral with at least two digits. The operation *mk3Dig* created the cardinal digits and ordinal digits by attaching the cardinal digits' disjunctive prefixes. Fig. 3 exemplifies cardinal numeral in digits and words for gender G1(omo_aba) in Ekegusii. For gloss -"four hundred and eighty-two".

[3] https://www.englisch-hilfen.de/en/grammar/zahlen.htm

[4]https://www.grammaticalframework.org/lib/doc/gfdoc/Numeral.html

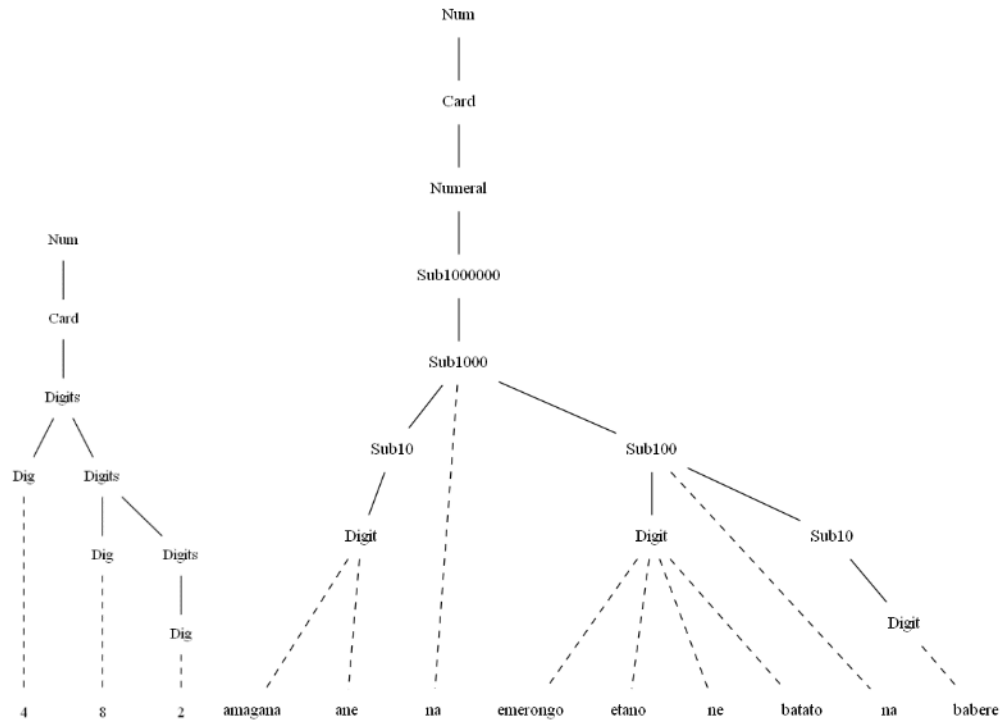

Fig. 3. Numeral examples.

e) Pronoun

Personal and possessive pronouns are the two forms of pronouns implemented in GF. The personal pronoun acts as a noun phrase, thus requires an agreement in terms of gender, number, and person. The possessive pronoun in GF is implemented as a quantifier. Thus, a determiner hence inflects for gender and number. The parameter *PronForm* with values *Pers* for the personal pronoun and *Poss* for possessive pronoun are used to model in the paradigm *mkPron* that takes five arguments inputs (two strings- one for personal and a stem for the possessive, gender, number, and person in that order) as exemplified by the lexicon of the pronoun "he" below. The output is two sets of strings, namely: the personal pronouns that act as a noun phrase and the inflection table for the possessive pronoun based on gender and number.

```
he_Pron = mkPron "ere" "je" G1 Sg P3 ;
```

```
mkPron: (i, mine : Str) -> Cgender -> Number -> Person ->
 {s: PronForm => Str ; a : Agr} = \i,mine, g,n,p ->
 { s = table {
            Pers => i;
            Poss n g => case <n,g> of {
                        <Sg ,_> => ProunSgprefix g + mine ;
                        <Pl, _> => ProunPlprefix g + mine}} ;
 a = toAgr g n p } ;
```

Pronoun paradigm

f) Preposition

Some prepositions inflect for gender and number, for example, the preposition "of" which was established through elicitation. Most of the other prepositions are just strings like in most of the other languages. The mkPrep paradigm was used to model preposition as shown below, where string *s* was used to implement the preposition, while *s1* was the string for infusion. Finally, the Boolean operator determined whether a specific preposition can be infused or not (true value meaning fused and vice versa).

```
mkPrep = overload {
 mkPrep : Str ->Bool-> Prep = \str,bool ->
 lin Prep {s = \\n,g => str ; s1= infusedstring;isFused = bool } ;
 mkPrep : (Number => Cgender => Str) ->Bool-> Prep = \t,bool ->
 lin Prep {s = t ; s1= infusedstring; isFused = bool} ;};
```

g) Other categories

Quantifier was defined as strings that inflect for gender and number. Determiners show an indefinite number of people or objects [32]. They include but are not limited to every, much, all, etc. and inflect for gender plus an inherent number. Moreover, some come before the noun they modify while others come after the noun. To show the determiner's position in relation to the noun it modifies, *isPre* a Boolean parameter is used. T*rue* indicates it comes before and *false* shows it comes. Adverbs do not inflect; hence are mere strings in their definition. However, on adjectives, there were adverbs formed out of adjectives. Therefore, to accommodate them at the syntax phase, since adjective inflects for gender and number, the adverb was configured to inflect for agreement (gender, number and person). Person three(P3) was used as a constant.

B. Syntax

The syntax is implemented using the dominant SVO topology and the parameters were exchanged among the categories in order to ensure syntactic agreement (concord agreements). The V topology is also implemented primarily where personal pronouns are used as the subject(S), thus pro-drop of the subject since it's represented in the verb using the subject marker. Finally, SV is implemented where the verb does not have a compliment.

The CN production rules were implemented in different ways. First, using the noun of either valency one or two, then

NP and a relational noun or usage of "of" together with a noun phrase. Finally, a CN was modified by an adjective, relative clause, adverb, sentence and noun phrase, respectively. Overall, nine production rules for CN were implemented.

Syntactically, determiners phrase can be formed from numerals and quantifiers, with the latter been the central and the former optional. Here the focus is constructing a determiner from more than one category (speech tags). Two production rules were modeled. In the first one, the determiner is formed from quantifier and numeral, while in the second rule, there is the addition of ordinal numeral. The determiner is a post modifier of a noun hence the reason the Boolean *isPre* is true. Figure 4 shows an example of rule one for the two-grammar using gloss" these seven".

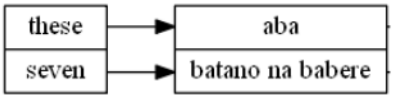

Fig. 4. Determiner example.

Adjective phrases (AP) are formed in several ways. One rule implemented simple positive degree AP while the Comparative degree AP was implemented using positive adjective plus noun phrase and the "than" string. AP rules were also made from a relational adjective. AP rules were formed by AP been modified by NP, reflexive pronoun, adverbs, sentence complement and noun phrases. Figure 5 exemplified AP formed by a comparative adjective with NP as its modifier, for the gloss better than some students.

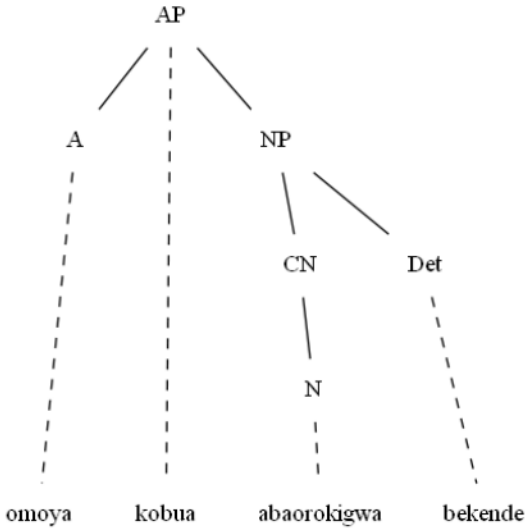

Fig. 5. Example of AP parse tree.

The NP was implemented from the common noun, proper names, determiners, pronouns, and recursion of NP with adverbs, pre-determiners, and determiners. NP. The implementation used two parameters: case and agreement (concord), which are needed when combining the NP with a verb phrase. The case values were *nom* for the nominative case, while *NPoss* was introduced to cater to noun phrases from personal and possessive pronouns. Since personal pronouns are NP and can be inferred from the verb's subject marker morpheme, the field *isPron* was used to store the information on whether the current NP is a pronoun or not to enable future pro-drop when needed. Complex are formed using a pre- or post-modifier of NP. The pre-modifiers are pre-determiner and determiner, while post-modifiers are past participle verbs, relative clauses, and adverbs. In total, ten productions of NP were implemented. Figure 6 shows word alignment for NP "all my three black eyes" in English. The NP. consists of a predetermined, possessive pronoun, cardinal numeral, adjective and a noun.

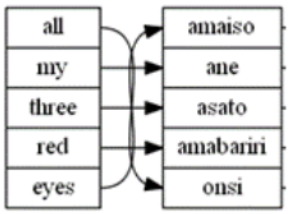

Fig. 6. Noun phrase word alignment.

The Verb phrase implementation mirrored the structure of verb implementation. Therefore, its linearization is the same as that of the verb and the VP morphology paradigm *regVP* uses similar strings as those of the verb with only two addition, *compl* for the verb's object and *inf* for infinitive verbs. The subcategorization of verbs was taken care of through *compl* (one place, two-place, and three-place verb). The VP complements that are listed below. In total, 21 syntax rules were implemented:

- Use of the verb or the verb phrase.
- Use of verb *to be* and its complements (auxiliary verbs).
- Use of adverbs complements.
- Verb passivation of the verb.
- Reflexive complement.

Fig. 7 shows an example of VP for the gloss "I read the best book".

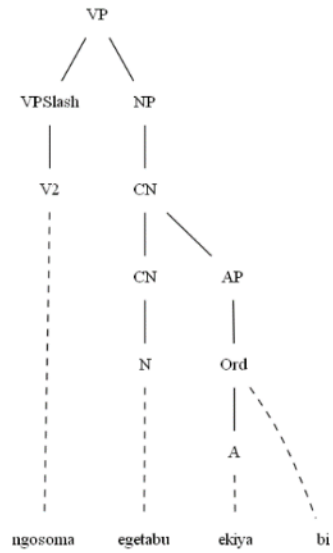

Fig. 7. Egekusii VP parse tree.

The next category to model was clauses: declarative, question and relative clauses. All clauses have undetermined polarity, tense and anteriority, which is fixed at the sentence level. The question clause uses the parameter *QForm* with values *QDir* and *QIndi*r for direct and indirect questions.

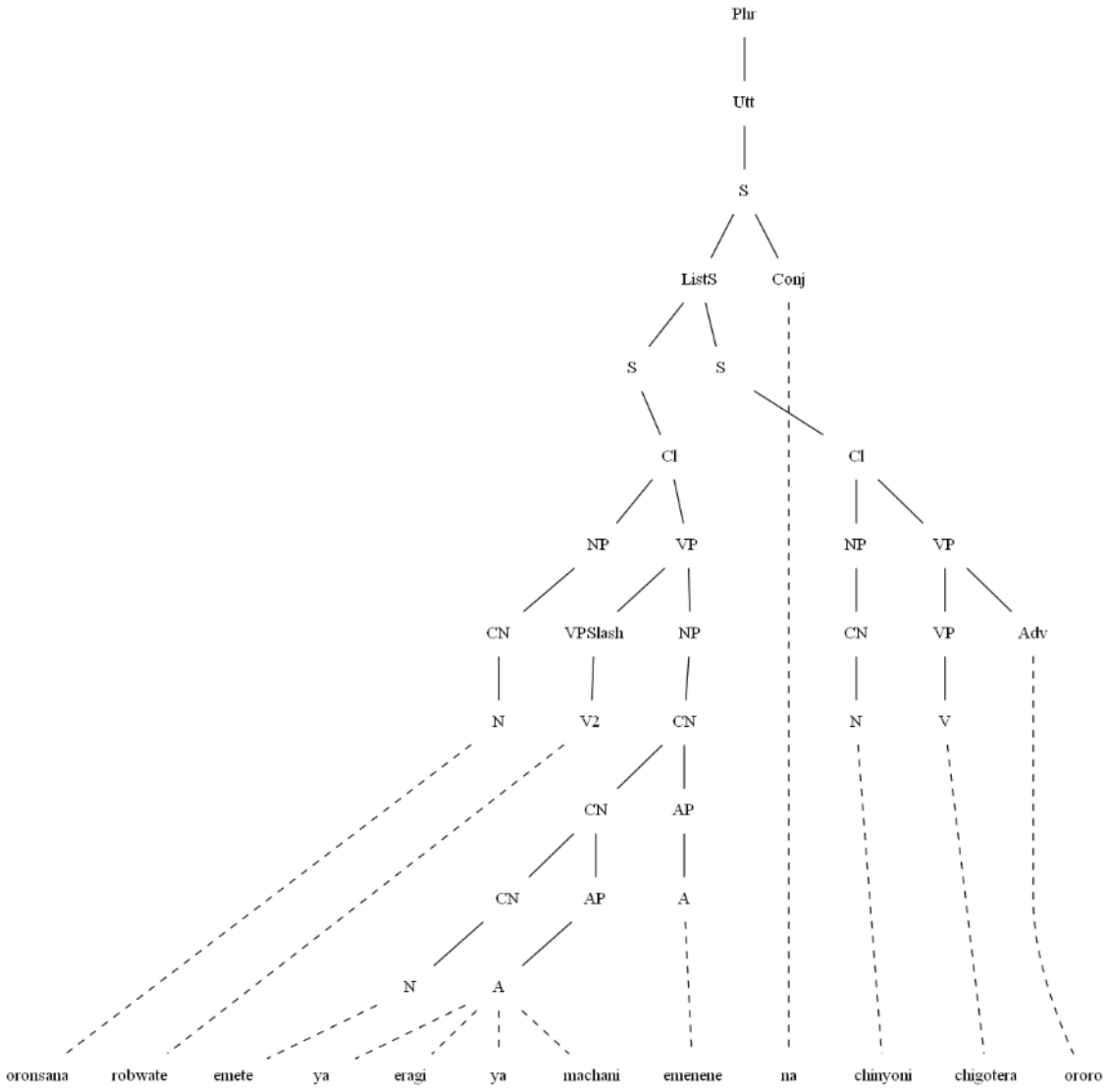

Fig. 8. Clause/Sentence parse tree.

The direct question clause is formed by changing the declarative clause's tone to high or using a question mark. Declarative clauses are formed using the SVO topology where the S is a noun phrase, while V is the s field of the Verb phrase and O is the compl field of the Verb Phrase. Figure 8 is an example of a clause for "oronsana robwate emete ya eragi ya machani emenene na chinyoni chigotera ororo" with gloss "The forest has big green trees and birds sing there". The parse tree shows the combination of NP and VP makes both clauses. The VP in the first clause consists of the two-place verb with an NP as an object, while the second clause is made of a one-place verb and adverb.

The two ways used to implement question clauses (*QCl)* were through the yes or no answer questions or through Interrogative. These interrogatives are: interrogative Pronouns (*IP*), interrogative Adverbs (*IAdv*), Interrogative Quantifiers (*IDet*), copula interrogative complement (IComp) and their modifiers. The relative clause (RCl) was formed in three ways. The basic way is to use a clause. The second and third involve verb phrases and a sentence that lacks a noun phrase been modified by a relative pronoun (RP). In total 19 production rules were implemented.

The sentences were primarily implemented by fixing the tense, anteriority, polarity at the question, declarative and relative clauses. Other ways include the use of embedded sentences such as question sentences and infinitive verb phrase. An adverb can modify a sentence with a comma or not. Finally, sentences were constructed using the subjunctive, relative clause and imperative verbs. The utterance was not implemented from sentences, questions and imperatives but also using one word, especially where it is an answer to a question in the following categories: noun phrases, interrogative adverb, interrogative pronouns, common nouns, numerals, verb phrases, adjective phrase, adverbs, and interjections. Fig. 9 exemplifies one-word utterance.

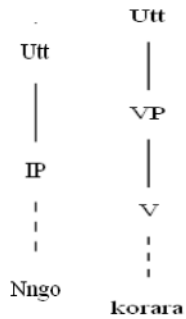

Fig. 9. Utterance Examples.

The phrase is the start category and three productions were implemented by prefixing and suffixing utterance with phrasal category and Noun phrase as a vocative (Voc), respectively [8].

Finally, production rules for coordination were implemented for the following categories: sentences, adverbs, interrogative adverbs, noun phrases, adjectives, relative sentences, common nouns and determiner phrases.

In total, at the syntax phase, the Ekegusii computational grammar had 163 production rules.

IV. Results and Discussion

GF is a multilingual ecosystem; therefore, a machine translation from English to Ekegusii was set up to evaluate the grammar precision. Consequently, the Bilingual Evaluation Understudy (BLEU), Word Error Rate (WER) and Position Independent Error Rate (PER) metrics, which are commonly used metrics for evaluating machine translation [33], were used. BLEU (ranges from 0 to 1 or expressed as a percentage) demonstrated a good correlation of machine translation to human judgment [34]. PER and WER based on Levenshtein distance [35] are excellent metrics to investigate Ekegusii errors since this language has a lot of nasal insertion, deletion and substitution, especially the joining of morphemes at the word level. Besides, all-inclusive error analysis taxonomy [36] by comparing hierarchical [33], [37] and linguistic [36] taxonomies was used to analyze the errors resulting from the machine translation manually.

A 100-sentence test suite (in the English language) developed specifically for Bantu languages [15] was used to perform the machine translation. An Ekegusii expert translated the test suite into the Ekegusii language to act as the gold standard. The human translation was subjected again to another expert to confirm the translation correctness. The 100 English sentences were parsed (strings to abstract trees) and linearized (abstract trees to strings) to Ekegusii using the developed Ekegusii grammar. This machine translation output formed the candidate or target language translation. The machine translations, Human translation (gold standards) and the source language (English) were in text files. The three sets were compared using the online Tilde software to extract the BLEU score where the first two were of the same language. The WER and PER metrics were extracted using Perl scripts by comparing the candidate and human translation.

Table V presents the extracted English to Ekegusii machine translation metrics score.

TABLE V: TRANSLATION METRICS

Metrics		Percentage
BLEU	1-gram	80.50
	2-gram	69.25
	3-gram	61.86
	4-gram	55.95
PER		19.49
WER		23.90

The BLEU score measures the similarity index by comparing the same phrase length(n-gram) between the target(candidate) and reference (gold standard) sentences. Though, 1,2,3 and 4 grams (phrase length) are scored. To addressing fluency of the translation since GF is known to over generate, the longer n-gram (4-gram) is used. The Ekegusii 55.95% BLEU score is encouraging for a language with much morphophonological transformation. The metrics PER and WER were used to investigate errors because the former does not penalize position while the latter does and this had a huge effect on accuracy, especially where two consecutive adjectives were used in a sentence as illustrated by Fig. 9, where a sentence has correct translation. However, due to two consecutive adjectives: red(*chimbariri*) and small (*chinke*) are interchanged in the target(candidate) translation, it results in 50% WER and 0% PER. Besides, the implication is high on the BLEU score since it scores partly 22.59%. Table 5.5 shows 19.49% and 23.90% for PER and WER scores. An in-depth analysis of the errors by manual annotation using the comparative taxonomy is shown in Fig. 10.

Source	-	-	small red seeds smell
Human	100.00	1.00	chintetere chinke chimbariri chigotiokerera
Machine	22.59	1.00	chintetere chimbariri chinke chigotiokerera

Fig. 9. Position interchanged error.

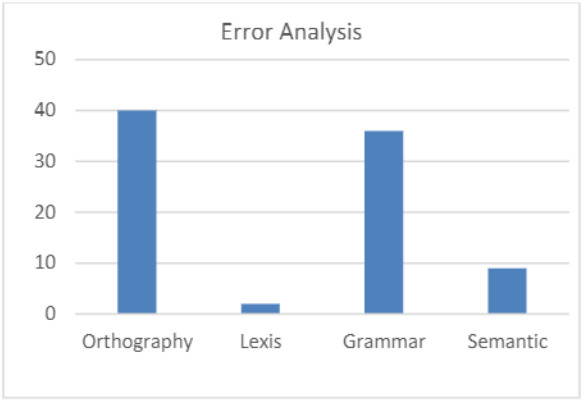

Fig. 10. Manual error analysis.

Orthography errors involved many misspelled words requiring addition, deletion, or substitution of one or more letters at the word level. These errors are mainly due to phonological issues resulting from nasal deletion and insertion. Most of the phonological rules were not available in the descriptive grammars and thus not captured in the morphology since they were realized at the evaluation stage. Fig. 11 shows when two consonants meet (r and g) the g is deleted in Ekegusii due to nasal issues. Therefore, these were not inflectional errors but phonological errors.

Fig. 11. Ekegusii orthography error.

Some words were added or subtracted in the human reference or target reference to ensure the translation is semantic correctly. Such errors are at the lexis level. Grammar errors were the highest and mostly related to the verb phrase. There was mis-selection in the verb tense. For example, Ekegusii has variants of past tense (immediately,

near, far, remote). When all the variants are implemented in GF due to the large verb inflection table, the compiler took a long time (more than 8 hours) to process. In fact, in most of the cases, the process was killed. Therefore, for testing purposes, we only used one tense in each category and this led to the scenario shown in Fig. 11 where the human translation is in the remote past tense while the machine translation is in the far past tense. Therefore, coping with this limitation of time complexity in GF led to several errors.

Source	-	-	the twenty very bad men drank beer
Human	100.00	1.00	abasacha emerongo ebere ababe mono banywete amarwa
Machine	64.35	1.00	abasacha emerongo ebere ababe mono bakanywa amarwa

Fig. 11. Example of verb tenses error.

Semantic errors occurred when action had to be explained by more than one word or the word used in a specific context never made sense and Fig. 11 shows an example of the former. Verbs contributed most of the errors through verb tenses and phonology vowel change (morphophonological transformation).

V. Conclusion

In this paper, we have formalized the Ekegusii grammar in the interlingual GF ecosystem. In extending the GF resource grammar, we have provided a computational grammar with a BLEU score of 55.95% that is a significant step toward creating a basic language resource kit (BLARK) for this under-resourced language. The grammar provides the platform of building controlled applications on top of it besides generating bilingual corpus, which can be used to experiment with data driven approaches. The future work proposed is to work on the morphophonological transformation in the verb phrase to increase accuracy.

References

[1] Ghilic-Micu, B., Mircea, M., & Stoica, M. (2011). Knowledge based economy–technological perspective: implications and solutions for agility improvement and innovation achievement in higher education. Amfiteatru Economic Journal, 13(30), 404-419.
[2] Bender, E. M., Flickinger, D., & Oepen, S. (2008). Grammar engineering for linguistic hypothesis testing. In Proceedings of the Texas Linguistics Society X Conference: Computational linguistics for less-studied languages (pp. 16-36).
[3] Krauwer, S. (2003). The basic language resource kit (BLARK) as the first milestone for the language resources roadmap. Proceedings of SPECOM 2003, 8-15.
[4] Guthrie, M. (1948). The classification of the Bantu languages. Pub. For the International African Institute by the Oxford Univ. Press.
[5] Wagner, G. (1970). The Bantu of Western Kenya: with special reference to the Vugusu and Logoli. Oxford University Press.
[6] Otiso, Z. (2008). The morpho syntactic analysis of Ekegusii verb derivations in the Minimalist program. Masters Dissertation Kenyatta University, Kenya.
[7] Ranta, A., El Dada, A., & Khegai, J. (2009). The GF resource grammar library. Linguistic Issues in Language Technology, 2(2), 1-63.
[8] Ranta, A. (2011). Grammatical Framework: Programming with multilingual grammars (Vol. 173). Stanford: CSLI Publications, Center for the Study of Language and Information.
[9] Paikens, P., & Gruzitis, N. (2012, May). An implementation of a Latvian resource grammar in Grammatical Framework. In LREC (pp. 1680-1685).
[10] Angelov, K. (2011). The mechanics of the Grammatical Framework. Doctoral Dissertation, Chalmers University, Sweden.
[11] Ljunglöf, P. (2004) Expressivity and Complexity of the Grammatical Framework. Doctoral Dissertation, Chalmers University, Sweden.
[12] Ranta, A.. (2006). Type Theory and Universal Grammar. Philosophia Scientiæ. Travaux d'histoire et de philosophie des sciences, (CS 6), pp.115-131.
[13] Détrez, G., & Ranta, A. (2012, April). Smart paradigms and the predictability and complexity of inflectional morphology. In Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (pp. 645-653).
[14] Ranta, A. (2007). Modular grammar engineering in GF Research on Language and Computation, 5(2), 133-158.
[15] Kituku, B., Nganga, W., & Muchemi, L. (2019). Towards Kikamba computational grammar. Journal of Data Analysis and Information Processing, (pp 250-275).
[16] Kituku, B. (2019). Grammar engineering for Swahili. International Journal of Computer and Information Technology (pp 194-200).
[17] Bamutura, D., Ljunglöf, P., & Nebende, P. (2020, May). Towards Computational Resource Grammars for Runyankore and Rukiga. In Proceedings of The 12th Language Resources and Evaluation Conference (pp. 2846-2854).
[18] Ombui, E., & Wagacha, P. W. (2014, June). InterlinguaPlus Machine Translation Approach for Local Languages: Ekegusii & Swahili. In Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages (pp. 68-72).
[19] Elwell, R (2008). Finite state methods for Bantu verb morphology. Proceedings of Texas Linguistic.
[20] Ombui, E. O., & Muchemi, L. (2015). Wiring Kenyan languages for the global virtual age: An audit of the human language technology resources.
[21] Osinde, K.N. (1988). Ekegusii morphophonology: an analysis of the major consonantal process. Masters Dissertatiom. Nairobi University.
[22] Basweti, N.O.(2005). A morphosyntactic analysis of agreement in Ekegusii in the minimalist program. Masters Dissertation. Nairobi university. Kenya
[23] Ongarora,D,O. (2008). Bantu Morphostmtax: a study of Ekegusii. Doctoral Dissertation, Iawaharlal Nehru University. India.
[24] Ashton, E.O. (1947). Swahili grammar, Including Intonation. 2nd Edition. London:Longmans.
[25] Reichenbach, H. (1947). The tenses of verbs. Time: From Concept to Narrative Construct: a Reader, pp.1-12.
[26] Munyao, K.M. (2006) The Morph Syntax of Kikamba Verb Derivations: A Minimalist Approach. The University of Nairobi, Nairobi, Kenya.
[27] Whiteley W.H (1965) An Introduction to the Kisii. Nairobi: EALB.
[28] Bitutu, T. (1991). The Syntactic Patterns of Code Switching in Ekegusii-English. Nairobi: Unpublished MA Thesis. Kenyatta University.
[29] Rugemalira, J.M. (2007) The Structure of the Bantu Noun Phrase. SOAS Working Papers in Linguistics , 15, 135-148.
[30] Carr, M., & Verner, J. (1997). Prototyping and software development approaches. Department of Information Systems, City University of Hong Kong, Hong Kong, 319-338.
[31] Hammarström, H., & Ranta, A. (2004, March). Cardinal Numerals Revisited in GF In Workshop on Numerals in the World's Languages, Leipzig, Germany.
[32] Mbuvi, M.K. (2005) The Syntax of Kikamba Noun Modification. Unpublished Master's Dissertation, University of Nairobi, Kenya.
[33] Vilar, D., Xu, J., Luis Fernando, D.H. and Ney, H. (2006) Error Analysis of Statistical Machine Translation Output. LREC 2006, Genoa, Italy, 697-702.
[34] Koehn, P. (2004) Statistical Significance Tests for Machine Translation Evaluation.Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing , Barcelona, Spain, July 2004, 388-395.
[35] Levenshtein, VI (1966) Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Soviet Physics Doklady, 10, 707-710.
[36] Costa, Â., Ling, W., Luís, T., Correia, R., & Coheur, L. (2015). A linguistically motivated taxonomy for Machine Translation error analysis. Machine Translation, 29(2), 127-161.
[37] Bojar, O. (2011). Analyzing error types in English-Czech machine translation. The Prague Bulletin of Mathematical Linguistics, 95(1), 63-76.