A FINITE-STATE APPROACH TO LINGUISTIC CONSTRAINTS IN ZULU MORPHOLOGICAL ANALYSIS

Sonja E. Bosch & Laurette Pretorius

ABSTRACT

Conceptually, finite-state approaches are well suited to the modelling and implementation of regular linguistic (in this case morphophonological) behaviour. However, it is common knowledge that, while morphophonological phenomena are mainly regular, the development of large coverage morphological analysers for real natural languages also requires the accurate modelling of idiosyncratic behaviour.

In this paper it is firstly shown how linguistic constraints in an agglutinative language such as Zulu, can be handled in a Xerox style finite-state computational morphological analysis by means of so-called *flag diacritics*. Secondly, the use of *multiple levels of finite-state transducers* for the analysis/generation of certain forms of idiosyncratic behaviour is discussed within the context of the development of a large coverage morphological analyser for Zulu that is both efficient and accurate.

1. INTRODUCTION

Zulu is classified under the South-Eastern zone of the Bantu language family and, as one of the eleven official languages of South Africa, is spoken by approximately 9 million mother-tongue speakers. In terms of natural language processing, particularly computational morphology, the Bantu languages including Zulu certainly belong to the lesser-studied languages of the world. The only Bantu language for which a computational morphological analyser has been fully developed so far is Swahili (Hurskainen 1992).

As a morphologically complex language which is predominantly agglutinating in nature, Zulu contains a variety of linguistic constraints, such as long distance dependencies, that is morphological forms which are dependent on other morphological forms elsewhere in the word, often at the extreme ends of words; circumfixes or co-ordinated pairs consisting of a prefix and a suffix; or forward-looking feature requirements where the presence of one morpheme requires the presence of another morpheme. Various forms of idiosyncratic behaviour are also observed, for instance regarding the restriction of combinations of noun prefixes with verb roots in the formation of deverbative nouns.

The aim of this paper is to show how such linguistic constraints can be handled in a Xerox style finite-state computational morphological analysis of Zulu, with particular emphasis on aspects such as overgeneration and the size of the finite-state networks involved. Section 2 gives a concise overview of the relevant Zulu morphology and introduces various common linguistic constraints in Zulu morphology, while Section 3 contains a brief discussion of the computational approach followed, using the Xerox finite-state toolkit with its so-called flag diacritics as a useful feature-setting and feature-unification device. Sections 4 and 5 constitute the main parts of this paper and address the computational modelling and implementation of the mentioned linguistic constraints (section 4) and certain instances of idiosyncratic behaviour in Zulu morphology (section 5). The phenomena discussed in section 4 differ from those in section 5 in that the former exhibit regular behaviour and may be described by means of linguistic rules while the latter are of an *ad hoc* nature and require a somewhat different approach.

2. TYPICAL CONSTRAINTS IN ZULU MORPHOLOGY

The definition of *morphology* as 'the study of the grammatical structure of words and the categories realized by them' (Matthews 1997: 233), suggests that morphology is mainly governed by regular well-defined processes or rules. However, natural language is also characterised by deviations from such rules. These deviations often manifest as *linguistic constraints*. Different sources define the concept of *constraints* as follows:

> Any statement, in some particular framework or description, which prohibits some derivation, process, structure or combination of elements which would otherwise be allowed. (Trask 1996: 89)
>
> A term used in LINGUISTICS ... to refer to a CONDITION which restricts the application of a RULE, to ensure that the sentences generated are WELL FORMED. (Crystal 1997: 85)
>
> Any restriction either on the application of a rule or process or on the well-formedness of a representation. (Matthews 1997: 318)

Representation in the last definition refers to a structure assigned to a form at any (here morphological) level of description or analysis.

From these definitions it is clear that two aspects need explication. Firstly, the regular behaviour or *rules* need to be *described* or explained and secondly, the *prohibitions* or *restrictions* that are observed in the application of these rules need to be *discussed.* The remainder of this section consists of a discussion of aspects of Zulu morphology pertaining to particular word formation rules and to the associated morphological constraints that may occur in their application. Various rules are described and illustrated by means of examples. The section is concluded with a list of typical constraints in Zulu morphology that form the basis for the remainder of the paper.

The emphasis in this discussion on aspects of Zulu morphology is on the two basic morphological systems which characterise the morphological structure of Zulu, namely the noun classification system, and the ensuing system of concordial agreement.

The *noun classification system* categorises nouns into a number of noun classes, as determined by prefixal morphemes also known as noun prefixes. These noun prefixes have, for ease of analysis, been divided into classes with numbers by scholars who have worked within the field of the Bantu language family. The following are examples of Meinhof's (1932:48) numbering system of some of the noun class prefixes:

umu-	class 1	aba-	class 2	umuntu/abantu	'person/persons'
u-	class 1a	o-	class 2a	udokotela/odokotela	'doctor/doctors'
umu-	class 3	imi-	class 4	umuthi/imithi	'tree/trees'
isi-	class 7	izi-	class 8	isitsha/izitsha	'dish/dishes'
u(lu)-	class 11	izin-	class 10	uthi/izinti	'stick/sticks'
u(bu)-	class 14			ubuntu	'humanity'

In general, noun prefixes indicate number, with the uneven class numbers designating singular and the corresponding even class numbers designating plural. However, this is not always the case, since some nouns in so-called plural classes do not have a singular form; class 11 nouns take their plurals in class 10, while a class such as 14 is not associated with number.

The noun prefix typically constitutes two parts, namely a preprefix (the initial vowel) and a basic prefix. This division is significant for the analysis in the sense that some classes such as 1a and its plural class 2a do not have a basic prefix at all. In other instances such as classes 11 and 14 the basic prefixes are often discarded, with the result that only the preprefix appears in the surface form. Nevertheless, the embedded basic prefix needs to be recognized by a morphological analyser since the whole concordial system of the grammar is based on the noun prefixes.

The significance of noun prefixes is not limited to the role they play in indicating the classes to which the different nouns belong. In fact, noun prefixes play a further important role in the morphological structure of Zulu in that they link the noun to other words in the sentence. This linking is manifested by a system of concordial agreement, which is the pivotal constituent of the whole sentence structure of the Zulu language, and governs grammatical correlation in verbs, adjectives, possessives, pronouns and so forth. The concordial morphemes are derived from the noun prefixes and usually bear a close resemblance to the noun prefixes, as illustrated by the bold printed morphemes in the following example:

> ***Aban***_twana_ ***aba***_khulu_ ***ban***_ga_***zi***_funda_ ***izin***_cwadi_ ***za***_bo._
> *Aba-ntwana aba-khulu ba-nga-zi-fund-a izin-ncwadi za-bo.*
>
> Children-who are big-they may read them-letters-of them.
> 'The big children may read their letters.'

In this sentence, the class 2 noun *abantwana* 'children' governs the subject concord *ba-* in the verb *bangazifunda* 'they may read them', as well as the possessive concord *za-* in *zabo* 'of them' and the adjective concord *aba-* in the qualificative *abakhulu* 'who are big'. The class 10 noun *izincwadi* 'letters' determines concordial agreement of the object concord *-zi-* in the verb.

The predominantly agglutinating nature of the Zulu language is clearly illustrated in the above sentence, each word of which consists of more than one morpheme. This complex morphological structure will be discussed very briefly by referring to two of the most complex word types, namely nouns and verbs.

Nouns as well as verbs in Zulu are constructed by means of the two generally recognized types of morphemes namely *roots* and *affixes*, the latter consisting of prefixes and suffixes. The majority of roots are bound morphemes since they do not constitute words by themselves, but require one or more affixes to complete the word. The root is generally regarded to be 'the core element of a word, the part which carries the basic meaning of a word.' (Poulos & Msimang 1996: 170). For instance, in the example *izincwadi,* the root that conveys the semantic significance of the word is *-ncwadi* 'letter'. The morphemes *i-* and *-zin-* are prefixes of the root.

A morphological distinction may be made between two types of nouns:

Firstly, nouns formed from roots that do not require suffixes. Such roots are not derived from other word categories, and cannot be reduced to any simpler form. The noun root *-ntu* which occurs in the noun *umuntu* 'person', is a complete root.

Secondly, nouns formed from roots that do require suffixes. Such roots are derived from other word categories, such as verb roots, adjective stems and ideophones. The verb root *-theng-* 'buy' occurs in the deverbative noun *umthengi* 'buyer' and needs the suffix *-i* for completeness. Nouns may also be derived from verb roots with extensions, also known as extended roots. In the example *umthengisi* 'seller' the extended root *-thengis-* 'sell' incorporates the causative extension *-is-* and also needs the suffix *-i* for completeness.

In the formation of nouns it should be noted that roots, which do not require any suffixes for completeness, as a well as roots to which final suffixes have been added, may both be termed noun stems (Poulos & Msimang 1996: 170).

In the case of the verb, the core element which expresses the basic meaning of the word is the verb root. The essential morphemes of a Zulu verb are a subject concord (except in the imperative and infinitive), a verb root and a terminative, as illustrated in the following example:

*ba-**fund**-a*
subject concord-root-terminative
'they read'

Over and above the subject concord, the form of which is determined by the class of the subject noun, a number of other morphemes may be prefixed to a verb root, e.g.

*ba-nga-zi-**fund**-a*
subject concord-potential morpheme-object concord-root-terminative
'they may read them'

It should be noted that whereas object concords also show concordial agreement with the class of the object noun, all other verbal affixes are class independent. Furthermore verbal affixes have a fixed order in the construction of verbs, with the object concord prefixed directly to the verb root.

Suffixes such as verbal extensions may be inserted between the verb root and the terminative. In the following example it will be noted that the terminative has changed to the negative *-i* in accordance with the negative prefix *a-*, e.g.

*a-ba-**fund**-is-i*
negative morpheme-subject concord-root-extension-terminative (negative)
'they do not teach'

Following this concise explanation of morphological rules in Zulu, various restrictions or constraints in the application of these rules are observed, e.g.

(a) separated or long-distance dependencies (e.g. affixes that cannot co-occur in the same word, i.e. incompatible morphemes such as the present tense morpheme *-ya-* with other tense morphemes, or with negative morphemes);

(b) roots showing irregular morphotactic behaviour (e.g. a certain group of verb roots are restricted to a specific suffix in the formation of imperatives);

(c) circumfixes, that is co-ordinated pairs consisting of a prefix and a suffix (e.g. a negative prefix which requires a negative suffix);

(d) other constraints that are feature-based rather than phonological (e.g. the constraint on copula prefixes to combine with certain noun classes);

(e) forward-looking feature requirements such as the presence of one morpheme requiring the presence of another morpheme (e.g. an object concord requires a verb root).

In subsequent sections certain issues of computational morphology and the challenges involved in handling linguistic constraints in an agglutinative language such as Zulu in a Xerox style finite-state computational morphological analysis by means of so-called *flag diacritics*, will be addressed.

3. COMPUTATIONAL APPROACH

The suitability of finite-state approaches to computational morphology is well known and has resulted in numerous software toolkits and development environments for this purpose. For the work reported on in this paper the state-of-the-art *Xerox finite-state toolkit* (Beesley and Karttunen 2003) is used.

The Xerox software tool for modelling the *morphotactics* is ***lexc***. An accurate specification of the Zulu word structure, that is all and only word roots in the language, all and only the affixes for all parts-of-speech (word categories) as well as a complete description of the valid combinations of these morphemes for forming all and only the words of Zulu, is created as a ***lexc*** script file and compiled into a so-called finite-state network. The words generated by this network are morphotactically well-formed, but still rather abstract lexical or morphophonemic words.

The *morphophonological* (phonological and orthographical) *alternations* are modelled with the Xerox regular expression language. Here the changes (orthographic/spelling) that take place between the lexical and surface words when morphemes are combined to form new words/word forms, are described. These regular expressions are then compiled into a finite-state network by means of the ***xfst*** tool.

Finally, the two mentioned finite-state networks are *combined* (composed) together into a *single network*, a so-called *lexical transducer*, which constitutes the morphological analyser. It is note-worthy that these finite-state networks (transducers) are *bi-directional* devices, which facilitate morphological analysis in the one direction and morphological generation in the other. It is customary to refer to the analyses as strings in the so-called upper language and to surface words as strings in the so-called lower language. It remains a challenge to build such lexical transducers that *analyse and generate all and only* the words of a given language, in this case Zulu.

A particularly useful device offered by the Xerox finite-state toolkit as an extension is the so-called *flag diacritics*. Flag diacritics provide a means of feature-setting and feature-unification that keep transducers small, enforce desirable results such as linguistic constraints, and simplify grammars. In particular, they are used to block illegal paths at run time by the analysis and generation routines. In ***lexc*** and ***xfst*** they are treated as multi-character symbols spelt according to the two templates *@operator.feature@* and *@operator.feature.value@*. Typical operators that we use are Unification Test, Positive (Re)Setting, Negative (Re)Setting, Require Test, and Disallow Test. For a detailed discussion, see Beesley and Karttunen (2003).

4. COMPUTATIONAL MODELLING AND IMPLEMENTATION OF CONSTRAINTS

The challenge of handling linguistic constraints such as the ones mentioned in Section 2 within the finite-state context may be met in various ways (see, for example, Beesley and Karttunen 2003). We mention two, namely finite-state filters and Xerox flag diacritics, and illustrate by means of an example why the Xerox flag diacritics are our preferred approach.

While a detailed discussion of the syntax of ***lexc*** is outside the scope of this article (see Beesley and Karttunen 2003), it suffices to think of the code fragment in *figure 1* as a simple finite-state transducer with three states **A, B** and **C** and bi-directional arcs between **A** and **B** and **B** and **C**, labelled **a** and **c** on the upper side and **b** and **d** on the lower side respectively, as shown in *figure 2.* A cascade of LEXICONs may then be thought of as such concatenations of transducers.

```
LEXICON A
a:b          B;
LEXICON B
c:d          C;
LEXICON C
             #;
```

Figure 1: Code fragment associated with finite automaton in figure 2. The # symbol indicates the end of the cascade of LEXICONs.

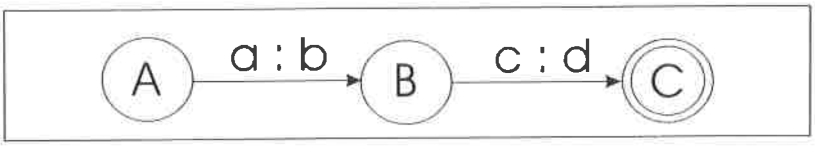

Figure 2: The transition from state A to state B consumes an input a and outputs b. Similarly, the transition from state B to state C consumes an input c and outputs d. State C, designated as final state, represents the # symbol in figure 1.

Consider the fragment (*Figure 3*) of an intuitive, but over-recognising and over-generating implementation of some class prefix - noun stem combinations in Zulu.

```
...
LEXICON Prefixes
umu   NStem;
aba   NStem;
umu   NStem;
imi   NStem;
isi   NStem;
izi   NStem;
in    NStem;
izin  NStem;
ubu   NStem;

LEXICON NStem
ntu         #;
thi         #;
khathi      #;
dlovu       #;
```

Figure 3: ***lexc*** script fragment for implementing class prefix - noun stem combinations.

The lexical transducer that results from an implementation, part of which is shown in *figure 3*, analyses and generates correct words such as *umuntu, abantu, ubuntu, umuthi, imithi, isikhathi, izikhathi, indlovu* and *izindlovu*, but also many invalid words, examples of which are shown in *figure 4*.

```
Lexc>  random-surf
NOTE:  Using SOURCE.
imithi
abatshe
izithi
izidlovu
imintu
imithi
izikhathi
izintu
izindlovu
imitshe
abatshe
izithi
amadlovu
imikhathi
```

Figure 4: The ***lexc*** tool command, `random-surf`, may be used to obtain a number of words that are successfully analysed by the network of figure 3. Only three of the above strings are valid Zulu words namely `imithi`, `izikhathi` and `izindlovu`, illustrating the extensive spurious overgeneration.

In order to constrain the combination of class prefixes with noun stems, we may employ a finite-state (regular expression) filter (*Figure 5*), appropriately composed with the lexical transducer (*Figure 6*).

```
Define rule1 0 <- [%^CL12|%^CL34|%^CL78|%^CL910|%^CL14];
Define rule2 $[[%^CL12 ?+ %^CL12]|[%^CL34 ?+ %^CL34]|[%^CL78 ?+
%^CL78]|[%^CL910 ?+ %^CL910]|[%^CL14 ?+ %^CL14]];
Read regex rule1 .o. rule2;
```

Figure 5: ***xfst*** script for the filter that may be used to ensure only valid noun prefix – noun stem combinations. Note: The `%`-symbol literalises the `^`-symbol, a special symbol in ***xfst*** regular expression syntax. The regular expression `rule2` simply checks that only valid class prefix – noun stem combinations are allowed by requiring that the multicharacter symbols, encoding the class information, occur in pairs of identical symbols, separated by any one or more symbols (`?+`), as shown above.

```
Multichar_Symbols
^CL12        ^CL34        ^CL78        ^CL910       ^CL14
...
LEXICON Prefixes
umu^CL12:umu             NStem;
aba^CL12:aba             NStem;
umu^CL34:umu             NStem;
imi^CL34:imi             NStem;
isi^CL78:isi             NStem;
izi^CL78:izi             NStem;
in^CL910:in              NStem;
izin^CL910:izin          NStem;
ubu^CL14:ubu             NStem;

LEXICON NStem
ntu^CL12:ntu             #;
ntu^CL14:ntu             #;
thi^CL34:thi             #;
khathi^CL78:khathi       #;
dlovu^CL910:dlovu        #;
```

Figure 6: **lexc** script fragment, implementing class prefix - noun stem combinations, to be composed with the filter in *figure 3*. Note: Noun class information is introduced in the upper language via the multi-character symbols as shown.

Finally, *figure 7* shows a flag diacritics (functional) equivalent of the network that results from the previously discussed composition of the filter in *figure 5* and the network in *figure 6*.

In the noun stem lexicon `NStem` of the script each noun stem is entered together with a flag diacritic (multi-character symbols with special syntax) capturing its class information, as shown in *figure 7*. The feature is CL, the class number, and the values are the strings 1-2, 3-4, 7-8, 9-10 and 14 respectively. In the LEXICON Prefixes the values of the feature CL are set by means of the P-operator and in the LEXICON NStem the unification is done with the U-operator. In this context, unification means that if the feature CL has been set for example, by means of @P.CL.1-2@, then the unification will succeed if and only if the value in the @U.CL.xxx@ flag diacritic is compatible with the current value of the feature. In particular, @U.CL.1-2@ will cause unification to succeed and @U.CL.3-4@ will cause it to fail. This unification takes place at runtime.

```
Multichar_Symbols
@P.CL.1-2@  @P.CL.3-4@  @P.CL.7-8@  @P.CL.9-10@
@P.CL.14@
...
LEXICON Prefixes
umu@P.CL.1-2@     NStem;
aba@P.CL.1-2@     NStem;
umu@P.CL.3-4@     NStem;
imi@P.CL.3-4@     NStem;
isi@P.CL.7-8@     NStem;
izi@P.CL.7-8@     NStem;
in@P.CL.9-10@     NStem;
izin@P.CL.9-10@   NStem;
ubu@P.CL.14@      NStem;

LEXICON NStem
ntu@U.CL.1-2@     #;
ntu@U.CL.14@      #;
thi@U.CL.3-4@     #;
khathi@U.CL.7-8@  #;
dlovu@U.CL.9-10@  #;
```

Figure 7: ***lexc*** script fragment implementing class prefix - noun stem combinations by means of flag diacritics.

While *figure 7* demonstrates the convenience and elegance of using flag diacritics, *figure 8* shows the accuracy achieved in the modelling by using this technique.

```
Lexc>  random-surf
NOTE:  Using SOURCE.

imithi
umuthi
imithi
umuthi
abantu
imithi
indlovu
ubuntu
abantu
abantu
abantu
umuntu
izikhathi
imithi
indlovu
```

Figure 8: As before, the ***lexc*** tool command, `random-surf`, is used to obtain a number of words that are successfully analysed by the network of *figure 7*. As expected, all the above strings are valid Zulu words. The repetition of words is due to the software, but what is significant for our purposes is the 100% correctness of the words generated.

The question arises as to what the computational implications of the two approaches are *for the user*. *Table 1* shows the respective sizes of the networks involved. As expected, the intuitive network is small and the networks of *figures 6* and *7* are similar to one another and of the same order of magnitude as that of *figure 3*. What is, however, significant is the size of the filter – almost an order of magnitude larger than the other networks for this particular example. While the composed network is the smallest of all, the intermediate step, namely the building of the filter network produces a large network, which has to be stored for use in the subsequent composition. For large coverage lexical transducers this phenomenon may render the use of filters impractical and problematic.

Network	Size (Kb)	States	Arcs
Figure 3	1.2	24	38
Filter in figure 5	10.1	113	678
Figure 6	2.3	36	47
Figures 5 and 6 composed	0.940	26	33
Figure 7	2.3	36	47

Table 1: Sizes of the various networks in figures 3, 5, 6 and 7.

In the Zulu morphological analyser under discussion, we made extensive use of the Xerox flag diacritics for the modelling of the morphotactics of linguistic constraints. Different modelling requirements were met by employing a number of different and appropriate operators available. In subsequent subsections the various kinds of constraints, as indicated in Section 2, are explained and illustrated by means of Zulu examples.

(a) **Separated or long-distance dependencies** include affixes that cannot co-occur in the same word. An example of such incompatibility is the long present tense morpheme *-ya-* which is incompatible with negative morphemes. The following Zulu examples *bayakhumbula* (positive) and *abakhumbuli* (negative) are analysed as follows:

bayakhumbula 'they remember'
ba[SC2]ya[LongPres]khumbul[VRoot]a[VerbTerm]

abakhumbuli 'they do not remember'
a[NegPre]ba[SC2]khumbul[VRoot]i[VerbTermNeg]

However, the ungrammatical form**a-ba-ya-khumbul-i* is not analysed since *-ya-* is incompatible with the two negative morphemes *-a-* and *-i*.

For the implementation of this constraint we employ the P-, U- and R-operators with respect to the NEG feature to enforce the appropriate behaviour. The relevant flag diacritics are shown in boldface in the ***lexc*** fragment below. Once the feature NEG has been set to the value ON by means of the flag diacritic @P.NEG.ON@ no unification is possible with the flag diacritic @U.NEG.OFF@ and any path containing these two multi-character symbols will be blocked. However, the use of @R.NEG.ON@ ensures that any valid path contains a preceding @P.NEG.ON@, thereby enforcing the co-occurrence of a negative prefix morpheme, such as a[NegPre] and a subsequent negative verb terminative, such as i[VerbTermNeg].

For the analysis of *bayakhumbula* a fragment of the appropriate cascade of LEXICONS in ***lexc*** is given in figure 9. The operational semantics of the U-operator are such that in the case of unification on a neutral (previously unset) feature, it acts like the P-operator. In this example the U-operator in LEXICON LongPres will set the possibly neutral feature NEG to the value OFF. The next occurrence of @U.NEG.OFF@ affects unification on the NEG feature. Note that the complete cascade is not shown since the focus is on the sequence of flag diacritics (in bold face) governing the relevant linguistic constraint.

```
LEXICON SubjectConc
ba[SC2]@P.SC.ON@:ba@P.SC.ON@   IntermedPrefixes;
...

LEXICON LongPres
ya[LongPres]@U.NEG.OFF@@P.LPT.ON@:ya@U.NEG.OFF@@P.LPT.ON@
                               ObjectConc;

LEXICON VRoot
khumbul                        VPSClass15;
...

LEXICON VerbTerm
a[VerbTerm]@U.NEG.OFF@@R.SC.ON@@D.FT@:
a@U.NEG.OFF@@R.SC.ON@@D.FT@           #;
```

Figure 9: ***lexc*** script fragment for the implementation of the separated dependency between *-ya-* and the final *-a* in *bayakhumbula.*

For the analysis of *abakhumbuli* the appropriate cascade of LEXICONS in ***lexc*** is given in *figure 10.*

```
LEXICON NegativePrefixA
a[NegPre]@P.NEG.ON@:a@P.NEG.ON@     SubjectConc;

LEXICON SubjectConc
ba[SC2]@P.SC.ON@:ba@P.SC.ON@  IntermedPrefixes;
...

LEXICON VRoot
khumbul     VPSClass15;
...

LEXICON VerbTerm
i[VerbTermNeg]@R.NEG.ON@@R.SC.ON@@D.FT@:
i@R.NEG.ON@@R.SC.ON@@D.FT@               #;
```

Figure 10: ***lexc*** script fragment for the implementation of the separated dependency between the initial *a-* and the final *-i* in *abakhumbuli.*

(b) Irregular morphotactic behaviour of roots/stems is illustrated in the restriction of compatibility of a specific suffix with a certain group of verb roots. In the formation of the imperative in Zulu, the verb stem (i.e. verb root plus verb terminative) suffixes the morpheme *-ni* if more than one person is being addressed, e.g.

buyani! ‘return!’ (plural)

analysed as

buy[VRoot]a[VerbTerm]ni[ImpSuf]

However, in the case of consonant verb roots in the plural imperative, the suffix of the verb stem is *-nini*, e.g.

dlanini! ‘eat!’ (plural)

analysed as

dl[VRoot]a[VerbTerm]nini[ImpSuf]

Here the U-operator with respect to the feature ConsVerbRoot combined with the value ON is used for limiting the suffix *-nini* to occur only with a particular sub-group of verb roots, namely consonant verb roots. The R-operator is used to enforce the presence of the imperative suffix *-nini* instead of the usual *-ni*.

The cascade of lexicons is given in figure 11.

```
LEXICON VRoot
dl                                  VMSClass15;

LEXICON VMSClass15
@U.CL.15@@U.ConsVerbRoot.ON@        EndVRootMarker;
...

LEXICON ImpSuffix
nini[ImpSuf]@R.ConsVerbRoot.ON@:
nini@R.ConsVerbRoot.ON@             #;
```

Figure 11: ***lexc*** script fragment that implements the co-occurrence of the consonant verb *-dl-* with the imperative suffix *-nini.*

(c) Circumfixes are co-ordinated pairs consisting of a prefix and a suffix, such as a negative prefix that requires a negative suffix. In the current prototype the following examples *asiphuzi* (present tense) and *asiphuzanga* (perfect tense) are analysed as:

asiphuzi ‘he does not drink’
a[NegPre]si[NegSC7]phuz[VRoot]i[VerbTermNeg]

asiphuzanga ‘he did not drink’
a[NegPre]si[NegSC7]phuz[VRoot]anga[VerbTermNeg]

It should be noted that this constraint does not apply to negative verbs in the future tense nor to verbs in the passive.

The constraints on the negative prefix *a-* requiring a verb terminative *-i* in the present tense and *-anga* in the perfect tense are modelled by means of the P- and R-operators. The implementation details are similar to the example *abakhumbuli* in subsection (a), above. Flag diacritics are used to bind together the two halves of a circumfix, as illustrated below. The first half is the negative prefix *a-*, which is set positively by the P-operator, while the second half is the verb terminative, either *-i* or *-anga*, depending on the tense, which undergoes a require test by means of the R-operator. For the require test to succeed in the example under discussion, the feature NEG must currently be set to the value indicated (i.e. ON), as illustrated in thc cascade of lexicons in *figurc 12*.

```
LEXICON NegativePrefixA
a[NegPre]@P.NEG.ON@:a@P.NEG.ON@     SubjectConc;

LEXICON SubjectConc
si[SC7]@P.SC.ON@:si@P.SC.ON@        IntermedPrefixes;


...

LEXICON VerbTerm
i[VerbTermNeg]@R.NEG.ON@@R.SC.ON@@D.FT@:i@R.NEG.ON@@R.SC.ON@
@D.FT@  #;
anga[VerbTermNeg]@R.NEG.ON@@R.SC.ON@:anga@R.NEG.ON@@R.SC.ON@
        #;
```

Figure 12: **lexc** script fragment that implements the circumfixes *a-* and *-i* in *asiphuzi* and *a-* and *-anga* in *asiphuzanga*.

(d) Other feature-based constraints are illustrated by means of the copula prefixes which combine only with certain noun prefixes. Each of the following three copula prefixes is restricted as far as its occurrence with noun prefixes is concerned, i.e.

- *yi-* combines with noun prefixes commencing with *i-*;
- *ngu-* combines with noun prefixes commencing with *u-* (excluding class 11), *o-* and *a-*; while
- *wu-* only combines with noun prefixes commencing with *u-* and *o-*.

Examples illustrating this constraint are:

yimithi 'they are trees'
yi[CopPre]i[NPrePre4]mi[BPre4]thi[NStem]

ngumfana 'it is the boy'
ngu[CopPre]u[NPrePre1]mu[BPre1]fana[NStem]

ngobaba 'it is father and company'
ngu[CopPre]o[NprePre2a] baba[NStem]

ngamanzi 'it is water'
ngu[CopPre]a[NPrePre6]ma[BPre6]nzi[NStem]

wubaba 'it is father'
wu[CopPre]u[NPrePre1a] baba[NStem]

wobaba 'it is father and company'
wu[CopPre]o[NprePre2a] baba[NStem]

This phenomenon is modelled by means of the P- and D-operators, as shown in the cascade of lexicons in *figure 13*. In this case, the P-operator is used to set the value of the indicated feature, namely the copula prefix (CopYi or CopNgu or CopWu), while the D-operator is a disallow test to ensure that the invalid copula prefix – noun prefix combinations are excluded. This is an example of the useful combination of the P- and D-type flag diacritics for restricting idiosyncratic morphological behaviour of prefixes.

```
LEXICON CopulaPrefixes
yi [CopPre]@P.CopYi.ON@:yi@P.CopYi.ON@            NounPrefixes;
ngu [CopPre]@P.CopNgu.ON@:ngu@P.CopNgu.ON@        NounPrefixes;
wu [CopPre]@P.CopWu.ON@:wu@P.CopWu.ON@            NounPrefixes;

LEXICON NounPrefixes
u[NPrePre1]mu[BPre1]@U.CL.1-2@ @P.NomSuf.ON@@D.CopYi@@D.Poss1a@:
^U^MU@U.CL.1-2@@P.NomSuf.ON@@D.CopYi@@D.Poss1a@
                                                 BeginNStemMarker;

a[NPrePre2]ba[BPre2]@U.CL.1-2@
@P.NomSuf.ON@@D.CopWu@@D.CopYi@@D.Poss1a@:
^A^BA@U.CL.1-2@@P.NomSuf.ON@@D.CopWu@@D.CopYi@@D.Poss1a@
                                                 BeginNStemMarker;

i[NPrePre4]mi[BPre4]@U.CL.3-4@
@P.NomSuf.ON@@D.CopNgu@@D.CopWu@@D.Poss1a@:
^I^MI@U.CL.3-4@@P.NomSuf.ON@@D.CopNgu@@D.CopWu@@D.Poss1a@
                                                 BeginNStemMarker;
```

Figure 13: ***lexc*** script fragment that implements the copula prefixes – noun prefixes constraints in section 4(d).

(e) Forward-looking feature requirements imply that the presence of one morpheme requires the presence of another morpheme, for instance an object concord must be followed by a verb root. This requirement is significant in the Bantu languages since verb-like constructions may also be formed from other roots or stems, which do not allow object concords. In the case of copula constructions for instance, the copula may be preceded by a subject concord (e.g. *u-ngumuntu* 'he is a person'), and the construction may also be negativised (*aka-ngumuntu* 'he is not a person'). Should this constraint on the occurrence of the object concord not be implemented, overgeneration will occur with object concords appearing in conjunction with subject concords, even in copula constructions.

The occurrence of an object concord marks the expectation that a verb root will follow (the P-operator, @P.Verb.ON@ in LEXICON ObjectConc in figure 14), which is indeed fulfilled by means of the R-operator in the subsequent LEXICON VExtContents.

An example including the object concord of the first person singular *-ngi-* is:

ungibona 'you see me'
u[SC2ps]ngi[OC1ps]bon[VRoot]a[VerbTerm]

```
LEXICON SubjectConc
u[SC2ps]@P.SC.ON@@U.NEG.OFF@:
u@P.SC.ON@@U.NEG.OFF@         IntermedPrefixes;
...

LEXICON ObjectConc
ngi[OC1ps]@P.OC.ON@@P.Verb.ON@:ngi@P.OC.ON@@P.Verb.ON@
BeginVRootMarker;
...

LEXICON VExtContents
...                           NomSufContents1;
@R.Verb.ON@                   VerbTerm;

LEXICON VerbTerm
a[VerbTerm]@U.NEG.OFF@@R.SC.ON@@D.FT@:
a@U.NEG.OFF@@R.SC.ON@@D.FT@    #;
```

Figure 14: ***lexc*** script fragment that enforces the combination of an object concord and a subsequent verb root.

Another example of a forward-looking feature requirement is the case of the progressive aspect prefix *-sa-* which modifies the meaning of a verb by adding the concept 'still' in the positive and the 'no longer' in the negative, e.g.

sisakhumbula	'We still remember'
asisakhumbuli	'We no longer remember'

However, *-sa-* becomes *-se-* if used in combination with an adjective stem in a copulative construction, as in:

basebancane	'she is still small'

In other words, the occurrence of *-se-* as progressive aspect prefix, requires the presence of an adjective stem. In this case the R-operator is used, as shown in the LEXICON ProgressiveSA in *figure 15*.

```
LEXICON ProgressiveSA
se[ProgPre]@R.AdjStem.ON@:se@R.AdjStem.ON@      AdjBasicPrefixes;
sa[ProgPre]@R.Verb.ON@:sa@R.Verb.ON@            FutureTense;
```

Figure 15: ***lexc*** script fragment that ensures that *-se-* occurs with an adjective stem and *-sa-* with a verb root.

This concludes the discussion of five different types of linguistic constraints that may be modelled and implemented by means of ***lexc*** lexicon cascades and flag diacritics. In the next section we address a rather different problem, namely the ad hoc nature of the formation of deverbative nouns for which a different computational solution is proposed.

5. MULTI-LEVEL APPROACH TO CERTAIN IDIOSYNCRATIC BEHAVIOUR

The form of idiosyncratic behaviour that is addressed in this section originates from the fact that deverbative nouns cannot arbitrarily be formed from any verb root (Van Eeden 1956: 712). At present the resolution of this issue, that is the valid combinations of noun prefixes and verb roots in the formation of deverbative nouns, is mainly determined in the current prototype of the analyser from entries in existing dictionaries (the known forms) and occurrences of such combinations in corpora (the as yet unlisted forms).

5.1 Root-level analysis of known forms

The computational approach followed in the case of known forms is to *list* the *known forms* in the noun stem lexicon of the morphological analyser, together with appropriate class information. In this way, the morphological *analysis/ generation* takes place *up to the stem-level*, with flag diacritics ensuring valid noun prefix – noun stem combinations as before. The example *umfundisi* 'preacher' would thus be listed in the noun stem lexicon of the analyser as follows:

```
LEXICON NStem
...
fundisi                NClass1-2;
...
```

The question now arising, is how the further analysis of a deverbative noun stem to verb root level should proceed. It may be argued that the analyser should produce all noun stems that may potentially originate from a verb root by means of noun prefixes and appropriate nominal suffixes, once a verb root is listed in the verb root lexicon of such an analyser. However, since the combinations of verb roots with noun prefixes and nominal suffixes are idiosyncratic, this approach is a potential source of significant spurious overgeneration if modelled according to any kind of explicit rules. This phenomenon may be thought of as the combination of three (separated) morphological entities without explicit rules. The absence of explicit rules that govern this process renders the use of additional flag diacritics somewhat cumbersome and impractical.

As an alternative approach to prevent such overgeneration we proceed in levels of analysis/generation and adopt the following procedure: A morphological analysis is performed up to the noun stem by the usual morphological analyser transducer, in which all known noun stems together with their class information are listed in the noun stem lexicon. A finite-state network that (a) contains all the valid deverbative extensions[1] as well as the various nominal suffixes in Zulu; and (b) is able to identify the verb root part, the deverbative extensions and nominal suffixes in any given noun stem, is composed onto the morphological analyser network. This then facilitates a further level of morphological analysis up to the verb root level, as in the following diagram:

Preprefix	Basic Prefix	Verb Root	Verb extension	Deverbative / nominal suffix
u-	-mu-	-fund-	-is-	-i

The fact that the known noun stems are listed in the noun stem lexicon of the analyser ensures that only valid verb root – nominal suffix combinations will be successfully analysed up to the noun stem level. By systematically isolating the finite sequence of suffixes, limited in number, at the next level, the verb root level of analysis/generation is achieved without the necessity or availability of an explicit verb root list.

[1] Possible combinations of extensions to verb roots are for instance: *-el-*, *-w-/-iw-*, *-ek-*, *-akal-*, *-is-*, *-an-*, *-ulul-*, *-elel-*, *-isw-*, *-isel-*, *-isan-*, *-iselw-*, *-elan-*, *-elisan-*, *-ululis-*, *-ululek-*, *-ululan-*.

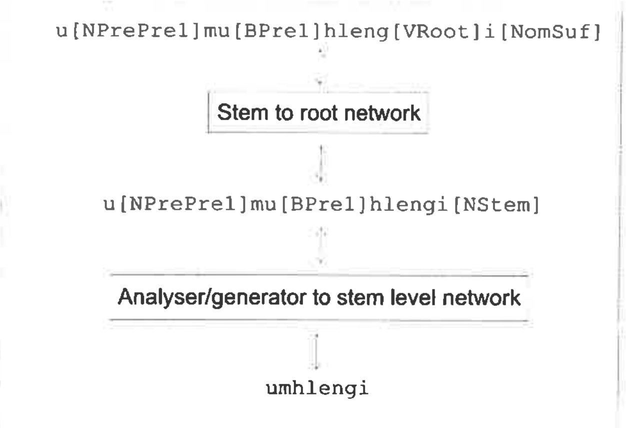
u[NPrePrel]mu[BPrel]hleng[VRoot]i[NomSuf]
Stem to root network
u[NPrePrel]mu[BPrel]hlengi[NStem]
Analyser/generator to stem level network
umhlengi

Figure 16: Analysis/generation proceeding in two steps

An example of a nominalised verb root is ***umhlengi*** 'helper', with ***-hlengi*** appearing in the noun stem lexicon of the lexc script of the analyser together with its class information (class 1–2). The morphological analyser produces the analysis:

`u[NPrePrel]mu[BPrel]hlengi[NStem]`

By appropriately *composing* (a well-defined operation on networks (Beesley and Karttunen 2003)) the two networks in figure 16, one network is obtained which represents the entire analysis/generation in one step:

`u[NPrePrel]mu[BPrel]hleng[VRoot]i[NomSuf]`

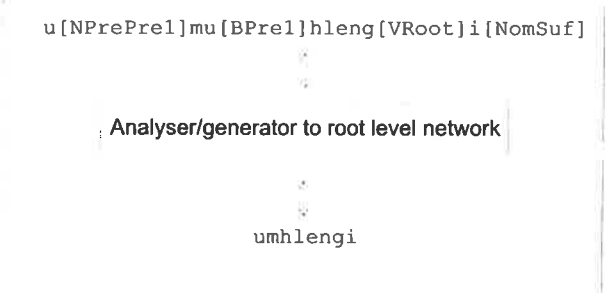
u[NPrePrel]mu[BPrel]hleng[VRoot]i[NomSuf]
Analyser/generator to root level network
umhlengi

Figure 17: Analysis/generation with composed network

5.2 Root-level analysis of extracted forms

We subsequently describe a procedure by which *unlisted forms* (new valid combinations of noun prefixes, verb roots and nominal suffixes) may be *computationally extracted* from language corpora as candidates for possible inclusion in the mentioned noun stem lexicon. This procedure makes use of the so-called guesser variant of the morphological analyser that identifies possible new deverbative noun stems. In this process, such new noun stems may be analysed to the verb root level by means of the 'composed on network' *without the requirement that the stems be listed* in the noun stem lexicon of the analyser, as discussed in 6.1. Strings for which such analysis succeeds, may be considered as candidates for new deverbative noun stems for inclusion in the noun stem lexicon.

In a case such as the deverbative noun *isihlengi* 'guard'[2], the noun stem *-hlengi* does not appear in the noun stem lexicon of the ***lexc*** script of the analyser with the appropriate class information (class 7–8), and will therefore not be successfully analysed.

However, by applying the guesser variant of the analyser to *isihlengi*, the output results in the following analyses:

```
i[NPrePre7]si[BPre7]hlengi+Guess[NStem]
i[NPrePre5]li[BPre5]sihlengi+Guess[NStem]
i[NPrePre9a]sihlengi+Guess[NStem]
```

This procedure may be successfully used for mining text corpora for 'new' deverbative noun stems or noun stems with 'new' class information. These candidate deverbative noun stems and their class information are verified manually by lexicographers/linguists before inclusion into the noun stem lexicon of the analyser. The same process is applicable to nouns derived from other word categories such as adjectives, relatives, pronouns and ideophones.

If the verb root network is composed onto the enhanced morphological analyser (with newly included noun stems), as described above, then *-hlengi* will be further analysed to the verb root level, yielding

```
i[NPrePre7]si[BPre7]hleng[VRoot]i[NomSuf]
```

Another similar procedure is the identification of nouns derived from *extended verb roots.* Two examples are *isingeniso* 'introduction' and *ingeniso* 'income', both nouns are derived from the extended verb root *-ngenis-* 'cause to enter; bring in; introduce'. In both examples the verbal extension that has been suffixed to the basic root *-ngen-* 'enter', is the causative extension *-is-* .

The first example *isingeniso* appears in the word list, and is analysed as follows:

```
i[NPrePre7]si[BPre7]ngeniso[NStem]
```

[2] As in *isihlengi sedolo* 'knee guard'

The second noun *ingeniso* does not occur in the electronically available word list, but when found in a corpus needs to be identified for inclusion. This word is analysed as follows by the guesser:

```
i[NPrePre9]n[BPre9]ngeniso+Guess[NStem]
i[NPrePre5]li[BPre5]ngeniso+Guess[NStem]
i[NPrePre9a]ngeniso+Guess[NStem]
```

After manual verification and inclusion in the noun stem lexicon, the resulting verb root level analysis of *ingeniso* would yield the following result:

```
i[NPrePre9]n[BPre9]ngen[VRoot]is[CausExt]o[NomSuf]
```

This concludes the discussion of the use of *multiple levels of finite-state transducers* for the analysis/generation of certain forms of idiosyncratic behaviour within the context of the development of a large coverage morphological analyser for Zulu.

6. CONCLUSION AND FUTURE WORK

In the paper it was illustrated in detail how Xerox flag diacritics are used to model linguistic constraints accurately in order to prevent overgeneration, and to implement such constraints efficiently in order to keep the resulting finite-state networks as small as possible. Furthermore, it was shown how *multiple levels of finite-state transducers* for the analysis/generation of certain forms of idiosyncratic behaviour, in combination with the so-called guesser variant of the Zulu morphological analyser/generator may be used in discovering *new*, as yet unlisted, deverbative nouns from corpus data. This technique proves particularly useful in extending, updating and enhancing machine-readable lexicons and serves the same purpose of information retrieval from running text as described for Swahili by Hurskainen (1997). The work forms part of an extended project aimed at the development of large coverage XML machine-readable lexicons for use in computational morphological analysis as well as for further natural language processing applications.

Acknowledgement

This material is based upon work supported by the National Research Foundation of South Africa under grant number 2053403. Any opinion, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Research Foundation.

REFERENCES

BEESLEY, Kenneth. R. & Lauri KARTTUNEN 2003. Finite state morphology. Stanford, CA: CSLI Publications.

BOSCH Sonja E. & Laurette PRETORIUS 2003. Building a computational morphological analyser/ generator for Zulu using the Xerox finite-state tools. *Proceedings of the Workshop on Finite-State Methods in Natural Language Processing, 10th Conference of the European Chapter of the Association for Computational Linguistics, April 13–14 2003, Budapest, Hungary*, pp. 27–34.

CRYSTAL, David 1997. *A Dictionary of Linguistics and Phonetics*. Oxford: Blackwell.

HURSKAINEN, Arvi 1992. A two-level formalism for the analysis of Bantu morphology: An application to Swahili. *Nordic Journal of African Studies* 1(1): 87–122.

------ 1997. A language sensitive approach to information management and retrieval: the case of Swahili. In: R. K. Herbert (ed.), *African Linguistics at the Crossroads: Papers from Kwaluseni*: 629–642. Köln: Rüdiger Köppe Verlag.

MATTHEWS, Peter H. 1997. *The Concise Oxford Dictionary of Linguistics*. Oxford: Oxford University Press.

MEINHOF, C. 1932. *Introduction to the Phonology of the Bantu Languages*. Berlin: Dietrich Reimer/Ernst Vohsen.

POULOS, George & Christian T. MSIMANG 1996. *A Linguistic Analysis of Zulu*. Pretoria: Via Afrika.

PRETORIUS, Laurette & Sonja BOSCH 2003. Finite-State Computational Morphology: An Analyzer Prototype for Zulu. *Machine Translation* 18: 195–216.

TRASK, R.L. 1996 *A Dictionary of Phonetics and Phonology*. London: Routledge.

VAN EEDEN, B. I. C. 1956. *Zoeloe Grammatika*. Universiteitsuitgewers en Boekhandelaars (Edms.) Stellenbosch: Beperk.