Morphological analyzer

A practical example for Tagalog

What to use this script for

- This script is a little sample of one of the steps of the morphological analyzer.

- It assumes we are already given a form with its lemma and the affix that is added to the root to form the lemma. The example included in the script is the form ‘nararamay’ that corresponds to the lemma ‘maramay’, which is formed by the prefix ‘ma’ plus the stem ‘ramay’.

Python Script

```
import re

'''
This function finds the first vowel of the root
'''
def first_Vowel(root):
        first_vowel_match =re.match('^(.*?)?([aeiou])(.+)?$',
root)
        vowel = first_vowel_match.group(2)
        return vowel
'''
Finds the first consonant of the root
'''
def first_Consonant(root):
        first_consonant_match
=re.match('^(.*?)?([aeiou])(.+)?$', root)
        consonant = ''
        if first_consonant_match.group(1):
                consonants =
first_consonant_match.group(1)
        consonant = consonants[0]
        return consonant
```

```
        return consonant
'''
Changes the 'm-' of the prefix by 'n-' which happens
for progressive and perfective aspects.
'''
def prefix_phon_change(prefix):
        m_match = re.match('^m(.+)', prefix)
        if m_match:
                prefix_changed = 'n'+ m_match.group(1)
        return prefix_changed

'''
Finally this function checks if the form matches the shape expected of a
verb with the affix 'ma', marked for progressive aspect. We assume we
already have the lemma and the prefix. If it matches, it prints the word
with its attributes.
'''
def Assign_progressive(lemma, forma, prefix):
        forma_attributes = []
        aspect = 'progressive'
        match_prefix = re.match('^'+prefix+'(.+)$', lemma)
```

```
        if match_prefix:
                root = match_prefix.group(1)
                consonant = first_Consonant(root)
                vowel = first_Vowel(root)
                prefix = prefix_phon_change(prefix)
                match_form = re.match(prefix+consonant+vowel+root, forma)
                if match_form:
                        forma_attributes.append(forma)
                        forma_attributes.append(lemma)
                        forma_attributes.append('verb')
                        forma_attributes.append(aspect)
                else:
                        print('Sorry, this form does not match the criteria')
        return forma_attributes

forma = 'nararamay'
lemma = 'maramay'
prefix = 'ma'

output = Assign_progressive(lemma, forma, prefix)
print('\t'.join(output))
```

Script 3/3

Functions

We have 4 functions:

- *Assign_progressive:* this function checks if the form of the word matches the form that a progressive verb with the affix ‘ma’ should have. If it matches it, it prints the form with its lemma and attributes.
- *prefix_phon_change:* this one changes the prefix ‘ma’ into ‘na’ which happens for the progressive aspect.
- *first_Vowel*: finds the first vowel of the root
- *first_Consonant*: finds the first consonant of the root if it has it

Then the first consonant and first vowel are used to match the reduplication.

Examples

To try the script with other forms of progressive, ‘ma’-affixed verbs, find the examples below and substitute the existing one in the script. Change both the form and the lemma in lines 56 and 57 of the script respectively.

forma	lemma	meaning	POS	aspect
natutulog	matulog	sleep	verb	progressive
natitiyak	matiyak	ensure	verb	progressive
namumuhunan	mamuhunan	invest	verb	progressive
nararamay	maramay	implicate	verb	progressive

Time to try it yourself!

MADRID, SPAIN

José Echegaray 8 , building 3, office 4
Parque Empresarial Las Rozas
28232 Las Rozas

SAN FRANCISCO, USA

1700 Montgomery Street, Suite 101
CA 94111