2nd International Conference on Arabic Language Resources and Tools

Cairo (Egypt), 22 - 23 April 2009

Implementation of infixes and circumfixes in the spellcheckers

Taha Zerrouki‹*, Amar Balla

*National institute of Computing INI Algiers
High Council of Arabic language in Algeria
t_zerrouki@ini.dz, taha_zerrouki@gawab.com
a_balla@ini.dz

Abstract

Most of spell checkers and morphological analyzers of natural languages are based on the affixes (suffixes and prefixes) extraction. But there are some other languages which use another kind of word changes like infixes and circumfixes. An infix is an affix that is inserted within a root. or stem. A circumfix is an affix, a morpheme that is placed around another morpheme. Circumfixes contrast with prefixes, attached to the beginning of the words; suffixes, that are attached at the end; and infixes, inserted in the middle.

This features are not implemented now in spellcheckers, and can give more puissance and facility to model more languages rules and affixations. This paper concerns Arabic language which uses internal word changes and has more affixes dependency, and diacritics (HARAKAT) which are ignored usually. These characteristics exist in other languages. In this paper, we illustrate a try to add infixes, circumfixes support, ignoring diacritics to open source spell checkers Aspell and Hunspell. And what's the advantages of this new features in the languages morphology.

Introduction

Most of actual spell checkers are based on the word segmentation into prefixes, stems and suffixes, then check if the found affixes are permitted for the given stem according to a word list and a affixes rules.
In some cases, words have internal changes which can't be presented with affixes. So, developers use other complex solution to do it. There are some kind of internal changes: ignoring diacritics, infixes and circumfixes. Those kinds exist in several natural languages, but not supported by most of spell checkers.
In this paper, we are explaining how these changes work, and trying to implement this new features by modifieg in Hunspell and Aspell open source spellcheckers.

Definitions

We define in this section infixes, ignoring diacritics, circumfixes, Hunspell spell checker, Aspell spell checker.

Infixes definition

Infix is an affix inserted inside another morpheme (Hartmann, 1972) . This is common in Austronesian and Austroasiatic languages. For example, the Tagalog language has borrowed the English word *graduate* as a verb. In this language, a grammatical form similar to the active voice is formed by adding the infix <*um*> as close to the left edge of the verb(provided that the /m/ does not act as the end of a syllable) , so a speaker saying "I graduated" uses the derived form *gr**um**aduate*. (It is conventional to set off infixes with <angle brackets>, rather than the hyphens used to set off prefixes and suffixes.)

The Semitic languages have a form of ablaut (changing the vowels within words, as in English *sing, sang, sung, song)* which is sometimes called infixation, but there is often no identifiable infix common across the vocabulary. However, Arabic uses a common infix, <ت> <*t*> for Form VIII verbs, usually a reflexive of Form I. It is placed after the first consonant of the root; an epenthetic *i-* prefix is added since words cannot begin with a consonant cluster. An example is اجتهد *ijtahada* "he worked hard", from جهد *jahada* "he stroves". (The words "ijtihad" and "jihad" are nouns derived from these two verbs.)

In Seri some verbs form the plural stem with infixation of *-tóo-* after the first vowel of the root; compare the singular stem *ic* 'plant (verb)' with the plural stem *itóoc*. Examples: *itíc* 'did s/he plant it?' and *iti**tóo**c* 'did they sow it?'.

English has very few infixes, and it does have marginal. A few are heard in colloquial speech, and a couple more are found in technical terminology.

- The infix <*iz*> or <*izn*> is characteristic of hip-hop slang, for example *h**iz**ouse* for *house* and *sh**izn**it* for *shit*. Infixes occur in some language games. The <*ma*> infix, whose distribution was documented by linguist Alan C. L. Yu, gives a word an ironic pseudo-sophistication, as in *sophisti**ma**cated, saxo**ma**phone,* and *edu**ma**cation*.
- Chemical nomenclature includes the infixes <*pe*>, signifying complete hydrogenation (from *pi**pe**ridine),* and <*et*> (from ***et**hyl),* signifying the ethyl radical C_2H_5. Thus from *picoline* is derived *pi**pe**coline,* and from *lutidine* is derived

*lu**pe**tidine;* from *phenidine* and *xanthoxylin* are derived *phen**et**idine* and *xanthoxyl**et**in*.

Tmesis is sometimes considered a type of infixation. It is found in English profanity, such as *in**fucking**credible* and *abso**bloody**lutely*. See the article expletive infixation.

Note that sequence of prefixes or suffixes do not result in infixes: An infix must be internal to a single morpheme. Thus the word *originally,* formed by adding the suffix *-ly* to *original,* does not turn the suffix *-al* into an infix. There is simply a sequence of two suffixes, *origin-al-ly*. In order for *-al-* to be considered an infix, it would have to be inserted in the non-existent word **originly*. The "infixes" in the Bantu languages are generally sequence of prefixes of these types.

Circumfix definition

A **circumfix** is an affix, a morpheme that is placed around another morpheme. Circumfixes contrast with prefixes, attached to the beginning of the words; suffixes, that are attached at the end; and infixes, inserted in the middle. See also epenthesis. Circumfixes are extremely common in Indonesian (Alan 2004).

The circumfix is probably most widely known from the German past participle (*ge- -t* for regular verbs). The verb *spielen*, for example, has the participle *gespielt*. Dutch has a similar system (*spelen – gespeeld* in this case)(Alan 2004).

In Hebrew, *magdelet* "magnifier", for example, the root is *gdl* "big" (in the H-stem *hagdel* "to enlarge") and the circumfix is *m- -et*.

In Japanese, the honorific circumfix *o- -ni naru* and *o- -suru* are used; for example *yomu → o-yomi ni naru* (respectful), *o-yomi suru* (humble) (Boeckx, 2004).

In Berber languages the feminine is marked with the circumfix *t…t*. The word *afus* (hand) becomes *tafust*.

The negation in Guaraní is also done with circumfixes *nd- -i* and *nd- -mo'ãi* for future negations.

In Arabic, circumfix is common, especially in the future tense conjugation, like y-ktb-on ي-كتب-ون.

Aspell definition

GNU Aspell is a spell checker designed to eventually replace Ispell. It can either be used as a library or as an independent spell checker.. Aspell support Unicode and multiple dictionaries at once.
It works on multiple plateformes and can be used by a lot of application (Aspell, 2007).
.
its main features are

- Is an actual library that other programs can link to instead of having to use it through a pipe.
- Can learn from user's misspellings.
- Support Unicode and multi-languages and multiple dictionaries .
- Support compound words
- Dictionary Compression.
- Simple affix rules syntax

Hunspell definition

Hanspal Spellchecker is the next generation of Myspell, has been improved in order to support additional features for European languages, especially for Hungarian language, as well as other languages such as German and Turkish (Hunspell, 2007; Németh et al. 2004; Halacsy et al. 2004).
I had participated in program development to add new Arabic features which ignore Arabic diacritics (harakat), this features is added in 1.1.5 version (Hunspell, 2007).
Main features of Hunspell spell checker and morphological analyzer:

1. Unicode support (affix rules work only with the first 65535 Unicode characters(
2. Morphological analysis (in custom item and arrangement style(
3. Max. 65535 affix classes and twofold affix stripping (for agglutinative languages, like Azeri, Basque, Estonian, Finnish, Hungarian, Turkish, etc)
4. Support complex compoundings (for example, Hungarian and German)
5. Support language specific features (for example, special casing of Azeri and Turkish dotted i, or German sharp s)
6. Handle conditional affixes, circumfixes, fogemorphemes, forbidden words, pseudoroots and homonyms.

Spellchecking

In this paragraph we talk about how Spellcheckers work, especially the auditors aspell and Hunspell. We have chosen open source spell checkers, because that they allow code access, modification, and give description to build a new dictionary.
Open source spellcheckers are derived from Ispell which gave Myspell, Aspell and Hunspell. This checkers are universal and multi-lingual.
A Language pack usually consists of two files, the dictionary file and affix rules file (Table 1).
In Hunspell, there are tow files, word list ar.dic and affix file ar.aff. In Aspell there are compressed word list ar.wl, and data affix rules ar_affix.dat and others files for configuration.
For two cases, the same syntax is used because it's inherited from Ispell, but there are some differences.
Affix file: contains a list of possible affixes and the rules of application,
word list file: contains a list of words and possible rules for each word.

Words check

- When launching the program, the Language files is loading.
- - Convert the text into words.
- for each word:
- search for the prefix according to the first letter.
- the affix can be null
- The prefix is deleted, add strip to the greeted stem,
- search the resulted stem in the dictionary.
- If the resulted stem exists, it tests if the affix rule is accepted for this word.

Example

affix file	word list	correct word	incorrect word
SFX T Y 1 SFX T 0 ly . PFX Y Y 1 PFX Y 0 un .	like/TY	unlike likely	lylike Likeun liked

Table 1 format of affixes in spellcheckers

Suggestions :

When the spell checker find a incorrect word, it gives suggestions, by changing one or more letters to find a nearly correct word.

New features

After studying spell checkers characteristics and features in order to add Arabic language support (Zerrouki, 2007; Ayaspell, 2008), we note that needed features can't be expressed by existing features, but it must be add, like ignoring vocalization (HARAKAT and TATWEEL) and the internal changes in the word (إعلال وإبدال , geminating).

In order to explain the impact of infixes in Arabic dictionary, we give some statistics of ayaspell project:
In first version of the dictionary, there are :

- 14391 verbs, more than 4000 are made by inffixation.
- Nouns: 28434 derivation nouns, 13374 nominals, 8408 subject nouns, 1809 object nouns, 4843 others classification of nouns.
- 960 prepositions
- 12177 special nouns.

With the use of infixes, we can – in theory- derivate more then 32 words from 10000 verbs, these will reduce the dictionary to 23000 words (10000 verbs, 1000 preposition, 12000 nouns) instead of 55000 words existing actually.
We have modified the code of Aspell and Hunspell in order to add new features to resolve the following problems:

- ignoring Arabic diacritics.
- Internal changes with the use of infixes.
- Dependency of suffixes and prefixes with the use of circumfixes.

Ignoring Arabic diacritics

Most of Arabic texts are unvocalized, then spell checkers ignore Arabic diacritics and tatweel. For example the word كَتَبَ is incorrect if the spell checker doesn't ignore diacritics.

Solution
We have added a new feature on the spell checker config file, and modified the code. When the spellchecker read a word, it ignores specified characters from the word.
For Hunspell case (Table 2), we have added the IGNORE flag into affix file.
This feature can be used in other languages

example

ar.aff	ar.dic	txt.good
SET UTF8 IGNORE ـًـَـِـ	كتب	كـــــــــتب كَتَبَ كُتِـــــــب ـكتبــــــــ

Table 2 Ignoring Arabic diacritics in Hunspell
For Aspell (Table 3), we have added a new feature "DIACRITICS" to ignore followed letters in parameter.
We have named this feature "DIACRITICS" instead of IGNORE, because there is another feature with the same name to ignore accented letters.
The syntax of this feature in the .dat file is illustrate in the example.

Example:

ar.dat	ar.wl	txt.good
SET UTF8 diacritics ـًـَـِـ	كتب	كـــــــــتب كَتَبَ كُتِـــــــب ـكتبــــــــ

Table 3 Ignoring Arabic diacritics in Aspell

Infixes

The most difficult in the spellchecking, is the representation of internal word changes with affixes, for example the verb قام

قام <= قامت، <= قمت

Qam=>qamt (add suffix 't')
Qam =>qmt (add suffix 't' and strip the 'a').
Qam=>yqom (add prefix 'y' and swap 'a' into 'o')
To represent this case we must add different stems to dictionary which increase its size and make affix rules more complex (Table 4 ,Table 5).

example:

Ar.af	Ar.dic	good	wrong
#t suffx SFX T Y 1 SFX T 0 ت #y prefix PFX Y Y 1 PFX Y 0 ي	قام/T قم/T قوم/Y	قامت قمت يقوم	يقام

Table 4: internal changes in Arabic word

Ar.af	Ar.dic	good	wrong
#t suffx SFX T Y 1 SFX T 0 t #y prefix PFX Y Y 1 PFX Y 0 t	qam/T qm/T qom/Y	qamt qmt yqom	yqam

Table 5: internal changes in Arabic word transliterated

Solution

We have added a new feature on the affix rules, we express any letters in a word by a period '.' (Table 6, Table 7)

For example, the prefix 'y.o' means that the 'y' is a prefix, and the 'o' is added after the first letter and we obain,

Qm+y.o=> **yqom**

The suffix 'y.' means tha 'y' is added before the last letter of word. Then we obtain

Istql + y. => istq**y**l

We can also express striping internal letters in the same way. The strip condition 'a.' in a suffix rule means that the 'a' letter before the last letter, will be striped.

Example : istq**a**l (strip 'a.') => istql.

Example:

The previous examples are presented like this,

Ar.aff	Ar.dic	good	wrong
SET UTF8 # لزيادة تاء الماضي SFX T Y 21 SFX T 0 ت ا. ت . T سابقة # لزيادة ياء المضارع PFX Y Y 1 ي.و ا. Yسابقة	#مدخل واحد فقط قام/TY	قامت قمت يقوم	يقام

Table 6: use of infixes feature in Arabic

Ar.aff	Ar.dic	good	wrong
SET UTF8 #suffix t SFX T Y 21 SFX T 0 t SFX T a. t #prefix y PFX Y Y 1 PFX Y .a y.o	#مدخل واحد فقط qam/TY	qamt qmt yqom	yqam

Table 7 : use of infixes feature in Arabic

It's possible to use this feature for other languages, we can generate 'swum' and 'swam' from 'swim' in English.

We tested this feature on Aspell and Hunspell, and we had good results.

Circumfixes

In Arabic verb conjugation, it's common to have dependency affixes, like y-ktb-on. Hunspell and Aspell (in 0.61 version) offer a possibility to combine affixes, and express affixes dependency with circomfix feature. Both of spellcheckers use the same rule (Table 8).

But we think that this feature is not suitable for languages with a lot of affixes dependency because rules become more complex, and hard to maintain.

Example

ar.aff	ar.dic	txt.good	txt.wrong
CIRCUMFFIX X PFX Y Y 2 PFX Y 0 ي/X PFX Y 0 ت/X PFX W Y 1 PFX W 0 ت/X SFX N Y 2 SFX N 0 ون/XY SFX N 0 ن/XW	فعل/N	يفعلون تفعلون تفعلن	يفعل فعلون فعلن تفعل يفعلن

Table 8 circumfix in Arabic word

In this case:

- we have the rule
  - SFX N 0 ون/XY

  Which means that the flag X links that the suffix ون 'on' with the prefix Y.
  - It gives yfalon, tfalon
  - The rule
  - SFX N 0 ن/XW
  - Combines the suffix 'on' with the prefix W, it gives tfaln but not yfaln.

We note that this will complicate the rules when we have more classifications.

solution
We suggest to integrate dependent affixes in the same rule (Table 9, Table 10), like this 'y-on','t-on','t-n', called circumfixes.
The '-' substitutes the stem.
Fal +y-on =>**y**fal**on**
Fal +t-on =>**t**fal**on**
Fal +t-n =>**t**fal**n**
We activate this feature on the affix file with the SPLITAFFIX flag.

ar.aff	ar.dic	txt.good	txt.wrong
SPLITAFFIX PFX Y Y 5 PFX Y 0 ي–ون PFX Y 0 ت–ون PFX Y 0 ت–ن PFX Y 0 ي PFX Y 0 ت	/فعلN	يفعلون تفعلون تفعلن يفعل تفعل	فعلون فعلن يفعلن ي–ونفعل ت–ونفعل ت–نفعل

Table 9: use of circumfix with the new feature

ar.aff	ar.dic	txt.good	txt.wrong
SPLITAFFIX PFX Y Y 5 PFX Y 0 y-on PFX Y 0 t-on PFX Y 0 t-n PFX Y 0 y PFX Y 0 t	fal/N	Yfalon Tfalon Tfaln Tfal Yfal	Falon Faln Yfaln y-onfal t-onfal t-nfal

Table 10 : use of circumfix with the new feature (transliterated)

Notes
- affixes without '-' are treated normaly.
- It's possible to combine circumfixes and infixes.
- Circumfixes expressed with prefix rules are processed as prefixes with additional suffixes.
- Circumfixes expressed with suffix rules are processed as suffixes with additional prefixes.
- It's possible to combine circumfixes.
- The difference between prefixed circumfixes an suffixed circumfixes is the side of striping.

Example (Table 9)
−The prefix
PFX Y i y-on
Strips 'i' from the word begin :
Istfal => ystfalon
The suffix
SFX Y a y-on
Strips 'a' from the word end :
mcha => ymchon
- If the verb needs to have strip from tow side, we can express this by combining circumfixes
- Istqsa => **y**stqs**on**
PFX Y i y-on/S : strip 'i' from word begin
SFX S a 0 : strip 'a' from word end.

We have tested this feature with Huspell, we obtained good results. In the aspell case tests are in progress.

Conclusion

In this article we had illustrated the need for some new features to process infixes, circumfixes and Arabic diacritics ignore. We have attempted programming these new features in the open source spell checkers aspell and Hunspell.
It should be noted that the philosophy of open source gives a strong push for continuous development of programms at low cost material and human resources, and contribute to the progress and advancement, through the Arabic support in this area.
These advantages have been applied in preparing the Dictionary of the Arabic language and the properties Al-I'lal, the letters replacement and diacritics ignore.
We intend in the future to think about two important solution for the Arabic language to consider diacritics during the spell checking, and the support of geminating in affixation.

References

Ayaspell (2008), Arabic dictionary project http://ayaspell.sourceforge.net
Hunspell (2007) http://hunspell.sourceforge.net
Aspell (2007) spellchecker http://aspell.net/.
Boeckx, Cedric & Fumikazu Niinuma (2004), "Conditions on Agreement in Japanese", Natural Language and Linguistic Theory 22 (3):453-480, <http://www.springerlink.com/content/xp77097378842 286/>Németh L. and al. (2004) Leveraging the open-source ispell codebase for minority language analysis. In Proceedings of SALTMIL. European Language Resources Association.
Halacsy Peter, Andras Kornai, Laszlo Nemeth, Andras Rung, Istvan Szakadat, and Vikto Tron. (2004). Creating open language resources for Hungarian. In Proceedings of the LREC, Lisbon, Portugal
Schaback Johannes and Fang Li. (2007). Multi-level feature extraction for spelling correction. In IJCAI-2007 Workshop on Analytics for Noisy Unstructured Text Data, pages 79–86, Hyderabad, India
Hartmann, R.R.K., and F.C. Stork. (1972). Dictionary of language and linguistics. London: Applied Science
Alan C. L. Yu (2004) Reduplication in English Homeric Infixation, University of Chicago, http://washo.uchicago.edu/pub/nels34.pdf
Zerrouki T. 2007, Programming sides to support arabic language in Hunspell open source spellchecker, Days study on Arabic language processing, Bechar university, Algeria.
Zerrouki T. 2007, Infixes and Circumfixes for support Arabic language spellchecking, Arabic softwares conference, High Council of Arabic language, Algeria