International Journal of Computer and Information Technology (ISSN: 2279 – 0764)
Volume 08 – Issue 06, November 2019

Grammar Engineering for Swahili Language

Benson Kituku
Department of Computer Science
Dedan Kimathi University of Technology
Nyeri, Kenya
Email: Benson.kituku [AT] dkut.ac.ke

Abstract— **Most of the African languages are under resourced languages hence suffer from data sparsity due to lack of sufficient digital corpora making data driven methods not efficient for developing language technology resources. However, the availability of digital devices and ubiquitous computing demands these low-density languages to have language resources for application purposes. Therefore, this paper describes the engineering of Swahili grammar using Grammatical Framework (GF), a rapid grammar writing tool and formalism. A morphology rule based driven approach has been used where morphology is developed first, then followed by the syntactic part. The typical evaluation metrics BLEU and PER metrics were used to evaluate the grammar resulted in encouraging 77.95% and 9.46% respectively. The work is a significant step for the low resourced Swahili language since it provides a morphological analyzer and interlingua machine translation in the GF ecosystem which is useful in the analysis and generation of the language. Finally, the grammar lays a foundation for the development of controlled natural language applications on top of the Swahili grammar and the platform for extracting bilingual corpus for use in data driven methods.**

Keywords— ***Computational grammar, Grammatical Framework, low density language, morphology, syntax, inflection***

I. INTRODUCTION

The exponential growth of the internet and computers, coupled with high mobile phone penetration, has led to great demand for machine-human communication in the global information space. To minimize the language barrier (machine to human) for the under resourced languages, then grammar engineering is of great importance. This paper describes the development of computational grammar for low density Swahili language, which lays a foundation for the development of domain-specific application and production of other technologies.

The Swahili language belongs to the large Bantu family and is one of the official languages of Kenya and Tanzania, commanding millions of speakers. Guthrie [1] classified it under zone G, group 40, language 2[G42]. The language grammar is highly agglutinative, inflective and uses the nominal class system (class gender) and concord for noun agreements [2, 3, 4]. Nominal[1] class system [2] is based on morphology (affix to a noun stem) or syntax (agreement affixes to verbs) and the latter has been used in this work. Two noun classes based on the number (singular and plural) forms class gender [5]. Table I summarizes all the class gender in the Swahili language.

Swahili language, though widely used in written and formal communication, very few computational resources are available. Hurskainen [6] and Lipps [7] have developed a Swahili morphology analyzer using the finite-state approach, on the other hand, De Pauw [8] has also developed morphology analyzer using data driven approach. Nganga [9] developed a partial morphology analyzer using GF, that this paper has improved to include all categories plus the syntax. Finally, there exists a bilingual Machine translation between Egekusii and Swahili based on the carabao system [10] plus the google[2] translation system available online. Therefore, at the moment, there is no available computational grammar for the Swahili language which can be used to develop applications.

TABLE I. SWAHILI CLASS GENDER

Class Gender		
Syntax	Morpho	GF
a_wa	m_wa	G1
u_i	m_mi	G2
li_ya	ji_ma	G3
ki_vi	ki_vi	G4
i_zi	n_n	G5
u_zi	u_u	G6
u_u	u_u	G7
u_ya	u_u	G8
ya_ya	n_n	G9
i_i	n_n	G10
ku_ku	ku_ku	G11
pa_pa	pa_pa	G12
mu_mu	mu_mu	G13

II. GRAMMATICAL FRAMEWORK

Grammar engineering is the process of using formal grammar theories to create a grammar that machine can parse and/or generate and requires grammar formalism, grammar development toolkit and algorithms [21]. GF is a toolkit based on functional programming paradigm (types and modules), the logic framework of abstract plus concrete syntax and categorical grammar formalism and used for the rapid development of multilingual grammar resources and applications [11,12] and encompasses the requirement for grammar engineering. GF allows the development of resource grammar that covers syntactic and morphological parameters and principles of a language for general wide coverage use. Categories and functions declared at abstract syntax are the ingredients for semantic constructions that help to build trees [12]. In addition, concrete syntax provides a

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

[2] https://translate.google.com/#view=home&op=translate&sl=auto&tl=en&text=wewe%20waja

way of mapping the abstract syntax trees into strings of the specific language. There are several concrete syntaxes but one abstract syntax thus the Interlingua ecosystem. Parsing transforms language-specific concrete syntax into an abstract tree (language analysis), while linearization transforms abstract trees to string in a specific language (language generation). GF uses parameter defined by keyword *param* to capture grammar features need by a category for inflection and uses functions known as operation defined by keyword oper) to implement the inflection table. The operation is implemented as a smart or low-level paradigm

III. THEORETICAL BACKGROUND

Formal grammar given by definition 1 uses lexical rules and syntax rules to formalize a natural language grammar [13, 14] The terminal inflects depending on the grammar features of a specific category, e.g. number, case, person etc. The inflection is modeled using a regular expression (algebraic way of specifying inflection pattern in a language) given by definition 2[15].

Definition 1: Formal grammar G is 4 tuple G= (N, S, P, T)

- N is Finite set of variables(Non-terminals) which can be replaced by other variables or terminals
- T is Terminals or actual words in the language
- S is a special non-terminal where all derivation start called start symbol
- P are production rules describing how to replace grammar symbols

Definition 2: Ways of building a regular expression

- ε use of empty morpheme
- a use of single morpheme
- a| b union of more than one morpheme
- a.b concatenation of more than one string
- a* recursive concatenations of zero or more of morpheme a

In the next subsections, we describe the morphology of the categories and syntax structures of Swahili phrases.

A. Morphology

Morphology is a way of building words from morphemes or generating word forms [15]. The Swahili language is an agglutinative language and its morphology is affected by Morph phonological transformation. The noun class gender (concord) influencing the morphology of all categories through a prefixing morpheme. Throughout this paper, the use of the syntax class gender (concord) has been adopted. The noun structure consists of singular(sg) and plural(pl) prefixes that form the class gender as per Table 1, followed by the root and optional suffix" ni" which results in location case [3,16] Example 1 exemplify noun morphology.

Example 1

Singular	**Plural**
m-ti	mi-ti
G2_sg –root	G2_pl -root
Tree	Trees

The adjective which modifiers noun consist of the prefix (concord) that must agree with the class gender of the noun been modified and is concatenated with the adjective root. The concatenation is influenced by morph-phonological rules [3, 4]. Example 2 demonstrates the inflection of adjectives. It is essential to note the inflection of numbers (one, two, three, four, five and eight) follows the adjective pattern, while the rest are independent of the class gender

Example 2

Singular		**Plural**	
M-ti	mu-dogo	mi-ti	mi-dogo
G2_sg -root	G2_sg -adjroot	G2_pl -root	G2_pl -adjroot
Small	tree	Small	trees

Verbs are the most complex category in any Bantu language and consisting of many particles(morphemes) that are conjunctively in nature. The Swahili verb uses grammar features: polarity (positive and negative represented by the subject marker and negation respectively), Tense and anteriority (simultaneous and anterior) and person (P1, P2, P3) The Table II summaries all the morphemes possible in constructing of a verb in Swahili [3, 16]. The subject marker, tense, root and final vowel are the obligatory morphemes the rest are optional. The subject marker stands in place of a noun; for example, in Table 2 the morpheme *"tu"* stands for the pronoun *"we"* in English. Five tense exists in Swahili namely: present tense, habitual tense, past tense, future tense and conditional tense [1,9,16,17] and have the morpheme –na-, hu-,-li-, -ta- and -ngali- respectively as exemplified in Table iii.

TABLE II. VERB ARCHITECTURE

Architecture	Morpheme	Swahili
Prefixes	Negation	as per class gender
	Subject marker	as per class and person
	Tense/Aspect	As per tense
	Relative marker	As per class
	Object marker	as per class
	Infinitive	"ku"
root		Root
extension suffix	Applicative	'' e/i"
	Causative	'' ish/esh"
	Passive	''w "
	Reversive	"u/ul"
	Reciprocal	'' an"
	Stative	"ik:
Final vowel		"a/e/i"

TABLE III. VERB TENSE

Tense	**Swahili**	**Gloss**
Present	Tu-na-lala	We are sleeping
Habitual	Hu-lala	We sleep
Past	Tu-li-lala	We slept
Future	Tu-ta-lala	We will sleep
Conditional	Tu-ngali-lala	We would sleep

In terms of the closed categories: determiners (e.g., that, these, those) are strings which inflect for class gender and number (singular and plural) [2,3]. Through the elicitation process, it was established some preposition inflect for class gender and number, for example "of" while others have independent strings. The adverb category does not inflect.

B. *Syntax*

SVO (Subject Verb Object) is the Swahili language central topology for a sentence [3, 4, 16, 17]. The noun phrase is the subject, while the verb phrase represents the verb. The argument of the verb phrase depending on the verb valence forms the object that can be a noun phrase or verb phrase or both. The lexical items use concord to form syntactic agreement. Since the Verb has a subject marker that stands in place of the noun phrase while the object marker stand in place of the object implies the verb phrase can act as a full sentence.

A noun phrase consists of a noun and its modifiers that include: adjective (Adj), numbers (num), determiner (Det) whether possessive (poss) or demonstrative (dem) [18] and they order is per equation 1. Besides, the personal pronoun is treated as NP by themselves. The verb phrase takes all the features of verb plus agreement

[[dem] [Noun] [Det <poss> <dem>] [[Num] [Adj]]   (1)

IV. IMPLEMENTING THE GRAMMAR IN GF

Experts of Swahili, books and postgraduate theses on Swahili grammar, dictionaries and journal papers were the sources of descriptive grammar and lexicons. Bottom-up rule-based morphology driven methodology was used to develop computational grammar based on the functional approach of GF. The part of speech tags morphology was modeled first then followed by the syntax. In GF, the Cgender for class gender was used.

A. *Noun*

The inflection of noun required three grammar features: class gender, number (singular and plural) and case (normative and locative). The regular expression regN and compoundN were used to model noun inflection with the former been used for simple noun and latter for the complex noun, which consists of more than one string. The function iregN was used for an irregular noun which listed all forms. Fig 1 shows the implementation of the regular expression while table IV output of regular expression compoundN using string "university" in the Swahili language.

TABLE IV. NOUN INFLECTION

Lang> l -lang=Kis -table university_N
s Sg Nom : chuo kikuu
s Sg Loc : chuoni kikuu
s Pl Nom : vyuo vikuu
s Pl Loc : vyuoni vikuu

B. *Adjectives*

The noun concord prefix which agrees with class gender and number is conjunctively attached to the root stem [11,13]. In some instances, the prefix is affected by the phonological process. In regular adjectives, the concord is attached as a prefix to the adjective root. regA regular expression was used to

```
compoundN : N -> N ->Cgender-> N = \chuo,kikuu,g ->
      { s = \\n,c =>chuo.s! n! c ++ kikuu.s!n! Nom ;
        g = g ; lock_N = <> }

regN : Str ->Cgender -> Noun =  \w, g ->
   let wpl = case g  of {
  G1=>case w of {
      "mwa" + _  => PrefixPlNom G1  + Predef.drop 3 w ;
        "mwi" + _  => "we"  + Predef.drop 3 w ;
         "ki" + _  => PrefixPlNom G4  + Predef.drop 2 w ;
         "m" + _  => PrefixPlNom G1  + Predef.drop 1 w ;
                _   =>  w };
  G2=>case w of {
       "mw" + _  => PrefixPlNom G2  + Predef.drop 2 w ;
        "mu" + _  => PrefixPlNom G2 + Predef.drop 2 w ;
               _  => PrefixPlNom G2  + Predef.drop 1 w };
G4=> case w of {
         "ki" + _  => PrefixPlNom G4  + Predef.drop 2 w ;
            "ch" + _  => "vy" + Predef.drop 2 w ;
              _   =>  w };
G6 |G8 => PrefixPlNom g  + Predef.drop 1 w;
  G11 |G12|G13 => "" ;
  _ => PrefixPlNom g + w };
            in iregN w wpl g ;
iregN :Str-> Str ->Cgender -> Noun= \man,men,g -> {
     s = table{
   Sg => table{Nom => man ;  Loc=> man + "ni"   };
    Pl => table{Nom => men ; Loc=> men + "ni" }} ;
     g = g  } ;
```

Fig 1. Noun Smart paradigm

implement simple adjective while cregA was used to implement complex adjectives such as colors which take a preposition, string and stem. The function *VowelAdjprefix* captures the phonological effects on the word-formation. Fig 2 exemplifies the two regular expressions, while Table V demonstrates an example using "big" and color brown as adjective examples.

TABLE V. ADJECTIVE INFLECTION

Lang> linearise -table big
s (AAdj G1 Sg) : mkubwa
s (AAdj G1 Pl) : wakubwa
. . . .
s (AAdj G13 Sg) : mkubwa
s (AAdj G13 Pl) :
Lang> linearise -table brown_A

```
s (AAdj G1 Sg) : wa rangi ya hudhurungi
                  . . . . . . . .
s (AAdj G13 Sg) : mwa rangi ya
hudhurungi
```

```
regA:Str -> {s : AForm => Str} = \seo -> {s = table {
AAdj G1 Sg=>case Predef.take 1 seo of {
"a"|"e"|"i"|"o"|"u" => VowelAdjprefix G1 Sg + seo;
        _ => ConsonantAdjprefix G1 Sg + seo };
                  . . . . . . . . . . . . . . . . . .
AAdj G13 Sg=>case Predef.take 1 seo of {
     "a"|"e"|"o"|"u" => VowelAdjprefix G13 Sg + seo;
"i" => VoweliAdjprefix G13 Sg + seo;
_ => ConsonantAdjprefix G13 Sg + seo AAdj _ Pl =>[]
}};

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

Fig 2. Adjective Smart paradigms

C. *Verbs and Verb Phrases*

The Grammatical Framework resource library by default provides positive and negative polarities, past, present, future, and conditional tenses and finally, simultaneous, and anterior [12]. The positive polarity was implemented using the subject marker morpheme, while the negative polarity the negation morpheme was used. The two morphemes require extra grammar features in order to allow agreement, namely: class 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 are also used to implement the verbs.

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

The operation of the verb has a record of four strings: string s is the various forms of verbs that can be generated in a specific language. The verb forms were: infinitive, extensional or derivative morphology form, general form with a final vowel ”a”, habitual and present negation form. The second record string as progV for the progressive verb, then inf for infinitive verb plus an imperative verb. The imperative verb inflects for polarity and parameter impForm (number and Boolean with the true been polite request while false been command). The smart paradigm regV and iregV is shown below implemented the best and worst-case regular expression using low-level mkVerb that generates an inflection table of 1267 words forms.

```
regV :Str -> Verb =\vika -> let  stem = init
                             vika in
  mkVerb vika (stem+"i") ("ku"+vika)("hu" +
                        vika ) ;
  iregV : Str -> Verb =\vika -> mkVerb vika
                       vika vika vika ;
mkVerb :(gen,preneg,inf,habit : Str) -> Verb=
             \gen,preneg,inf,habit ->
                        { s =table{
                      VPreNeg   => preneg;
                           VGen => gen;
                           VInf => inf;
                   Vhabitual =>habit;
                 VExtension type=> init gen +
                 extension  type

s1  =\\  pol,tes,ant,ag  =>  letv_prefix  =
           (polanttense.s!pol!tes!ant!ag).p1   ;
          in  case < tes, ant,pol > of {
    <Pres, Simul, Neg> =>  v_prefix + preneg ;
       <Pres, Simul,Pos> =>  v_prefix + gen;
        <_, _,_> => v_prefix +gen
 progV = [];
s2=\\pol,tes,ant,ag =>  case < tes ,pol> of {
<Pres,  Neg>  =>(polanttense.s!Neg!Pres!Simul!
ag).p1 + preneg  ;
     <_,  _> =>(polanttense.s!Pos!Pres!Simul!
                     ag).p1 + gen};
      imp=\\po,imf => case <po,imf> of {
                        <Pos,ImpF Sg False> =>
                       gen;
                        <Pos,ImpF Pl False> =>
              case last gen of {
                      "a"   => init gen +"eni";
                        _  => gen + "ni"    };
                         <Pos,ImpF Sg True> =>
              case last gen of {
                        "a"  => "u" + init gen
                     +"e";
                        _  => "u" + gen   };
                         <Pos,ImpF Pl True> =>
              case last gen of {
                        "a"  => "m" + init gen
                     +"e";
                        _  => "m" + gen   };
                     <Neg, ImpF Sg _> => "usi"
             + init gen +"e"    ;
                      <Neg,ImpF Pl _> => "msi"
             + init gen +"e"     }

                               };
```

The Verb phrase was implemented using smart paradigm regVP with five record strings: *s* for the general verb, *progV* for progressive verbs, *compl* for the object of the verb, *imp* for imperative verbs and *inf* for infinitive verbs. The subcategorization of verbs was taken care of through *compl* (one place, two-place, and three-place verb) which could be a verb phrase, noun phrase or adverbs, passivation or a combination of any. Twenty rules were modeled for the syntax phase for VP.

```
regVP run  = {
s =\\ ag,pol,tes,ant =>run.s1!pol!tes!ant!ag;
 compl=\\_=> [];
progV = run.progV;
imp=\\po,imf => run.imp!po!imf;
inf= run.s!VInf };
```

D. Numerals

Numeral can either be cardinal or ordinal. Cardinal describes quantity while ordinal shows order. Numerals can exist as digits or in words form. In GF words are in category numeral while digit in digits. The parameters DForm and CardOrd, as explained below, were used for numerals.

Parameter DFORM

- *unit* – presenting numbers from zero to nine
- *teen* – representing numbers from eleven to nineteen
- *ten* – resenting multiples of ten, e.g. twenty, etc.
- *hund* -representing multiples of hundred

parameter CardOrd.

- *NCard* – represent the cardinal numbers
- *Nord*- represent the ordinal numbers

In addition to the above parameters, numeral and digits categories have variable features of class gender and inherent feature of number. The only numeral one is singular; all others are plural. GF[3] implement numbers in the range of 0-999,999. The regular expression mkNum was used to generate numbers between one and five and its multiples which depend on class gender while between six and nine regNum was used to generate the various forms. Thousands were formed from the syntax. The regular expressions are demonstrated in Fig 3.

```
mkNum : Str -> Str -> {s : DForm => CardOrd =>
Cgender => Str} = \two, second -> {s = table {
    unit => table {NCard =>\\g => Cardprefix g + two ;
            NOrd => \\g => Ordprefix g ++ two} ;
    teen => table {NCard =>\\g =>"kumi na" ++
        Cardprefix g + two ;
        NOrd => \\g => Ordprefix g ++ "kumi na" ++
        Cardprefix g + two} ;
    ten => table {NCard =>\\g =>second ;
         NOrd => \\g => Ordprefix g ++ second ++
        "na" ++ Cardprefix g + two};
    hund => table {NCard =>\\g =>"mia " ++ two ;
        NOrd => \\g => Ordprefix g ++ "mia " ++ two }
    } } ;

regNum : Str ->Str -> {s : DForm => CardOrd =>
Cgender => Str} = \six,sixth -> {s = table {
   unit => table {NCard =>\\g => six ;
            NOrd => \\g => Ordprefix g ++ six} ;
  teen => table {NCard =>\\g =>"kumi na" ++ six ;
    NOrd => \\g => Ordprefix g ++ "kumi na" ++ six} ;
  ten => table {NCard =>\\g =>sixth ;
   NOrd => \\g => Ordprefix g ++ sixth ++"na" ++ six };
 hund => table {NCard =>\\g =>"mia " ++ six ;
        NOrd => \\g => Ordprefix g ++ "mia" ++ six}
    } } ;
```

Fig 3. Numbers smart Paradigms

E. Noun phrase

Swahili language NP just like the other inflective languages [16, 17] the morphological features determine the agreement of the phrase constituents. However, it is essential to note in the Swahili language the common noun (CN) is the same as a noun since there are no articles and this may lead to over generation when translating from it to other languages that have articles. In addition, CN was designed with two strings. The first to hold the CN and the second to be empty in order to allow the holding of the adjective. Unlike Indo- European languages demonstratives and numbers which come later are added in between the noun and adjective as shown by word alignment of NP in Fig 4 thus the configuration was to allow later and future insertion of demonstrative and numbers. Over forty rules were implemented for the noun phrase

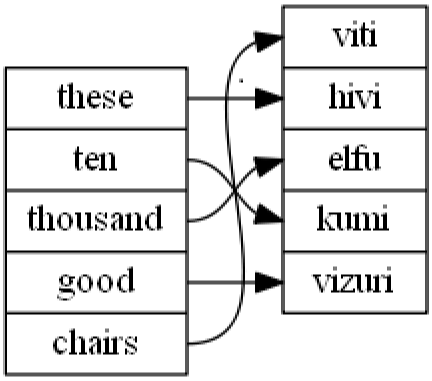

Fig 4 Noun phrase

F. Clause and sentences

Clauses have undetermined polarity, tense, anteriority which is fixed at the sentence level and three types declarative, question and relative clauses were modeled using SO topology. The two ways used to form question clauses were through the yes or no answer question or through Interrogative which are interrogative Pronouns, interrogative, Adverbs, Interrogative Quantifiers, copula interrogative complement and their modifiers. The relative clauses are formed by using a pronoun, in Addition to the verb phrase and sentence missing a noun phrase been modified by a relative pronoun.

V. RESULTS AND EVALUATION

The objectives of testing and evaluation were to check the quality of grammar and coverage. Parse trees will be used to demonstrated coverage while evaluation metrics for the quality of the grammar. The regular expressions were constructed for each inflective category while 163 syntax rules were constructed for the whole grammar

Fig 6 demonstrate coverage of syntax and morphology. The sentence "these small people will cut many trees." NP is formed by "these small people" while VP by "will cut" which is in the future tense, positive polarity and simultaneous anteriority while the compliment is an NP "many trees" thus the structure SVO here is formed by NP VP NP.

[3] https://www.grammaticalframework.org/lib/doc/gf doc/Numeral.html

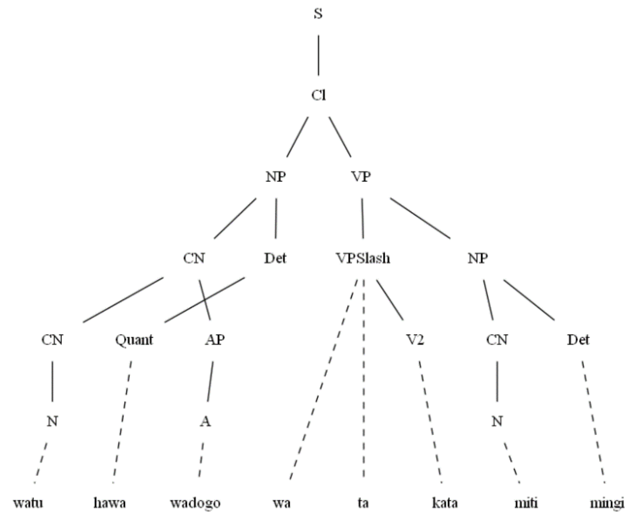

Fig. 5 Sentence parse tree

The preposition usage is demonstrated by Fig 6 and 7 in that the preposition “of” becomes “ya” in class gender G5 number singular while the preposition “on” combine with the noun “Meza” (table) to form “mezani” (on the table). Therefore, some of the preposition is infused into the noun while others are not. Regression testing [12, 19] was done during grammar development in order to ensure quality and also coverage. The full grammar was evaluated using bilingual evaluation understudy (BLEU) and position-independent error rate (PER) evaluation metrics were used to check the correctness and coverage in the resource grammar.

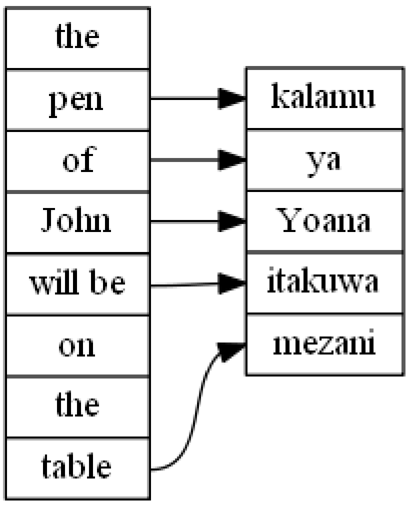

Figure 6. word alignment

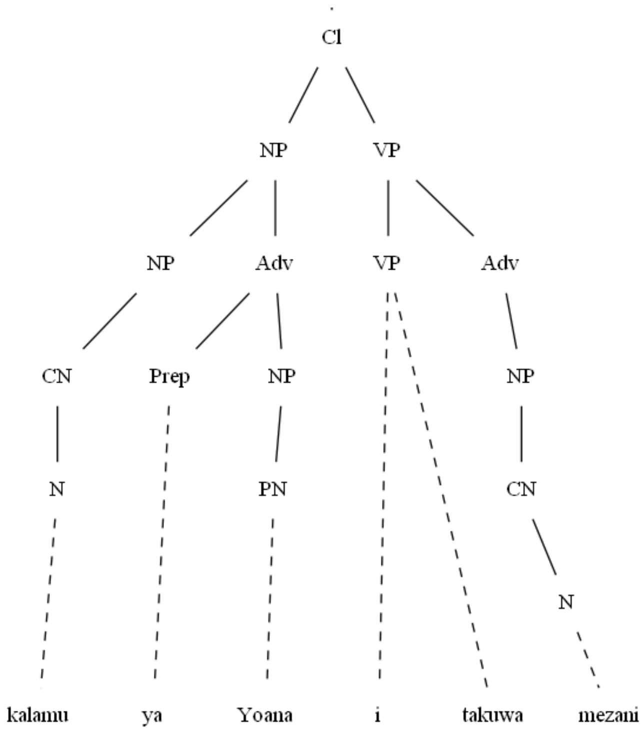

Fig 7 prepositional usage

The regression testing involved translating a specific category lexicon or phrase in a rule from the English language into a gold standard in the Swahili language, then implementing the function and comparing the outcome and if they differ, repeating the procedure until the GF output matches the Gold standard. This was done for each function that was implemented. Finally, BLEU and PER evaluation was applied after completing developing the Swahili grammar whereby100 English sentences were developed half from online GF treebanks[4] and Khegai [20] Russian work and the rest were developed by Swahili linguist from the over 500 open and closed categories lexicon provide in GF and translated to Swahili Gold standard so as to test the unique features of Swahili languages. All sentences first were in English. The Swahili linguist translated all of them to Swahili and become the reference (Gold standards) and through computational grammar, the machine translation (target) was obtained.

The results were subjected to an online tilde[5] machine translation evaluator which gave a promising average Bleu score of 77.95% based on cumulative 4-gram and PER of 9.46%. An examination of the output from the evaluation shows some words in Swahili language depending on context use different word forms and maybe there is need for more than one target file in order to take care of variation. Below is an illustration with the English word “break”. The colored part shows the difference.

Source the heavy questions break students
Human maswali mazito yanatatiza wanafunzi
Machine maswali mazito yanaharibika wanafunzi

[4] https://github.com/GrammaticalFramework/gf-rgl/tree/master/treebanks

[5] https://www.letsmt.eu/Bleu.aspx

VI. CONCLUSION

In this paper, the first wide coverage computational grammar for less-resourced Swahili language has been developed. which can enjoy translation to and from more than 30 other languages in GF. The morphology and syntax have been well covered and the translation from English to Swahili producing a high score of 77% BLEU scale reflects the ability of the grammar to be used for analysis and generation work. Computational grammar can be used to develop application grammars and controlled language application. For future work, I do propose an investigation of how grammar sharing can be used to develop a computational grammar for Bantu languages which are closely related to Swahili languages.

REFERENCES

[1] Guthrie, Malcolm. *The classification of the Bantu languages*. Pub. for the International African Institute by the Oxford Univ. Press, 1948.

[2] Ashton, Ethel O. "Swahili grammar, including intonation." (1947)

[3] Njogu, Kimani, and Dorothy K. Nganje. *KiSwahili kwa vyuo vya ualimu*. Jomo Kenyatta Foundation, 2006.

[4] Mwamzandi, Mohamed Yusuf. "Swahili word order choices: insights from information structure." (2014).

[5] Kaviti, L. K. "A minimalist perspective of the principles and parameters in Kikamba morpho-syntax." *Unpublished PhD Dissertation, University of Nairobi* (2004).

[6] Hurskainen, Arvi. "A two-level computer formalism for the analysis of Bantu morphology." *Nordic journal of African studies* 1.1 (1992): 87-119.

[7] Lipps, Jonathan. "xsma: A Finite-state Morphological Analyzer for Swahili." 2011

[8] De Pauw, Guy, and Gilles-Maurice De Schryver. "Improving the computational morphological analysis of a Swahili corpus for lexicographic purposes." *Lexikos* 18.1 (2008).

[9] Ng'ang'a, Wanjiku. "Building Swahili resource grammars for the Grammatical Framework." *Shall We Play the Festschrift Game? Essays on the Occasion of Lauri Carlson's 60th Birthday* (2012): 215-226.

[10] Ombui, Edward, and Peter Wagacha. "InterlinguaPlus Machine Translation Approach for Local Languages: Ekegusii & Swahili." *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*. 2014.

[11] Paikens, Peteris, and Normunds Gruzitis. "An implementation of a Latvian resource grammar in Grammatical Framework." *LREC*. 2012.

[12] Ranta, Aarne. *Grammatical Framework: Programming with multilingual grammars*. Vol. 173. Stanford: CSLI Publications, Center for the Study of Language and Information, 2011.

[13] Wang Y, Berwick RC. Towards a formal framework of cognitive linguistics. Journal of Advanced Mathematics and Applications. 2012, pp. 250-63.

[14] Mogensen, Torben Ægidius. *Basics of compiler design*. Torben Ægidius Mogensen, 2009.

[15] Jurafsky, Daniel, and James H. Martin. "Speech and Language Processing: An introduction to speech recognition, computational linguistics and natural language processing." *Upper Saddle River, NJ: Prentice Hall* (2008).

[16] Deen, Kamil Ud Deen Salah Ud. *The acquisition of Nairobi Swahili: The morphosyntax of inflectional prefixes and subjects*. Diss. University of California, Los Angeles, 2002.

[17] Marten, Lutz. "Formal syntax and African languages: Dynamic Syntax case studies of Swahili, Bemba, and Hadiyya." *SOAS Working Papers in Linguistics* 16 (2013): 195-213.

[18] Rugemalira, Josephat M. "The structure of the Bantu noun phrase." *NC Kula and L. Marten,(eds.)* (2007).

[19] Camilleri, John J. *A computational grammar and lexicon for Maltese*. Diss. Master's thesis, Chalmers University of Technology. Gothenburg, Sweden, 2013.

[20] Khegai, Janna, and Aarne Ranta. "Building and using a Russian resource grammar in GF." *International Conference on Intelligent Text Processing and Computational Linguistics*. Springer, Berlin, Heidelberg, 2004.

[21] Bender EM, Flickinger D, Oepen S. Grammar engineering for linguistic hypothesis testing. InProceedings of the Texas Linguistics Society X conference: Computational linguistics for less-studied languages 2008 (pp. 16-36).