Word Synthesizer Engine

Uma Maheshwar Rao[1], *G. Christopher*[2], *M.*
Center for Applied Linguistics & Translation Studies
University of Hyderabad
Hyderabad-46
[1]guraohyd@gmail.com, [2]efthachris@gmail.com

This paper describes the development of a Morphological Generator, a generic Engine which can be used for any language by plugging in a specific language data-base. This Generator synthesizes all and only the well-formed word forms. These word forms include both inflectional and productive derivational forms. This Morphological Generator engine is independent of language and works effectively and is based on word-and-paradigm method. This Computational model uses machine learning method based on morphological data base developed using word and paradigm model of Morphology. This method not only ensures coverage but also evolvement.

The engine takes as input a root and along with it its inflectional categories (features) like gender, number, person and case in case of nouns and verbal categories in case of verbs and other relevant inflectional endings depending on the category.

In this paper we describe how the Morphological Generator handles all of the inflectional forms in addition to the productive derivational forms. The Input and output are in Shakti Standard Form (SSF). When tested with languages like Telugu, Hindi and Tamil their accuracy was 97.2%, 98% and 94% respectively.

INTRODUCTION. One of the crucial and integral parts of the major Natural Language Processing (NLP) application such as Machine Translation System (MT) is a morphological generator or a word synthesizer. The morphological analyzer and generator are two essential and basic tools in a machine translation system. A morphological analyzer processes a word and analyses it into its root, along with its grammatical information depending upon its word class. A morphological generator does exactly the reverse of it, i.e. given a root and the relevant morphological category and the grammatical information it generates the corresponding word form of that root, a projection of its morphological category. The current Morphological Generator is based on a dictionary of roots; paradigm information, a set of feature-values and a set of add and/or delete rules. The rest of the paper gives a detailed description of the Engine and it's working. The Morphological Generator will be called as WordGen hereafter in this paper.

1. ORGANIZATION OF DATA. The current morphological generator requires the following basic resources or the morphological data base that is described below. These resources are of three types:

1.1. A LEXICON. A lexicon is a dictionary containing a list of roots/stems, each with its lexical category and paradigm type. It is organized in the form of a simple linear, non-hierarchical sequence of the root, delimiter, lexical category, delimiter and paradigm type as in Table 1.

S.No	**Root/Stem form**	**Lexical Category (lcat)**	**Paradigm Type**
1	*"winu"*	"v"	*"koVnu"*
2	*"maMwri"*	"n"	*"gaxi"*
4	*"welika"*	"adj"	*"lewa"*
5	*"appudu"*	"adv"	*"appudu"*
6	*"iwadu"*	"pn"	*"vAdu"*

TABLE 1: LEXICON

1.2. Feature value table. The Feature Value Table is essentially a list of affixes with their morph-syntactic feature values like gender, number, person and the relevant morphological category information stored in the form of a table. Feature value Table contains lcat, affix and case associated with nouns and pronouns, tense, aspect, modal categories with or without gender, number and person associated with verbs, the affixes associated with Adjectives and locative nouns as shown in Table 2.

S.No	Rule No.	lcat	Affix	Gender	Number	Person
1	719	v	*iwi*	m	pl	2
2	653	n	*wopAtu*	null	sg	null
3	1649	pn	*lekuMdA*	null	pl	null
4	963	adj	*ti*	null	pl	null

Table 2: Feature Value Table

1.3. Synthesis Rule Set (morphophonemic Rules). Maximally projected word forms are generated by an exhaustive set of concatenation or synthesis rules. The synthesis rule set is an exhaustive rule set, essentially a combination of concatenation processes which add the desired suffix to the given root/stem and which itself is appropriately modified by the relevant deletion rules. Both the add rule and deletion rule may apply vacuously in case the value of the character string to be added or deleted is null.

S.No	Concatenation of suffix by Add Rule	Stem/Root modification Delete Rule	Paradigm Type	Rule No.
1	*A*	*u*	*vAdu*	1649
2	*IsAdA*	*iyyi*	*wiyyi*	653
3	*akuMdA*	*u*	*poVg?du*	719
4	*NNilekuMdA*	*du*	*snehiwudu*	963

Table 3 : Synthesizer Rule Set

2. Computational model of WordGen. The Morphological Generator i.e. the WordGen is built on the basis of the morphological rules extracted from the compilation of the relevant and exhaustive listing of paradigmatic forms. These sets of paradigmatic forms with a shared lexeme describe the morphology of the language. The lexeme is matched against each word form for a common maximally matched sequence and extracting the unmatched portion as formative/functional element.

For example consider the following members of a paradigm as in Table 4.

Lexeme: ***winu,v.***

Paradigmatic Members/wordforms	*Common Maximal match*	*Functional/ Formative Element*	*Feature Values*
winnAdu	*win*	nAdu	Past-m-sg-3
wiMtAdu	*wi*	MtAdu	np-m-sg-3
winadu	*win*	adu	neg-m-sg-3
wini	*win*	i	nf-past
wine	*win*	e	nf-adjl-ppl
winu	*winu*	0	imp-sg

TABLE 4 : DESCRIPTION OF DATA GENERATION

The generator accepts roots and their morphological information in terms of category and the functional elements to generate all the corresponding forms.

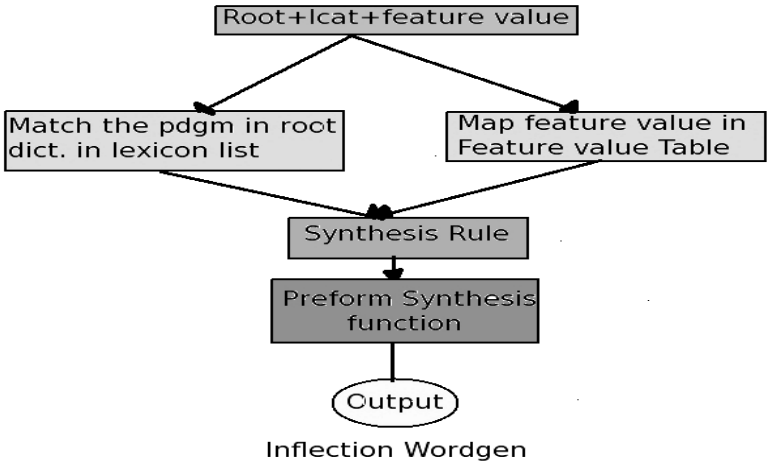


FIGURE 1: SYSTEM ARCHITECTURE

Figure 1 depicts the proposed model of *WordGen*. The architecture here involves the synthesis of word forms starting from the given root and the desired features, finding its category and the paradigm type in the lexical database, and then searching for the line in the synthesis table where the set of morpho-syntactic feature values are listed. Then accordingly carrying out delete and add functions, which involve the modification of the given root and the selection of the appropriate allomorph from the add rule followed by concatenation in the synthesis rule set.

The working of the *WordGen* can be viewed in step by step process, by using the data resources as shown below:
Root word= *vaccu*, lexical category=v, gender= any, number=any, person=any and suffix=*an*

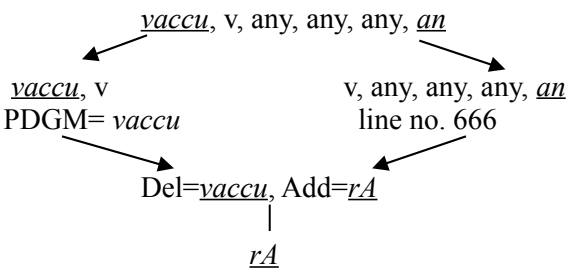


FIGURE 2

WordGen is able to generate the word forms with inflectional suffixes, but what about the productive suffixes. A new technique has been introduced to generate productive derivational word forms. A Floating Lexicon is devised to include derivational or compounding components of words. The basic architecture for this type of derivational module of *WordGen* is given below.

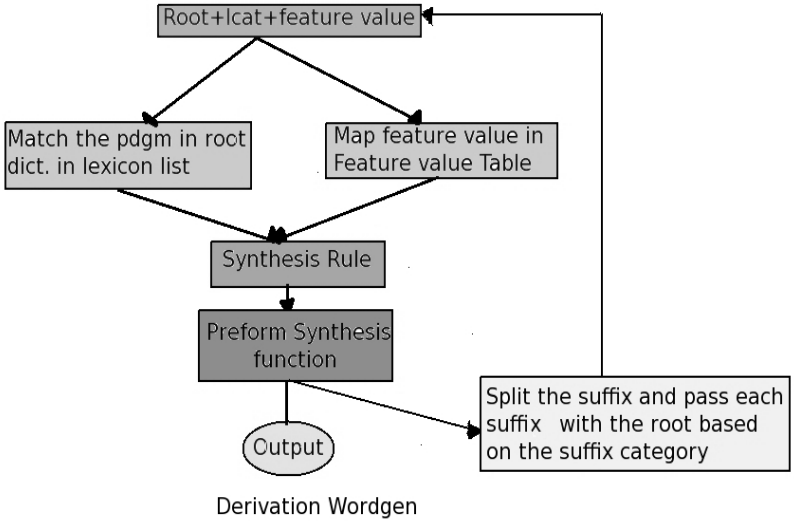
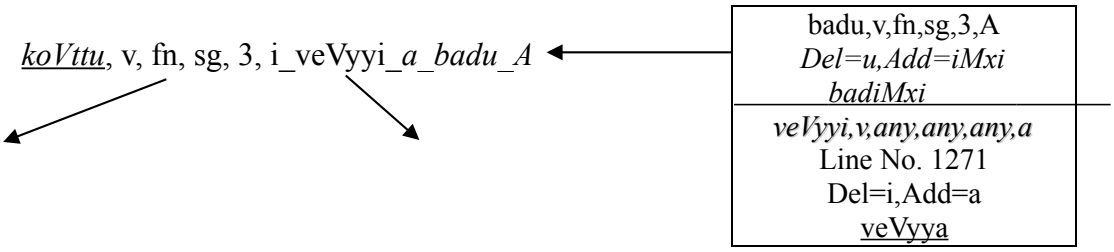


FIGURE 3

The working of the derivational *WordGen* can be viewed in step by step process, by using the relevant data resources.

Root word = *koVttu*, lexical category = v, gender = fn, number = sg, person = 3 and suffix = *i_veVyyi_a_badu_A*



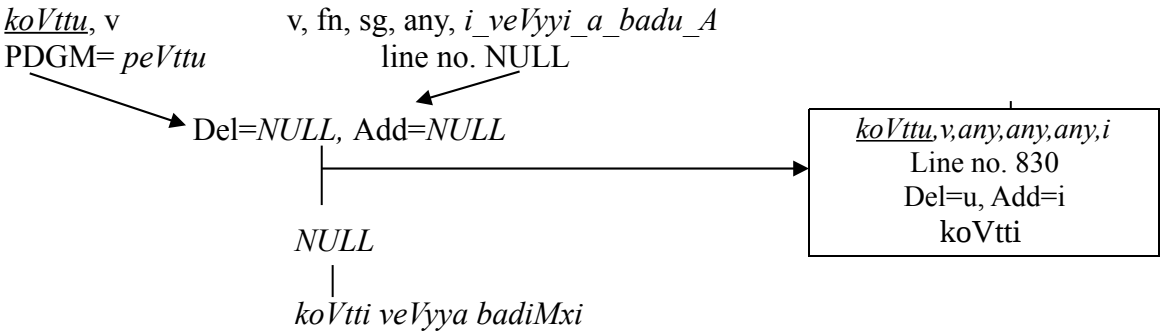


FIGURE 4

3. THE POSSIBLE COMBINATIONS OF FEATURE VALUES

NOUNS. root + mopho-syntatic functional categories
eg: noun + gender + number + person + d/o + case/other functional categories

PRONOUNS. root + mopho-syntatic functional categories
eg: pronoun + gender + number + person + d/o + case/other functional categories

ADJECTIVES. root + mopho-syntatic functional categories
eg: adjective + gender + number + person + d/o + case/other functional categories

VERBS. root + mopho-syntatic functional categories
eg: verb + gender + number + person + nf_m+ AuxVerb/other functional categories

NUMBER WORDS. root + mopho-syntatic functional categories
eg: number word + gender + number + person + d/o + case/other functional categories

NST (Nouns of Space and Time). root + mopho-syntatic functional categories
eg: locative + d/o + case/other functional categories

INDECLINABLES. They include the following.
Particles - look up in the root lexicon
eg: (Ewe, gAnI, EnA, etc.)
Adverbs - look up in the root lexicon
eg: adverb (bAgA, atlA, etc.)
Postpositions - look up in the root case marker list
eg: postposition (ni, ki, lo, va, etc.)

4. INPUT AND OUTPUT SPECIFICATION. Input for this computational model of Morphological Generator is in S*haskthi Standard Format (SSF); w*here we have a token number, token, pos-tag and its morphological analysis. All these are in different fields (Columns). It reads the fourth column i.e. *morph analysis*, in which the *1st field is root, 2nd is lex.cat, 3rd is gen, 4th is num, 5th is per, 6th is case (d/o), 7th is case marker/TAM, 8th is suffix*. By using all the seven elements of the

Morphological Analysis, the Generator generates the word forms and modifies the 2nd column i.e. *token* of the SSF format.

Input in SSF

```
<Sentence id="1">
1       ((      NP      <fs af='rAmudu,n,m,sg,3,d,ku,ku'>
1.1     rAmudu          NN      <fs af='rAmudu,n,m,sg,3,d,ku,ku'>
        ))
2       ((      NP      <fs af='Akali,n,m,sg,3,d,0,ku'>
2.1     Akali   NN      <fs af='Akali,n,m,sg,3,d,0,ku'>
        ))
3       ((      VGF     <fs af='veVyyi,v,m,sg,1,,A,A'>
3.1     veyyi   VM      <fs af='veVyyi,v,m,sg,1,,A,A'>
        ))
</Sentence>
```

Output in SSF

```
<Sentence id="1">
1       ((      NP      <fs af='rAmudu,n,m,sg,3,d,ku,ku'>
1.1     rAmudiki        NN      <fs af='rAmudu,n,m,sg,3,d,ku,ku'>
        ))
2       ((      NP      <fs af='Akali,n,m,sg,3,d,0,ku'>
2.1     Akali   NN      <fs af='Akali,n,m,sg,3,d,0,ku'>
        ))
3       ((      VGF     <fs af='veVyyi,v,n,sg,1,,A,A'>
3.1     vesiMxi         VM      <fs af='veVyyi,v,n,sg,1,,A,A'>
        ))
</Sentence>
```

5. Conclusion and Results. *WordGen* generates word forms for all the lexical classes where some sort of inflection is involved as in: nouns, pronouns, verbs, adjectives and locative nouns. This generator is designed to handle inflectional and productive derivational suffixes. The current version of the tool is integrated with IL-ILMT Hindi-Telugu, Telugu-Hindi, Telugu - Tamil and Tamil-Telugu systems (CALTS, University of Hyderabad).

References

Krishnamurti, Bh. 1985. *A Grammar of Modern Telugu*. Delhi, New York: Oxford University Press.

Uma maheswara Rao, G. 1999. *A Morphological Analyzer for Telugu* (electronic form). Hyderabad: University of Hyderabad.

Uma Maheshwar Rao, G., Chaithra, T.P., Santosh Jena. 2004. A Generic Architecture for Morphological Generators of Morphologically Complex Agglutinative Languages LECTURE COMPENDIUM, Symposium on Indian Morphology, Phonology & Language Engineering (SIMPLE'04), 13-16. Kharagpur; Indian Institute of Technology.

Uma maheswara Rao, G. and Amba Kulkarni, P. Christopher, Mala. 2007. *Morphological Analyzer and Its Functional Specifications for IL-ILMT System.* CALTS, Hyderabad: University of Hyderabad.

Uma maheswara Rao, G. and Amba Kulkarni, P. 2006. *Computer Applications in Indian Languages,* Hyderabad: The Centre for Distance Education, University of Hyderabad.