Nordic Journal of African Studies 13(3): 363–397 (2004)

Swahili Language Manager: A Storehouse for Developing Multiple Computational Applications

ARVI HURSKAINEN
University of Helsinki

ABSTRACT

Swahili Language Manager (SALAMA) is a computational environment for managing written Swahili language and for developing various kinds of language applications. Having been subject to development since 1985, it currently (2004) contains the Standard Swahili lexicon as fully as possible. As it is a system for managing the language, it includes also the full morphological and morpho-phonological description of Swahili, a rule-based system for solving the word level ambiguities, a rule-based system for tagging text syntactically (including alternatively a shallow Constraint Grammar parsing or a deep Dependency Grammar parsing), a rule-based system for handling idiomatic expressions, proverbs and other non-standard clusters of words, and a semantic tagging and disambiguation system for defining correct semantic equivalents in English. SALAMA facilitates also a raw translation from Swahili to English, including the correct surface forms in English (e.g. verbs, nouns and adjectives) and transfer rules for the correct English word order. An essential part in developing and testing SALAMA is the Helsinki Corpus of Swahili, which has been under construction since 1988 and is currently globally available at the Language Bank of Finland (www.csc.fi). The paper discusses all these features in detail.

Keywords: computational linguistics, language technology, morphology, syntax, disambiguation, machine translation

1. INTRODUCTION

Thirteen years ago I published the first description of SWATWOL, a two-level morphological analyser of Swahili (Hurskainen 1992), which was then to develop into SALAMA[1], a comprehensive language management environment. In order to make maximal use of resources in developing computational language application tools, it is advisable to construct a comprehensive system that facilitates various kinds of applications. Whatever the application is, it is likely to benefit greatly, if it can make use of the explicit linguistic information in the text. Language is encoded in a written form, and although all the linguistic information is covertly there, it is very hard to make use of it in computational applications if it is not clearly spelled out, in other words, if it is not properly analysed by providing the readings with appropriate tags.

[1] An early version of SALAMA was described in Hurskainen (1999).

This problem has been partly solved in approaches, where morphologically or even syntactically analysed text corpora have been made available to customers. It is expected that the users can make use of analysed texts and perhaps develop their own applications on that basis, without having to bother themselves about the basic linguistic analysis. For more advanced users and developers there are also online analysis programs available, so that the user may analyse also his own texts. Also this facility is becoming increasingly available, as it is for example in the Language Bank of Finland (www.csc.fi) for Finnish, Swedish and Swahili.

There are, however, problems in using tagged corpora and analysis programs. It often happens that the tagging system does not meet the requirements of the user, or that the tagging system is not accurate enough. The user has to post-process the analysis result, provided that he has skills for it. If one wants, for example, to develop a syntactic parser that takes the result of the morphological parser as an input, one is forced to handle the weaknesses of the morphological parser by trying to post-process the result, although the correct method would to fix the parser itself. In other words, the ideal case would be that the developer of the system has access to the language-specific modules of the morphological parser and skills to fix them. Therefore, the developer of a language management system for a language should have access to all modules, starting from the tokeniser, and a thorough understanding of each phase of the system. If this is guaranteed, there is hope that each of the problems is solved in the correct place and in an appropriate manner. While developing SALAMA, I have been fortunate in having worked out all the phases thus ensuring that there are no obscure results. The source of each faulty result can be immediately identified and the problem can be fixed in the optimal place and manner.

Below I will describe the main features of SALAMA. I will also describe briefly the various phases in developing the system.

2. INITIAL STEPS

After having returned from my third working period in Tanzania in 1985, I participated the course on two-level morphology held by Kimmo Koskenniemi, who had managed to describe the morpho-phonology of Finnish by using finite-state automata (Koskenniemi 1983). Because Swahili, like Finnish, is a morphologically complex language, I took as a challenge to try to describe Swahili morphology by using this cherished method. The task was a challenge also because of the word formation rules of Swahili. Most of affixes are prefixed either directly or via another prefix to the stem, and verbs have in addition also a fairly complex system of extensions.

TWOL, as the two-level system is called, takes a morphological dictionary and a rule file as input and performs morphological analysis. Earlier there was no automatic rule compiler and the morpho-phonological rules had to be compiled by had into a tabular format. Since the automatic compiler (Karttunen,

Koskenniemi and Kaplan 1987, Karttunen and Beesley 1992) became available, writing and testing rules has become commonplace.

Because the development of a full-scale analyser requires a large dictionary and fairly large amounts of text for testing, these resources had to be developed as well. At that time there were no texts or dictionaries in computer form. So the lexicon for the analyser had to acquired by scanning and editing a current dictionary, and first text corpora were compiled just by keying in. A number of students of African studies were of indispensable help in scanning and editing the texts.

By the end of the 1980's a fairly complete analyser, SWATWOL, was ready, and a description of it was published a few years later (Hurskainen 1992). There was also a field research project going on with the Institute of Kiswahili Research (Dar-es-Salaam) for surveying various forms of Swahili in Tanzania in 1988-1992 (Hurskainen 1995), and the outcome of this project also contributed to the development of SWATWOL. All the material of this project, the transcribed tapes as well as the word-lists, were put into computer form and made available globally to the researchers through the Helsinki University Language Corpus Server at the Institute of General Linguistics.

The basic version of SWATWOL was capable of recognizing and analysing some 10,000 common words of Swahili. Although the two-level rules for handling morpho-phonological variation had been tested and found reliable already by that time, the lexicon was still far from complete. It was expanded gradually with the material extracted from the accumulating corpus texts, as well as from lists of terminology coined by the National Swahili Council of Tanzania[2]. Texts, and new words, came from scanned books as well as from newspaper texts. When news texts started to appear in the Internet, beginning in 1997, the accumulation of the corpus accelerated, as well as the development of the SWATWOL lexicon.

As the acquisition of common lexical items continued, the need of handling special types of lexical units became important. Those include the multi-word concepts, reduplicated verbs, domain-specific terms, and particularly idioms and other such clusters of words that require special treatment.

There are at least three ways to handle multi-word concepts. In one method, each member of the multi-word unit is analysed as a separate word, and the interpretation of the cluster is left to the phase, where the rest of the syntactic analysis is carried out. In another method, especially if the cluster of words is fixed and unambiguous, the binding of the members together can be carried out already in the tokenizer. This requires that those multi-word units are listed in the same form also in the morphological lexicon. The method is simple and there hardly occurs morphological ambiguity in those word clusters. In the third method, the multiword unit is isolated after the morphological analysis by rules that make use of the full morphological information (Hurskainen 2004b). This

[2] The name of the council in Swahili is Baraza la Kiswahili la Tanzania (BAKITA).

method is suitable for handling idioms, for example, where the verb as a constituent part with a countless number of surface forms has to be handled.

Reduplication is quite common in Swahili, as it is in Bantu languages in general. Reduplicated forms of other word classes except verbs can be listed in the lexicon, because they have commonly a fixed form. The verbs are an exception and they require a different treatment. In verbs, the reduplicated part is the verb stem, including the final vowel. This concerns also the extended verb stems. I have tried to solve this problem by writing such two-level rules that would copy the verb stem in whatever form it appears in text. However, the rule compiler was not able to handle rules with more than five stem characters. So I had to abandon that approach. In the absence of a better method[3] for handling reduplication

3. THE BASIC FEATURES OF SWATWOL

As the name SWATWOL indicates, the TWOL parser described the language on two levels, which are conventionally termed as the lexical level and the surface level. It has also become customary that in descriptions the lexical level is located above and the surface level under it. In the system applied here (there are also other types of implementation), each character on the surface level is represented by a character on the lexical level, as shown in (1).

(1) Lexical level: `#mUalimu#`
Surface level: `0mwalimu0`

In (1) we see that the lexical character is normally realised as itself on the surface level. Only the lexical U is realised as w on the surface.

More complicated examples are in (2), where the noun prefix alternates on the surface.

(2) Lexical level: `#NIumbu#` `#NIbuzi#` `#NIta#`

Surface level: `0nyumbu0` `0m0buzi0` `0n0ta0`

Lexical level: `#NItaa#`
Surface level: `000taa0`

TWOL thus makes it possible to describe morpho-phonological variation in great detail and allows the description of phonological phenomena, if one wishes to use the system for such purposes.

[3] The Xerox tool package includes a convenient method for handling the reduplication problem by using composition (Beesley and Karttunen 2003). I am grateful to Lauri Karttunen for pointing out this possibility.

TWOL is constructed so that it takes the lexical description from the morphological lexicon, where the lexicon of the language is described, including all morphological features. So any grammatical word form of the language should be found described in the lexicon. If not, then the lexicon is faulty or defective.

Surface forms of the lexical strings are produced with the help of two-level rules. For example, there is a rule stating that U should be realised as w on the surface if there is a vowel on its right side. For the cases in (2) more than one rule is needed, because realisations are different and also the number of characters affected is two (N and I).

The major advantage of the two-level description is that the morphological lexicon can be kept fairly simple, because different surface variations can be controlled with rules.

One should remember, however, that many cases can be described with rules or without them. In theory the whole language can be described with the help of the lexicon only. It is rather a question of practicality than possibility which method of description should be adopted in each case. It is obviously not wise to write a rule for one or two words. But if the rule can apply to hundreds or thousands of cases, then rule writing is an obvious choice.

3.1. A two-level lexicon

Below in (3) is an example of how a two-level lexicon is constructed. The examples in (1) and (2) serve as words to be described.

(3)
(3a) LEXICON Start
Noun " N ";

(3b) LEXICON Noun
  NounClassPref1/2;
  NounClassPref9/10;

(3c) LEXICON NounClassPref1/2
mU NounClassStem1/2 "= 1/2-SG ";
wA NounClassStem1/2 "mU 1/2-PL ";

(3d) LEXICON NounClassStem1/2
alimu E "= { teacher } ";

(3e) LEXICON NounClassPref9/10
NI NounClassStem9/10 "= 9/10-SG";
NI NounClassStem9/10 "= 9/10-PL";

(3f) LEXICON NounClassStem9/10
umbu E "= { hartebeest }";
buzi E "= { goat }";

```
ta E "= { wax }";
taa E "= { lamp }";

(3g) LEXICON E
# #;
END
```

As can be seen in (3) above, the lexicon system is a tree structure. The recognition of each word starts from the beginning of the word and proceeds character by character to the end of the word. Each sub-lexicon represents a state. State (3a) contains only the beginning mark (#) of the string. The second part (Noun) on the line tells that the processing continues in the sub-lexicon called Noun. What is between double quotes will be given as output. The sub-lexicon Noun (3b) has only two continuation classes, which direct the processing to two possible sub-lexicons, each for separate noun classes. The sub-lexicon (3c) is for the noun prefixes of class 1/2. The equation mark immediately after the first double quote stands for the lemma form meaning that it is the same as the lexical string on that line. The second line in (3c) has *mU* after the double quote, indicating that the lemma should have *mU* (i.e. the singular prefix) attached to the stem. The sub-lexicon (3d) is for the stems of the class 1/2. In (3e) are the class prefixes for the class 9/10. The lexical prefix is *NI* is singular and plural. The sub-lexicon (3f) is for the stems of the class 9/10. Note that although each of the stems in (3f) receives the same prefix *NI*, on the surface level the realisations differ from each other. The sub-lexicon (3g) terminates the processing, and each string that survives until this point will be accepted as a grammatical word of Swahili. If the string does not pass through the network of the lexicon, it will be discarded as an ungrammatical word.

3.2.Two-level rules

As already mentioned, two-level rules regulate the surface form of the word. The noun class 9/10 is an example of how the lexicon structure can be kept simple although on the surface the variation is considerable. In (4) there is an example of a two-level rule.

```
(4) "Devocalisation of U in front of vowels"
    U:w  <=>  m  _  :V ;
```

The rule in (4) says that: change the lexical *U* into surface *w* if and only if on its left side there is *m* (realised on both levels as *m*), and on the right side there is a surface vowel.

The noun class prefixes of class 9/10 are much more difficult to handle, part of each of the rules in (5a-d) are needed to take care of them.

```
(5a)"Class prefix N > m (partial rule)"
      N:m   <=>   [#: | %*:]   _   I:0 [:b | :p | :v] ;
```

(5b)"Class prefix N > n (partial rule)"

```
     N:n  <=>  [#: | %*:]  _  I: [j: | :d | g: | z: | :V] ;
                           _  I: Cn:* V: [#: | %$:] ;
```

(5c)"Devocalisation of I when followed by a vowel"

```
     I:y  <=>      [#: | %*:] (:v)  _  [a: | e: | o: | u: |
%}:] ;
                                N:  _  :V ;
                      [#: | %*:] :m  _  :* %`: ;
```

(5d)"Elision of lexical I"

```
  I:0  <=>  [j | k:k | l | m | v]  _  i: ;
                       [l: | n: | z:]  _  %}: ;
                                  N:  _  C: ;
                             [z | l]  _  [a | e | o] ;
```

There is no space to explain what each of the rules stands for. They are combined rules, i.e. more than one context constraint is expressed in each rule. They are not rules for handling individual cases but rather such rules that apply always in similar environments.

4. AMBIGUITY

The interpretation of individual words is often not clear, because the word may have more than one interpretation. In other words, the readings are ambiguous, as we see in (6), where the five words described above are analysed.

(6)

```
"<mwalimu>"
        "mwalimu" N 1/2-SG { teacher }
"<nyumbu>"
        "nyumbu" N 9/10-0-SG { hartebeest } AN
        "nyumbu" N 9/10-0-PL { hartebeest } AN
        "nyumbu" N 9/10-0-SG { mule } AN
        "nyumbu" N 9/10-0-PL { mule } AN
"<mbuzi>"
        "mbuzi" N 9/10-0-SG { goat } AN
        "mbuzi" N 9/10-0-SG { coconut grater }
        "mbuzi" N 9/10-0-PL { goat } AN
        "mbuzi" N 9/10-0-PL { coconut grater }
"<nta>"
        "nta" N 9/10-0-SG { wax }
        "nta" N 9/10-0-PL { wax }
"<taa>"
        "taa" N 5a/6-SG { large flat fish , skate } AN
        "taa" N 9/10-0-SG { lamp , lantern } AR
        "taa" N 9/10-0-SG { discipline , obedience }
        "taa" N 9/10-0-SG { large flat fish , skate } AN
        "taa" N 9/10-0-PL { lamp , lantern } AR
```

"taa" N 9/10-0-PL { discipline , obedience }
"taa" N 9/10-0-PL { large flat fish , skate } AN
"utaa" N 11/10-PL { storage }
"taa" ADJ A-UNINFL { exalted } AR

Only *mwalimu* is unambiguously specified, and all the others have more than one interpretation. The fact that singular and plural forms are the same in class 9/10 causes ambiguity. The other source of ambiguity is the semantic interpretation. The word *nyumbu* means 'hartebeest' and 'mule', which both are animates.

The word *mbuzi* means a 'goat' but also a 'coconut crater'. The word *taa* has nine or more interpretations, depending on whether also each gloss in English should be considered as a separate reading.

It is clear that the kind of reading as in (6) requires that the correct reading is chosen on the basis of the context. This operation is called disambiguation, and we shall discuss it below.

5. DISAMBIGUATION AND SYNTACTIC MAPPING

The morphological analyser alone has only limited use, because the result contains ambiguous readings. Apart from a word-level spelling checker, it is hard to know what it is good for a wider community of the ordinary language users. Therefore, in order to be reliable and useful for a wide variety of applications it needs to be developed further. The immediate step is the disambiguation, i.e. the choice of the correct reading in a given context.

The question of ambiguity is an intricate problem. Although it is true that according to one calculation method Swahili is at least two-ways ambiguous in about half the cases in running text (Hurskainen 1996: 569), this concerns only the basic morphological ambiguity. There are also other types of ambiguity, such as the one caused by various semantic interpretations, ambiguities introduced purposefully to be handled by rules, etc. (Hurskainen 2004b). Therefore, it is not meaningful to state definitely how ambiguous Swahili words are. If some comparison should be made with English, for example, one could say that while English is highly ambiguous in relation to the part-of-speech category, Swahili is not. On the other hand, Swahili is highly ambiguous in relation to noun class affiliation, which in the case of some verb structures leads to more than 100 word forms on the morphological level alone. Also, if all the different interpretations of Swahili words in English are counted as instances of ambiguity, the amount of ambiguity will be multiplied.

Ambiguity in SALAMA is resolved in phases, starting from reliable rule-based resolution and proceeding through feature-based heuristic guessing to less reliable guessing based on probabilities (Hurskainen 2004b).

5.1. Disambiguation based on Constraint Grammar

As already mentioned above, a large part of ambiguity in Swahili derives from the homographic form of some noun class morphemes. This is particularly pronounced in verbs, which often have two or three such morphemes, each of which is two- or three-ways ambiguous. However, although the word may be ambiguous when considered in isolation, with the help of context, keys for disambiguation can often be found. This is possible, because the noun class system is composed of four different sets of noun class morphemes, i.e. there are separate sets for nouns, adjectives, pronouns and subject morphemes of verbs. Although there may be ambiguity within one set, when taking into consideration the whole phrase (noun phrase, verbal phrase etc.), the ambiguity often can be resolved. This feature is extensively used in disambiguation rules. The Constraint Grammar Parser CG-2 (Tapanainen 1996, 1999), which was mostly used for writing disambiguation rules, has two basic operations. One rule type deletes (hence constraint grammar) inappropriate readings. In (7) we have an example, where a deletion rule can be written.

(7)
```
"<*nitachukua>"
        "chukua" V CAP 1/2-SG1-SP VFIN { *i } FUT:ta z [chukua] { take , withdraw , transport , carry } SVO
"<hizi>"
        "hizi" V IMP z [hizi] { disgrace , dishonour , insult , inflict punishment , endanger } SVO AR
        "hizi" V <kwisha z [hizi] { disgrace , dishonour , insult , inflict punishment , endanger } SVO AR
        "hizi" PRON DEM :hV 9/10-PL { these }
"<.$>"
        "." { .$ }
```

The object for the finite verb *chukua* is the pronoun *hizi*, but this word form has also two other interpretations, both being verbs. In practice this form hardly ever appears as a verb and it can be deleted. The rule in (8) removes the verb interpretation.

(8)
```
REMOVE (V)
        IF (0 ("hizi")) ;
```

Note that this is a very local rule and applies to this case only. However, because pronouns are common in language, there is motivation to write such local rules. The same result would have been achieved also by writing a selection rule. In (9) we have an example of this.

(9)
```
SELECT (PRON)
        IF (0 ("hizi")) ;
```

The principle of agreement in Swahili provides ample possibilities to solve even complex ambiguities with the help of selection rules. In (10) we have a sentence that requires disambiguation.

(10)
```
"<*taa>"
        "taa" N CAP 5a/6-SG { large flat fish , skate } AN
        "taa" N CAP 9/10-0-SG { lamp , lantern } AR
        "taa" N CAP 9/10-0-SG { discipline , obedience }
        "taa" N CAP 9/10-0-SG { large flat fish , skate } AN
        "taa" N CAP 9/10-0-PL { lamp , lantern } AR
        "taa" N CAP 9/10-0-PL { discipline , obedience }
        "taa" N CAP 9/10-0-PL { large flat fish , skate } AN
        "utaa" N CAP 11/10-PL { storage }
        "taa" CAP ADJ A-UNINFL { exalted } AR
"<zote>"
        "ote" PRON :ote 9/10-PL { all }
"<mbili>"
        "mbili" NUM NUM-INFL CARD 9/10-PL { two }
"<zilivunjika>"
        "vunjika" V 9/10-PL-SP VFIN { they } PAST z [vunja] { break , damage , annul
} SVO EXT: STAT :EXT
"<.$>"
        "." { .$ }
```

All other words are unambiguous except *taa*, which has nine interpretations. Because the grammatical rule states that the subject prefix governs the subject, we can take this feature as a key for solving ambiguity. Because there is no explicit tag showing that *taa* is a subject, we have to construct a rule that imitates the rule for mapping the subject. The tag VFIN in *zilivunjika* states that this is a finite verb, which is expected to have a subject. The subject in Swahili is normally on the left side of the finite verb, but the distance is undefined. We can approach the problem in two ways. In the method exemplified in (11) we construct a fairly local rule, which is very reliable.

(11)
```
SELECT (N 9/10-0-PL lamp)
        IF (1C (PRON))
          (2C (NUM))
          (3 (V) + (9/10-PL-SP)) ;
```

The rule states that select the plural of the noun with the English gloss lamp, if the next word is a pronoun and the second to the right is a numeral and the third to the right is a verb with the subject prefix of the class 10 (i.e. 9/10-PL-SP). This rule disambiguates the word *taa* completely, but it is fairly local. Note that because of transparency the rule in (11) is written by using the actual tags of the morphological parser. The formalism allows the use of sets, which makes it

possible to write more general rules by using set names. The rule in (11) can be written as shown in (12).

(12)
```
SELECT (N 9/10-0-PL lamp)
            IF (1C PRON)
              (2C NUM)
              (3 V + SP-10) ;
```

But we may attempt writing a still more general rule, where we do not define specifically what there is between *taa* and the verb. The rule in (13) shows how this can be done.

(13)
```
SELECT (N 9/10-0-PL lamp)
            IF (*1 SP-10 BARRIER VFIN)
              (NOT *1 N BARRIER VFIN) ;
```

This rule states that select the plural of the noun with the English gloss lamp, if on the right there is a finite verb with the subject prefix of the class 10, and do not scan further. Also do not allow a noun to appear between the target and the finite verb. A noun is not allowed in between, because it could be the real subject, and in that case disambiguation of *taa* could not be based on the agreement between *taa* and the finite verb.

By applying any of the rules (11-13) we get the result as shown in (14).

(14)
```
"<*taa>"
            "taa" N CAP 9/10-0-PL { lamp , lantern } AR
"<zote>"
            "ote" PRON :ote 9/10-PL { all }
"<mbili>"
            "mbili" NUM NUM-INFL CARD 9/10-PL { two }
"<zilivunjika>"
            "vunjika" V 9/10-PL-SP VFIN { they } PAST z [vunja] { break , damage , annul
} SVO EXT: STAT :EXT
"<.$>"
            "." { .$ }
```

We have seen above that in some cases disambiguation can be based either on removing or selection. It depends on the problem which type of rule is most appropriate. Selection rules have a priority for two reasons. First, they are efficient and do not require other rules for disambiguating the problematic case. However, they can be used only when the danger of misapplication is absent. Second, selection rules are often grammatical rules, the application of which belongs to the good general guidelines of language description. Deletion rules are used only if selection is not possible.

In defining constraints for rule application, there are several means for writing the constraints. In a selection rule (12), for example, the basic constraint

type is to point out the tag, or a set of tags, which must occur in a certain position in relation to the target, so that the rule applies. The rule formalism allows scanning forward and backwards, either by using absolute distances (e.g. two words back), or unlimited scanning (13), which will be terminated by encountering a tag defined by the rule or by encountering a sentence boundary (which is set as a default in the formalism). A constraint may be stated positively or negatively (13), so that in a positive constraint the feature must occur, whereas in the negative constraint the stated feature is not allowed to occur.

Although in principle the rule writing by using linguistic rules should be straight forward, if the writer knows well the grammar of the language, it is not. In practice it is quite complicated and requires careful planning. For example, the grammatical rule says that the subject governs the subject prefix of the finite verb in a sentence. Without syntactic tags there is no indication which of the words is the subject. We know that it is either a noun or a pronoun (or is completely absent) and that it is on the left from the finite verb, but we do not know how far from the verb it is.

A useful strategy in disambiguation is to write the reliable rules first and place them into the first set of constraints. By establishing various sets of constraints it is possible to arrange the rules into the order of reliability, the least reliable rules last.

5.2. Heuristic rules

An unrestricted text is seldom without such words that the morphological parser does not recognize. Those may be real words that are not listed in the lexicon, or they may be typos of real words. The morphological parser should not tolerate typos, and for this reason they have to be treated separately. The next reliable solution is guessing, where features of the word-form are made use of in writing heuristic rules. In Swahili, noun class prefixes of nouns provide such information. For example, if a noun is misspelled in text, its grammatical features can be guessed, although all more refined information will remain obscure. Another case is a word with a capital initial letter elsewhere than in the beginning of a sentence is an indication that is a proper name. However, the use of heuristic guessing is limited and it is seldom fully reliable.

5.3. Guessing on the basis of probability

When the methods described above fail, the last alternative is to try to guess the correct choice by means of probability. On the purely morphological and syntactic level the above methods are practically sufficient. But when we have to make choices in semantic interpretations, we need additional methods that are often not fully reliable.

There are various methods for estimating probabilities. Large text corpora are indispensable for calculating the frequencies of word collocations or other types of co-occurrence of two or more words. The information calculated in this way can then be encoded into data banks. Such data banks can then be used for drawing information for different language applications. One such source of structured information is the WordNet type of database (Fellbaum [ed.]1998, Resnik 1998), where lexical words are organised so that various kinds of are explicitly shown. This information helps in determining the most obvious interpretation in each instance. For Swahili there is no WordNet yet, but it would certainly help a lot in determining the order of frequency between various meanings of a lexeme. Without systematic calculation based on real corpus data man-made estimation is prone to too many errors.

A useful method of finding dependencies between syntactic constituents is to use a Self Organising Map approach (Kohonen 1995), which on the basis of a large text corpus is able to show which constituents are likely to co-occur. On the basis of such clustering it is then possible to name the clusters and enrich the morphological lexicon with those cluster names and use them in writing semantic disambiguation rules.

5.4. Shallow syntactic mapping

CG-2 is an environment for writing rules for disambiguation and for syntactic mapping as well. So in the same environment, and even in the same rule file, rules for both of these operations can be written (Karlsson 1995, Tapanainen 1996). We have to remember, however, that the syntactic description achieved in this way is surface syntax with only a limited degree of depth. For example, the tag @SUBJ> tells that the word is a subject and that on its right side there is the main verb. The tag tells the direction where the main verb is to be found but it does not show its precise place. The same applies also to the specification of other syntactic constituents. They can be named, but their relation to the governing constituents is partial; in other words, it is under-specified. Therefore, the syntactic mapping of this kind is often called shallow syntax, because it does not merely label the constituents, but also indicates, in which direction the head is to be found. The mapping that has only the syntactic labels without indication of the direction of the head is called surface syntax. An example of shallow syntactic mapping is in (15).

(15)

```
"<*mtoto>"
        "mtoto" N CAP 1/2-SG { child , young person , juvenile } @SUBJ>
"<analima>"
        "lima" V 1/2-SG3-SP VFIN { he/she } PR:na z [lima] { cultivate , dig , farm , plough , till , hoe } SVO @FMAINVtr+OBJ>
"<shamba>"
```

```
        "shamba" N 5a/6-SG { farm , field , plot for cultivation , plantation , estate ,
countryside } @OBJ<
"<.$>"
        "." { .$ }
```

The subject tag @SUBJ> and object tag @OBJ<, as well as the finite verb tag @FMAINVtr+OBJ> have arrows to show the direction, where the link should be established, but no explicit link exists.

5.5. Full syntactic mapping

A more advanced step in syntactic mapping is the method, where a sentence is converted into a dependency tree.[4] This means that not only the syntactic label of each word is shown, but also the direction of the head and the distance from it for each constituent is specified. The Functional Dependency Parser (Järvinen and Tapanainen 1997) is such an environment, where dependency trees can be constructed. The system numbers each token of the sentence. The rules are often so written that each rule covers a fairly large number of different cases, while, when scanning, the rule formalism keeps track of the distance from the target and establishes links between the two constituents, i.e. the target and its head. When the system numbers each token, on the basis of the various rule applications the system is able to construct a tree, where the whole network of linked dependencies is shown. Such a mapping system produces a result that can be called deep syntactic analysis, or even full syntactic analysis.

Underneath the system there is a theory, according to which each sentence has a unique head, normally a finite verb, and all other members of the sentence are either directly or indirectly linked to the head. The main features of the Functional Dependency Grammar are drawn from Järvinen and Tapanainen (1997: 5):

"(1) The basic syntactic element is not a word, but a nucleus.

(2) Every element has one and only one head.

(3) The result is a tree.

(4) Functional dependencies are expressed by link names.

(5) The links may cross (non-projective constructions are allowed).

(6) Modifiers are not obligatory; valency defines possibility rather than obligatoriness to have arguments.

(7) The grammar is not generative. This means that the parser accepts every input sentence, and returns an analysis even for ungrammatical sentences to the extent that the structure is recoverable.

(8) Then dependency description is monostratal, i.e. there is one level of syntactic description, the surface-syntactic description, and no transformations."

---

[4] A useful comparison of the Constraint Grammar approach and the Dependency Grammar approach is found in Tapanainen (1999).

The major advantage of the dependency grammar type of parsing is that the syntactic structure is expressed explicitly. In semantic disambiguation, for example, it is possible to make use of the tags used for describing syntactic functions of various constituents. Especially, when there is a need to write rules, where the constraining feature is in an unspecified distance, it is possible to write the rule, if the feature can be labelled in advance.

The dependency type of parsing makes it also possible to identify such multi-word expressions, where the constituents are not necessarily arranged as a continuous sequence of words, i.e. they allow other words in between. This enhances the identification of practically all types of expressions in a sentence, although the exclusion of erroneous linking requires careful formulation of the rules as well as sufficient testing. When the sentence in (15) is analysed with the Dependency Grammar Parser, the result looks as in (16).

```
(16)
 main#1
 *mtoto
              "mtoto"  CAP N 1/2-SG { child , young person , juvenile } @NH >2
 analima
              "lima"  V 1/2-SG3-SP VFIN PR:na z { dig , farm , cultivate , plough , till , hoe }
              SVO @MAIN obj#3 subj#2 >1
 shamba
              "shamba"  N 5a/6-SG { farm , field , plot for cultivation , plantation , estate ,
              country side } @NH >3
.$
```

We see that the syntax is expressed with function labels of two types. First, there are the labels with an @ prefix indicating syntactic properties of individual words without any reference to the structure of the sentence. Then there are the link names which explicitly show to which constituent each of them is linked. The links are marked using ">number" and the heads "#number". For example, subj#2 means that it is the head of the subject, which is in position 2. The tag >2 indicates the link that is connected to the head with the corresponding number. In the same way obj#3 signifies a head for the object, and the object itself is marked with the corresponding tag >3 indicating a link between these two. The verb itself is the head, or main, and it is at the top of the structure; therefore the link label >1.

In (17) we have a sentence with a noun phrase *mwalimu huyu mwema*.

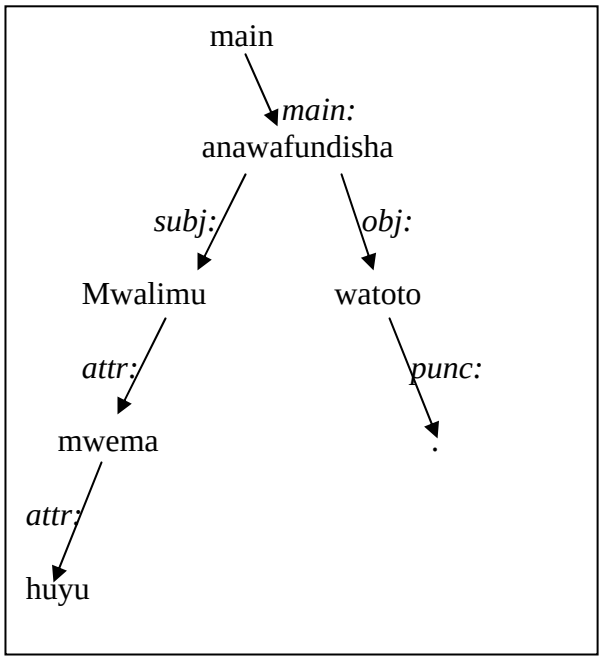
```
(17)
 main#1
*mwalimu
              "mwalimu"  N CAP  1/2-SG { a/the }  { teacher } @NH attr#2 >3
huyu
              "huyu"  PRON DEM :hV  1/2-SG { this } @POSTMOD attr#4 >2
mwema
              "ema"  ADJ A-INFL 1/2-SG  { good } @POSTMOD >4
```

```
anawafundisha
        "fundisha"  V 1/2-SG3-SP VFIN { he/she }  PR:na 1/2-PL3-OBJ OBJ { them }
        z [funda] { teach , instruct , inculcate , indoctrinate } SVO  EXT:  SVO-C CAUS
        :EXT @MAIN obj#5 subj#3 >1
watoto
        "mtoto"  N 1/2-PL { the }  { child , young person , juvenile } @NH >5
.$
        "."  { .$ }
```

In it, *mwalimu* has a head tag attr#2 which is linked to *huyu* (>2). The word *huyu* again is a head (attr#4) to *mwema* (>4). The word *mwema* is not a head to any constituent, and therefore it does not have a head tag. The verb *anawafundisha* is a head to the subject (subj#3) and is linked to it (>3). The verb is a head also to the object (obj#5) and is linked to it (>5). The object *watoto* is not a head and does not have a head tag.

The tree format of the sentence in (17) makes the output easy to read, as shown en (18). The difference between (17) and (18) is merely in the mode of output.

(18)



6. HANDLING MULTI-WORD EXPRESSIONS

In normal text, individual words usually constitute such units that can be handled as such in various phases of processing. Their morphological structure can be displayed and their semantic function, together with the meaning in another language, can be expressed. When proceeding towards the automatic translation of the language, the information needed is available in the disambiguated analysis of the sentence in the form that can be converted into the surface form of the target language as is shown in (19).

(19)
```
"<*mtoto>"
        "mtoto" N CAP 1/2-SG { a/the } { child , young person , juvenile } @SUBJ
"<alimpiga>"
        "piga" V 1/2-SG3-SP VFIN { he/she } PAST 1/2-SG3-OBJ OBJ { him/her } z
        [piga] { hit , beat } SVO @FMAINVtr+OBJ>
"<paka>"
        "paka" N 9/10-0-SG { a/the } { cat } AN @OBJ
"<.$>"
         "." { .$ }
```

On the basis of the information in (19) one can figure out that the sentence means 'The child hit the cat.' It is also possible to convert the sentence into the correct English form automatically, as we shall see later.

Let us take another example (20), where this is not simple any more.

(20)
```
"<*mtoto>"
         "mtoto" N CAP 1/2-SG { a/the } { child , young person , juvenile }
"<alimpigia>"
         "pigia" V 1/2-SG3-SP VFIN { he/she } PAST 1/2-SG3-OBJ OBJ { him/her } z
         [piga] { hit , beat } SVO EXT: APPL :EXT
"<simu>"
         "simu" N 9/10-0-SG { a/the } { telephone , telephone message }
         "simu" N 9/10-0-SG { a/the } { type of sardine or sprat } AN
         "simu" N 9/10-0-PL { the } { telephone , telephone message }
         "simu" N 9/10-0-PL { the } { type of sardine or sprat } AN
         "simu" N 9/10-0-SG { telephone , telephone message }
         "simu" N 9/10-0-SG { type of sardine or sprat } AN
         "simu" N 9/10-0-PL { telephone , telephone message }
         "simu" N 9/10-0-PL { type of sardine or sprat } AN
"<.$>"
         "." { .$ }
```

This exemplifies the problem encountered in processing unrestricted text. The sentence means 'The child called him (i.e. made a telephone call)', but this meaning does not become clear from the meanings of individual words.

Idioms are an example of such constructions where the total meaning cannot be derived on the basis of the meaning of each constituent. To this group belong also various sayings, which are not strictly idioms, and also proverbs.

In order to achieve precise translation of text, these multi-word expressions have to be isolated from the rest of the text and they should be treated as units. An idiom may contain several words, but its semantic meaning in another language can be expressed with one word. An example of a complicated idiom is in (21).

(21)
```
"<*amepaka>"
        "paka" V CAP 1/2-SG3-SP VFIN { he/she } PERF:me z [paka] { smear , spread
        , apply } SV SVO
```

```
"<mafuta>"
        "mafuta" N 6-PL { the } { oil , animal fat , lard }
"<kwa>"
        "kwa" PREP { at/to/for/with }
        "kwa" GEN-CON 15-SG { of }
        "kwa" GEN-CON 17-SG { of }
"<mgongo>"
        "mgongo" N 3/4-SG { a/the } DER:o { back }
"<wa>"
        "wa" GEN-CON 3/4-SG { of }
        "wa" GEN-CON 11-SG { of }
        "wa" GEN-CON 1/2-SG { of }
        "wa" GEN-CON 1/2-PL { of }
        "wa" 1/2-PL-SP
"<chupa>"
        "chupa" V IMP z [chupa] { jump down across , hop , leap , shrink } SV
        "chupa" N 5a/6-SG { a/the } { amniotic membrane & waters }
        "chupa" N 9/10-0-SG { a/the } { bottle }
        "chupa" N 9/10-0-SG { a/the } { amniotic membrane , waters }
        "chupa" N 9/10-0-PL { the } { bottle }
        "chupa" N 9/10-0-PL { the } { amniotic membrane , waters }
"<.$>"
        "." { .$ }
```

This idiom has five constituents, but they express a single idiomatic meaning. In the following I will show how idioms and other multi-word expressions can be handled so that the correct meaning of each expression is found.

6.1. Types of multi-word expressions

Multi-word expressions can be divided roughly into two main groups, those with a fixed form and those with inflecting parts in the expression.

6.1.1 Multi-word expressions with fixed form

An example of a multiword expression is *moja kwa moja* (straight ahead). It occurs only in this form. Other types of multi-word expressions are terms of various domains, where the expression is formed with the help of modifiers. An example of this type of expression is *utata wa mofolojia* (morphological ambiguity). Often the nouns have a singular and plural form, and both forms should be handled separately.

Multi-word expressions of this group can be handled in the morphological lexicon. This requires that the pre-processor (tokeniser) converts them first into single strings as shown in (22).

(22)
moja kwa moja > moja_kwa_moja
utata wa mofolojia > utata_wa_mofolojia

These strings have then to be in the morphological lexicon in the corresponding form as in (23).

(23)
```
moja_kwa_moja ADV "= { straight ahead } ";
utata_wa_mofolojia N 11-SG "= { morphological ambiguity } (ling) ";
```

6.1.2. Multi-word expressions with flexible form

In text there are also such multi-word expressions that can appear in more than one or two forms. The variation may occur in the choice between synonyms, in the choice of alternative writing forms, in the use of punctuation, in the optional use of syntactic arguments (e.g. object) etc. The biggest variation occurs in idioms, where in the typical case there is a finite verb as a key element.

These multi-word expressions must be handled after the morphological analysis has already been done. Below I shall discuss proverbs and idioms separately and describe solution for both of them phase by phase.

6.1.3. Proverbs

Proverbs constitute normally sentences, although often defective, but they can also be found as part of a longer sentence. They have a fairly fixed form but too much variation to be handled in the morphological lexicon. Although the meaning of a proverb can often be figured out on the basis of individual words, it cannot be elegantly translated into another language on this basis. Often there is also a need to express a proverb with the corresponding proverb in another language. For these reasons the correct solution is to isolate the proverb from the rest of text and handle it as a multi-word expression. The proverb *Chema chajiuza, kibaya chajitembeza* is analysed in (24).

(24)
```
"<*chema>"
        "ema" CAP ADJ A-INFL 7/8-SG { good }
"<chajiuza>"
        "uza" V 7/8-SG-SP VFIN { it } PR:a REFL-OBJ OBJ { himself/herself/itself } z
        [uza] { sell } SVO
        "uza" V 7/8-SG-SP VFIN { it } PR:a REFL-OBJ OBJ { himself/herself/itself } z
        [ua] { kill , murder } SVO EXT: SVO-C CAUS:z :EXT
"<,>"
        "," COMMA { , }
"<kibaya>"
        "baya" ADJ A-INFL 7/8-SG { bad , ugly , adverse , unacceptable , awful ,
        sinister , wicked }
```

```
"<chajitembeza>"
        "tembeza" V 7/8-SG-SP VFIN { it } PR:a REFL-OBJ OBJ { himself/herself } z
        [tembea] { walk , move around , travel around , fornicate } SV EXT: SVO-C
        CAUS:z :EXT
"<.$>"
        "." { .$ }
```

The proverb is isolated from text and given an interpretation with the help of Constraint Grammar rules. After applying the first set of rules, the result is s shown in (25).

(25)

```
"<*chema>"
        "ema" CAP ADJ A-INFL 7/8-SG { good }
"<chajiuza>"  S:23122/9
        "uza" V 7/8-SG-SP VFIN { it } PR:a REFL-OBJ OBJ { himself/herself } z [uza]
        { sell } SVO
"<,>"
        "," COMMA { , }
"<kibaya>"
        "baya" ADJ A-INFL 7/8-SG { bad , ugly , adverse , unacceptable , awful ,
        sinister , wicked }
"<chajitembeza>"  S:613/0
        "tembeza"  S:613 <<<<PROVERB { *what is good sells itself , what is bad
        loiters around for sale [*a good wine needs no bush] }
"<.$>"
        "." { .$ }
```

The CG rule for describing the proverb in (25) is shown in (26).

(26)

```
REPLACE (<<<<PROVERB { *what is good sells itself , what is bad loiters around for sale
        [*a good wine needs no bush] } ) TARGET ("tembeza")
        (-4 ("ema") + NCL-7)
        (-3 ("uza") + (REFL-OBJ))
        (-2 (COMMA))
        (-1 ("baya")) ;
```

We see in (25) that the reading of the last member of the proverb has been replaced by the tag <<<<PROVERB and with its description in English. Also a corresponding English proverb is given.

In the second phase, the tags of the other members of the proverb are removed and their place in the structure is described with angle brackets, as shown in (27). This is done with rules that make use of the structure of the proverb found in the last member of the proverb in (25).

(27)
```
"<*chema>"
        "ema" PROVERB>>>>
"<chajiuza>"
        "uza" PROVERB<>>>
"<,>"
        "," PROVERB<<>>
"<kibaya>"
        "baya" PROVERB<<<>
"<chajitembeza>"
        "tembeza" <<<<PROVERB { *what is good sells itself , what is bad loiters
        around for sale [*a good wine needs no bush] }
"<.$>"
        "." { .$ }
```

It is now easy to process the format in (27) further as required in language translation. The final form is shown in (28).

(28)
Chema chajiuza, kibaya chajitembeza { What is good sells itself, what is bad loiters around for sale [A good wine needs no bush] }

There are also proverbs with alternating forms, as is shown in (29).

(29)
Baada ya dhiki faragha.
Baada ya dhiki faraja.
Baada ya dhiki faraji.

These can be described with one rule as shown in (30).

(30)
```
REPLACE (<<PROVERB { *after trouble there is relief } ) TARGET ("faragha") OR
        ("faraja") OR ("faraji")
        (-2 ("baada_ya"))
        (-1 ("dhiki")) ;
```

Text types differ in regard to the frequency in using proverbs. A total of almost 2,000 proverbs[5] have been included into SALAMA, and it is estimated that there are hardly such proverbs missing that are in common use.

6.1.4. Idioms

For isolating idioms the approach is partly the same as with proverbs. The major difference here is that we have to retain the tags of the verb, except the gloss that is obsolete. In (20) we had an example of an idiom after morphological parsing,

[5] Proverbs were collected primarily from dictionaries and from the excellent collection of Wamitila (1999).

Mtoto alimpigia simu. When a rule for isolating the idiom in that sentence is applied, we get the result shown in (31).

(31)
```
"<*mtoto>"
        "mtoto" N CAP 1/2-SG { a/the } { child , young person , juvenile }
"<alimpigia>"
        "pigia" V 1/2-SG3-SP VFIN { he/she } PAST 1/2-SG3-OBJ OBJ { him/her } z
        [piga] { hit , beat } SVO EXT: APPL :EXT
"<simu>"
        "simu"  <IDIOM { call }
"<.$>"
        "." { .$ }
```

We see that *simu* is marked as the last member of an idiom and that the previous word also is part of the idiom. The meaning 'call' is attached to the last member.

In the second phase the output is as shown in (32).

(32)
```
"<*mtoto>"
        "mtoto" N CAP 1/2-SG { a/the } { child , young person , juvenile } @SUBJ
"<alimpigia>"
        "pigia" V 1/2-SG3-SP VFIN { he/she } PAST 1/2-SG3-OBJ OBJ { him/her } z
        [piga] SVO EXT: APPL :EXT  IDIOM-V> @FMAINVtr-OBJ>
"<simu>"
        "simu" <IDIOM { call }
"<.$>"
        "." { .$ }
```

The result shows that the verb has lost its original gloss and a tag IDIOM-V> has been added to show that it is part of the idiom, that it is a verb, and that the word to the right is the other member of the idiom. Because the gloss of the idiom is attached to the second member and not to the verb, and because the morphological information is attached to the verb, these two words have to be kept together also in the later processing. This is shown in (33) by putting the constituents into brackets and by reordering tags.

(33)
```
( N CAP 1/2-SG { a/the } { child , young person , juvenile } @SUBJ )
( V 1/2-SG3-SP VFIN  PAST 1/2-SG3-OBJ { call } [piga] SVO EXT: APPL :EXT IDIOM-
V> @FMAINVtr-OBJ> <IDIOM OBJ { him/her }  )
( { .$ } )
```

Now when we further process the sentence we get a translation in English (34).

(34)
A/the child called him/her.

We had a more complicated idiom in (21), *Amepaka mafuta kwa mgongo wa chupa*. We change it now a bit and add a proper subject in order to see that the

whole idiom becomes isolated. After the first phase of processing after morphological analysis the result is as in (35).

(35)
```
"<*mfanyakazi>"
        "mfanyakazi" N CAP 1/2-SG { a/the } DER:zi { worker , employee } @SUBJ
"<amepaka>"
        "paka" V 1/2-SG3-SP VFIN { he/she } PERF:me z [paka] { smear , spread ,
        apply } SV SVO
"<mafuta>"
        "mafuta" N 6-PL { the } { oil , animal fat , lard }
        "mafuta" N 6-PL { oil , animal fat , lard }
"<kwa>"
        "kwa" PREP { at/to/for/with }
"<mgongo>"
        "mgongo" N 3/4-SG { a/the } DER:o { back }
        "mgongo" N 3/4-SG DER:o { back }
"<wa>"
        "wa" GEN-CON 3/4-SG { of }
"<chupa>"
        "chupa"  S:14594 <<<<<IDIOM { flatter , praise falsely }
"<.$>"
        "." { .$ }
```

The meaning of the idiom is attached to the last member *chupa*. In the second phase, the other members of the idiom are marked (36).

(36)
```
"<*mfanyakazi>"
        "mfanyakazi" N CAP 1/2-SG { a/the } DER:zi { worker , employee } @SUBJ
"<amepaka>"
        "paka" V 1/2-SG3-SP VFIN { he/she } PERF:me z [paka] SV SVO  IDIOM-
        V>>>>> @FMAINVtr+OBJ>
"<mafuta>"
        "mafuta" IDIOM<>>>>
"<kwa>"
        "kwa" IDIOM<<>>>
"<mgongo>"
        "mgongo" IDIOM<<<>>
"<wa>"
        "wa" IDIOM<<<<>
"<chupa>"
        "chupa" <<<<<IDIOM { flatter , praise falsely }
"<.$>"
        "." { .$ }
```

The result in (37) shows that the idiom is really isolated.

(37)
```
( N CAP 1/2-SG { a/the } DER:zi { worker , employee } @SUBJ )
```

(V 1/2-SG3-SP VFIN PERF:me [paka] SV SVO IDIOM-V>>>>> @FMAINVtr+OBJ> <<<<<IDIOM { flatter , praise falsely })
({ .$ })

The last phase of processing is shown in (38).

(38)
A/the worker has flattered.

I have in brief described the types of problems we find in processing multiword expressions and shown how they can be handled elegantly. Proverbs are less problematic in normal text, because they are not very frequent in it. There are writers, however, who use them extensively, and also in forms that make them difficult to process with normal processing methods. In the current implementation of SALAMA there are about 2,000 proverbs described in the system.

Idioms are quite frequent in text, and without their proper handling automatic translation would not be possible. Idioms also belong to all types of text. The large majority of idioms has a verb as a key constituent and this makes it hard to handle them. Some verbs occur in a large number of idioms. The verb *piga* occurs in more than 250 idioms. More than 2,000 idioms [6] have so far been described in SALAMA.

7. SEMANTIC DISAMBIGUATION

How to determine the correct semantic meaning of a word in context is among the most difficult tasks in language processing, and a combination of strategies is needed for achieving satisfactory results (Wilks and Stevenson 1998; Stevenson and Wilks 2001). Handling of idioms, sayings and proverbs, discussed above, is part of this difficult problem. On a more general level, it is the question of deciding between synonyms, near-synonyms, and other glosses which cannot be readily categorised. Results obtained by such methods as the Self Organising Map (Kohonen 1995; Ng'ang'a 2003) and WordNet (Resnik 1998) are likely to improve the performance of the system.

[6] The idioms were collected from Swahili dictionaries, and especially from the sources compiled by Chuwa (1995) and Wamitila (2000).

8. APPLICATIONS OF SALAMA

SALAMA should be understood as an environment for developing various applications for language manipulation.[7] Below I shall discuss briefly some of the applications that are either already on the market or soon will be available.

8.1. Spelling checking and hyphenation

Among the first applications of a system such as SALAMA is the facility for checking whether the language written is correct. A word-level spell checker and hyphenator for Swahili was released in 2000 by Lingsoft with the name Orthografix 2 for Swahili. It is fully compatible with MS Word and works in Windows 95 and later versions of Windows.[8] The program also hyphenates Swahili text according to the rules designed especially for Swahili.

8.2. Testing dictionaries

SWATWOL can be tailored for testing existing dictionaries of Swahili. The SWATWOL morphological lexicon is made to mirror the dictionary, so that only that information is included in the lexicon which is in the printed dictionary. When such version of SWATWOL is used for analysing various texts, deficiencies of the dictionary become apparent in detail. A total of five Swahili dictionaries have already been tested with this method (Hurskainen 1994, 2002, 2004a).

8.3. Compilation of dictionaries and glossaries

We have seen above that the analysed text has, in addition to the linguistic information, also glosses in English. Some of the words have several alternative glosses, while other words have been divided into several entries according to their different glosses, which are not synonyms. There are also glosses that are needed in translation, such as articles and pronouns, but which in dictionary compilation are unnecessary. Yet the master lexicon contains much such information that can be effectively used in dictionary compilation (Hurskainen 2003).[9] I will discuss them and their implementation briefly below.

---

[7] A heated discussion has been going on concerning the promotion of Swahili in various domains and its use in computer technology (Ashford 2001; Qorro 2001; Kiputiputi 2001)

[8] More about Orthofrafix 2 for Swahili in: www.lingsoft.fi/orthografix/

[9] Prinsloo and de Schryver (2001) have discussed the issues of dictionary compilation without access to a morphological analyser.

8.3.1. Frequency lists

A modern dictionary is often designed for a certain user group. Ideally a dictionary should cover the vocabulary which the user needs. A useful method for ensuring this is to collect all the words which are found in the written and spoken materials of the user group. The dictionary designed for Primary School use, for example, should include the words used in teaching materials, including also the teaching manuals. If these materials are available in computer form, as they increasingly are, it is a simple and quick task to collect them as a text corpus and produce a frequency list with SALAMA.[10]

To produce a frequency list of words from a text is a trivial task, but it has very little use in a language like Swahili, where affixes are primarily prefixed to the words. A reliable list can be produced only with a parser that analyses text, gives the lemma of words, and makes a proper disambiguation on the basis of the context. A system like SALAMA is able to do this. An additional useful feature of SALAMA is that the user can also retrieve idioms from text with correct interpretation. The top part of a frequency list of Swahili produced with SALAMA is in (39)

(39)

324120	na CC { and }
152569	kwa PREP { at, to, for }
135749	wa V { be }
135671	katika PREP { in, at }
122973	sema V { say, speak, scold, speak against, advise, counsel, backbite, badmouth }
96886	la ADV { no, not } AR
63017	na PREP { with }
59806	nchi N 9/10 { country, land, ground }
57447	na AG-PART { by }
46636	mwaka N 3/4 { year }
44812	mtu N 1/2 { human being, person, individual }
38009	kama ADV { like, such as, if, in case } AR
37896	serikali N 9/10 { government } PERS
32222	ingine ADJ A-INFL { other }
31889	fanya V { do, act, commit, make, manufacture, manipulate }
31694	wakati N 11/10 { time, period of time, point of time, season, opportunity } AR
30596	baada ya PREP { after }
29991	chama N 7/8 { party, club }
28123	toa V { put out, remove, publish, produce, subtract, reduce }
26104	taka V { want, wish }
25840	kuu ADJ A-INFL { great, important, eminent, main, major, chief }
24773	mji N 3/4 { town, homestead }
24747	pia ADV { also, likewise, too }
24462	weza V { be able }

[10] A frequency list of Swahili was used as a basis for selecting words for Swahili-Suomi-Swahili -sanakira (Abdulla et al 2002).

8.3.2. Covering vocabularies

With SALAMA it is easy to produce covering vocabularies, i.e. vocabularies, where every word in text has been given an interpretation according to the context. A small extract is given in (40).

(40)
amani N 9/10 { peace } AR
andaliwa V [andaa] { cater, provide, prepare, brew, put in order } PASS
andamano N 5a/6 (andamana) < (andama) { demonstration, procession }
angamiza V [angamiza] { destroy, wreck }
anza V [anza] { begin, establish } APPL
bila PREP { without } AR
binadamu N 9/10 { human being, person } HUM
chuo N 7/8 { high school }
dai N 5a/6 { assertion, allegation, plaint, postulant }
dhehebu N 5a/6 { religious denomination } AR
dini N 9/10 { religion, spiritual belief } AR
fanya V [fanya] { do, act, commit, make, manufacture, manipulate }
fuatwa V [fuata] { follow, come after, pursue, imitate, be with } PASS
furika V [fura] { swell, bulge, swell in anger, be enraged } STAT
haki N 9/10 { justice, right, prerogative, ownership } AR
halaiki N 9/10 { abundance, crowd, troops (performing a display), gathering } AR
ishia V [isha] { finish, come to end } APPL
jangwa N 5a/6 { desert, wilderness, waste, barren ground }
jeshi N 5a/6 { army, force, troop, paratroops, great company, assemblage }
kada N 5a/6 { category } ENG
kama ADV { like, such as, if, in case } AR
katika PREP { in, at }

8.3.3. Domain-specific vocabularies

By using SALAMA, vocabularies can be compiled also for specific domains. Two methods, or their combination, can be used in this. In one method, developed by Sewangi (2000, 2001), term candidates are manually marked into the TWOL lexicon, and a special program is run for tracing the full terms, which often are multi-word concepts. In another method, terms are already included and marked in the TWOL lexicon, and they can, when the text is analysed, be identified by a specific code.

8.3.4. User-defined vocabularies

It is also easy to produce vocabularies, where the user defines the coverage according to need. If 1,000 most common Swahili words are cut out, the vocabulary in (40) will look as shown in (41).

(41)
andamano N 5a/6 (andamana) < (andama) { demonstration, procession }
angamiza V [angamiza] { destroy, wreck }
binadamu N 9/10 { human being, person } HUM
dhehebu N 5a/6 { religious denomination } AR
fuatwa V [fuata] { follow, come after, pursue, imitate, be with } PASS
furika V [fura] { swell, bulge, swell in anger, be enraged } STAT
halaiki N 9/10 { abundance, crowd, troops (performing a display), gathering } AR
jangwa N 5a/6 { desert, wilderness, waste, barren ground }
kada N 5a/6 { category } ENG

If 6,000 most frequent words are deleted, there is only one word left, as shown in (41).

(42)
"halaiki N 9/10 { abundance, crowd, troops (performing a display), gathering } AR

8.3.5. Rough 'translation'

The analysis result can be modified also in such a way that it serves as a rough 'translation'. Unnecessary tags are removed and only those needed in helping to understand the meaning of words remain. This version is shown in (43)

(43)

Tulishangaa	V [shangaa] { be amazed, be surprised, be dumbfounded (with wonder or horror) }
kuona	V [ona] { see, feel, oppress }
wenzetu	N 1/2 { our countryman }
wa	GEN-CON 1/2 { of }
Yanga	PROPNAME { Yanga }
wakikataa	V [kataa] { refuse, decline, disagree, disavow, reject, deny, renounce } AR
kucheza	V [cheza] { play }
na	PREP { with }
Simba	PROPNAME { Simba }
wakati	N 11/10 { time, period of time, point of time, season, opportunity } AR
hakukuwepo	V [wa] { be }
na	PREP { with }
tatizo	N 5a/6 (tatiza) { problem, difficulty } AR
lolote	PRON INDEF 5/6 { any }
.	{. }

8.4. Towards translation into English

The SWATWOL lexicon has been designed so that glosses in English have been provided for all entries. This is to ensure that, when text is analysed, all elements needed are there. The remaining task is to convert the text to meet the grammatical, and also stylistic, requirements of the target language. This involves various kinds of operations.

8.4.1. Conversion of phrase structures

The structure of phrases, especially of noun phrases, is quite different in Swahili and English. While in English the modifiers are normally before the noun head, in Swahili they are usually after the noun. Also the mutual order of modifiers is different. In (44) we have an example of this. The order in English is shown by numbers in front of each word.

(44)
```
5"<*watoto>"
        "mtoto" N CAP 1/2-PL { the } { child , young person , juvenile } @SUBJ
2"<wangu>"
        "angu" PRON POSS 1/2-PL SG1 { my , mine } @<NGEN
3"<wawili>"
        "wili" NUM 1/2-PL NUM-INFL CARD { two }
1"<hawa>"
        "hawa" PRON DEM :hV 1/2-PL { these }
4"<wazuri>"
        "zuri" ADJ A-INFL 1/2-PL { good , beautiful , pretty , gorgeous }
6"<wanacheza>"
        "cheza" V 1/2-PL3-SP VFIN { they } PR:na z [cheza] { play } SV SVO
        @FMAINVtr-OBJ>
"<.$>"
        "." { .$ }
```

The Swahili words can be also in another order, as shown in (45) and (46).

(45)
```
5"<*watoto>"
        "mtoto" N CAP 1/2-PL { the } { child , young person , juvenile } @SUBJ
2"<wangu>"
        "angu" PRON POSS 1/2-PL SG1 { my , mine } @<NGEN
1"<hawa>"
        "hawa" PRON DEM :hV 1/2-PL { these }
3"<wawili>"
        "wili" NUM 1/2-PL NUM-INFL CARD { two }
4"<wazuri>"
        "zuri" ADJ A-INFL 1/2-PL { good , beautiful , pretty , gorgeous }
6"<wanacheza>"
        "cheza" V 1/2-PL3-SP VFIN { they } PR:na z [cheza] { play } SV SVO
        @FMAINVtr-OBJ>

"<.$>"
        "." { .$ }
```

(46)
```
1"<*hawa>"
        "hawa" CAP PRON DEM :hV 1/2-PL { these } @NDEM>
5"<watoto>"
```

```
        "mtoto" N 1/2-PL { the } { child , young person , juvenile } @SUBJ
2"<wangu>"
        "angu" PRON POSS 1/2-PL SG1 { my , mine } @<NGEN
3"<wawili>"
        "wili" NUM 1/2-PL NUM-INFL CARD { two }
4"<wazuri>"
        "zuri" ADJ A-INFL 1/2-PL { good , beautiful , pretty , gorgeous } @<NADJ
        @<NADJ
6"<wanacheza>"
        "cheza" V 1/2-PL3-SP VFIN { they } PR:na z [cheza] { play } SV SVO
        @FMAINVtr-OBJ>
"<.$>"
        "." { .$ }
```

In this implementation, constituent reordering rules were written with Perl.

8.4.2. Constituent reordering within verbs

Inflected Swahili verbs have morphemes that in English are expressed with separate words. Also the order of these morphemes does not correspond to the order in English. The example in (47) illustrates this.

(47)

```
"<mtoto>"
        "mtoto" N 1/2-SG { a/the } { child , young person , juvenile } @SUBJ
"<anayekisoma>"
        "soma" V 1/2-SG3-SP VFIN { he/she } PR:na 1/2-SG-REL { who } 7/8-SG-
        OBJ OBJ { it } z [soma] { study , read , receive teaching , attend school } SVO
        @FMAINVtr-OBJ>
```

The object prefix gloss 'it' should be after the verb. Also the subject prefix gloss 'a/the' should be deleted if the explicit subject is there, as is the case here. Also these cases were handled with Perl.

8.4.3. Handling verb forms

The production of the correct verb form from each verb was a major task in SALAMA, because each verb in the morphological lexicon is represented only as an infinite form. On the basis of this form and the linguistic information attached to the verb reading it is possible to rewrite the verb into the form needed. The implementation of this module was done with BETA. In (48) and (49) we have an example of how verb forms are produced.

(48)
(N 1/2-SG { a/the } { child , young person , juvenile } @SUBJ)
(V NEG SG3-SP VFIN PERF-NEG:ja [soma] { study , read , receive teaching , attend school } SVO @FMAINVtr+OBJ>)
(N 7/8-SG { a/the } { book } @OBJ)

(49)
A/the child has not studied a/the book.

8.4.4. The plural forms of nouns

The plural forms of nouns are formed in the same way as the English verb forms, i.e. on the basis of the lexical singular form and the grammatical information of the analysis. These rewriting rules were written with BETA.

The proper handling of the article in English is a major problem and requires, in addition to rule-based methods, further studies for finding reliable solutions.

9. CONCLUSION

In this paper I have described the main features of the Swahili Language Manager, and also some of the application that can be derived from the system. The central idea in SALAMA is that it is a kind of storehouse of written Swahili. In it, the language is described as fully as possible.

The morphological lexicon is the central, although not the only, component for storing information. It is comprehensive in the sense that, in addition to the general lexicon, it contains concepts also from a variety of specific domains. By marking these domains with special tags it is possible to derive domain-specific lexicons from the master lexicon. Also the tags needed in translation can be included or excluded according to need.

Part of the lexicon, especially idioms, sayings and proverbs, are handled and described with the Constraint Grammar parser. The same parser is used for basic disambiguation and syntactic mapping. The Dependency Grammar parser is used for achieving deep syntactic analysis.

When the analysis part of SALAMA is performed, the information is used for converting the result into written English. This is done without making explicit use of an English parser. The reordering of constituents to meet the needs of English was implemented mostly with rules written with Perl. The production of correct word forms of English, such as verbs and nouns, for example, is another major task. This was implemented with Beta rewriting rules. The management of the English article, as well as the gender in pronouns, is not easy, and it is not yet implemented satisfactorily.

The biggest remaining problem is, however, the semantic disambiguation of such words, which have such shades of meaning that are not clear enough for

handling with explicit general rules. The results found with SOM (Self Organising Map) and Bayesian Networks are expected to enhance significantly the performance of SALAMA in translation.

REFERENCES

Abdulla, A., Halme, R., Harjula, L. and Pesari-Pajunen, M. 2002.

Swahili - Suomi - Swahili -sanakirja. Helsinki: Suomalaisen Kirjallisuuden Seura.

Ashford, R. 2001.

Technology and the Potential Spread of Kiswahili. In Mdee, J.S. and Mwansoko, H.J.M., *Makala ya kongamano la kimataifa KISWAHILI 2000 Proceedings.* Dar-es-Salaam: Taasisi ya Uchunguzi wa Kiswahili, pp 144-157.

Beesley, K, and Karttunen, L., 2003.

Finate State Morphology. Series: CSLI Studies in Computational Linguistics. Stanford. CA.

Chuwa, A. 1995.

Phraseological Units and Dictionary: The Case of Swahili Language. Ph.D. diss. University of Warsaw.

Fellbaum, C. (Ed.) 1998.

WordNet: An electronic lexical database. MIT Press.

Hurskainen A. 1992.

A Two-Level Computer Formalism for the Analysis of Bantu Morphology. An Application to Swahili. *Nordic Journal of African Studies* 1(1): 87-122.

Hurskainen A. 1994.

Kamusi ya Kiswahili Sanifu in test: A computer system for analyzing dictionaries and for retrieving lexical data. *Afrikanistische Arbeitspapiere* 37: 169-179.

Hurskainen A. 1995.

Computer Archives of Swahili Language and Folklore: General Description. In A. Hurskainen na S.A.K. Mlacha (eds.), *Lugha, Utamaduni na Fasihi Simulizi ya Kiswahili.* Dar-es-Salaam: Taasisi ya Uchunguzi wa Kiswahili (Dar-es-Salaam) na Idara ya Taaluma za Asia na Afrika (Helsinki). Pp. 1-15.

Hurskainen A. 1996.

Disambiguation of morphological analysis in Bantu languages. In: *Proceedings of COLING-96*, pp. 568-573.

Hurskainen, A. 1999.

SALAMA: Swahili language manager. *Nordic Journal of African Studies*, 8(2): 139-157. Available also in: www.njas.helsinki.fi

Hurskainen A. 2002.

Tathmini ya Kamusi Tano ya Kiswahili (Computer Evaluation of Five Swahili Dictionaries). *Nordic Journal of African Studies* 11(2): 283-300. Available also in: www.njas.helsinki.fi

Hurskainen A. 2003.

New Approaches in Corpus-Based Computational Lexicography. *Lexikos* 13 (AFRILEX-reeks/series 13: 2003): 111-132.

Hurskainen, A. 2004a.

Computational testing of five Swahili dictionaries. *Proceedings of the 20th Scandinavian Conference of Linguistics, Helsinki, 7-9.1. 2004*. http://www.ling.helsinki.fi/kielitiede/20scl/proceedings.shtml

Hurskainen A. 2004b.

Optimizing Disambiguation in Swahili. In *Proceedings of COLING-04, The 20th International Conference on Computational Linguistics*, Geneva 23-27.8. 2004. Pp. 254-260.

Järvinen, T. and Tapanainen, P. 1997.

A Dependency Parser for English. Technical Reports, No. TR-1. Department of General Linguistics. University of Helsinki.

Karlsson, F. 1995.

Designing a parser for unrestricted text. Karlsson, F. et al (Eds.), *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*: 1-40. Berlin: Mouton de Gryuter.

Karttunen, L., Koskenniemi, K. and Kaplan, R. M. 1987.

A compiler for two-level phonological rules. In Dalrymple, M., Kaplan, R., Karttunen, L., Koskenniemi, K., Sjaio, S., and Wescoat, M. (eds.), *Tools for Morphological Analysis*, vol. 87 - 108 of *CSLI Reports*, pp. 1 - 61. Center for the Study of Language and Information, Stanford University, Palo Alto, CA.

Karttunen, L. and Beesley, K. 1992.

Two-level rule compiler. Technical Report ISTL-92-2. Xerox Palo Alto Research Center. Palo Alto, CA.

Kiputiputi, O.M. 2001.

Kufundisha sayansi kwa Kiswahili. In Mdee, J.S. and Mwansoko, H.J.M., *Makala ya kongamano la kimataifa KISWAHILI 2000 Proceedings.* Dar-es-Salaam: Taasisi ya Uchunguzi wa Kiswahili, pp. 350-376.

Kohonen, T. 1995.

Self-Organizing Maps. Berlin: Springer.

Koskenniemi, K. 1983.

Two-level morphology: A general computational model for word-form recognition and production. Publications No.11. Department of General Linguistics, University of Helsinki

Ng'ang'a, J. 2003.

Semantic Analysis of Kiswahili Words Using the Self Organizing Map. Nordic Journal of African Studies, 12(3): 407-425. Available also in: www.njas.helsinki.fi

Prinsloo, D.J. and de Schryver G.-M. 2001.

Taking Dictionaries for Bantu Languages into the New Millennium with Special Reference to Kiswahili, Sepedi and Isizulu. In Mdee, J.S. and Mwansoko, H.J.M., *Makala ya kongamano la kimataifa KISWAHILI 2000 Proceedings.* Dar-es-Salaam: Taasisi ya Uchunguzi wa Kiswahili, pp 188-215.

Qorro, M.A.S. 2001.

Mjadala wa lugha ya kufundishia: Sababu zinazotolewa kupinga matumizi ya Kiswahiili kama lugha ya kufundishia. In Mdee, J.S. and Mwansoko, H.J.M., *Makala ya kongamano la kimataifa KISWAHILI 2000 Proceedings.* Dar-es-Salaam: Taasisi ya Uchunguzi wa Kiswahili, pp 332-349.

Resnik, P. 1998.

WordNet and class-based probabilities. In: Fellbaum (Ed.), *WordNet: An electronic lexical database*. MIT Press, pp. 239-263.

Sewangi, S. 2000.

Tapping the neglected resource in Kiswahili terminology: Automatic compilation of the domain-specific terms from corpus. *Nordic Journal of African Studies* 9(2): 60-84. Available also in: www.njas.helsinki.fi

Sewangi, S. 2001.

Computer-Assisted Extraction of Terms in Specific Domains: The Case of Swahili. Ph.D. thesis. Publications of the Institute for Asian and African Studies, 1. University of Helsinki.

Stevenson, M. and Wilks, Y. 2001.

The interaction of knowledge sources in word sense disambiguation. *Computational Linguistics* 27(3): 321-349.

Tapanainen, P. 1996.

The Constraint Grammar Parser CG-2. Publications No. 27. Department of General Linguistics, University of Helsinki.

Tapanainen, P. 1999.

Parsing in two frameworks: finite-state and functional dependency grammar. Ph.D. thesis, Department of General Linguistics, University of Helsinki.

Wamitila, K.W. 1999.

Kamusi ya Misemo na Nahau. Nairobi: Longhorn Publishers.

Wamitila, K.W. 2001.

Kamusi ya Methali. Nairobi: Longhorn Publishers.

Wilks, Y. and Stevenson, M. 1998.

Word sense disambiguation using optimised combinations of knowledge sources. In: Proceedings of the 17th International Conference on Computational Linguistics and the 36th Annual Meeting of the Association for Computational Linguistics, pp. 1398-1402.