Optimizing the construction of the English morphological analyser

Hurskainen, Arvi

University of Helsinki, Institute for Asian and African Studies

2020

Hurskainen , A 2020 ' Optimizing the construction of the English morphological analyser ' Technical Reports on Language Technology , no. 54 , University of Helsinki, Institute for Asian and African Studies , Helsinki . < http://www.njas.helsinki.fi/salama/optimizing-the-construction-of-the-english-morphological-analyser.pdf >

http://hdl.handle.net/10138/317510

Technical Reports on Language Technology
Report No 54, 2020
http://www.njas.helsinki.fi/salama

Optimizing the construction of the English morphological analyser

Arvi Hurskainen
Department of Languages, Box 59
FIN-00014 University of Helsinki, Finland
arvi.hurskainen@helsinki.fi

Abstract

The construction of a morphological analyser for English is a fairly simple operation. The language has only a few morphological features, and they can be easily described. However, listing all wordforms as separate entries in the lexicon is certainly not the optimal solution. When using finite state transducers as developing environments, it is customary to list the word stems of various POS categories into separate sub-lexicons, and the inflection suffixes (and prefixes) into other sub-lexicons. Because the comprehensive analysis system tends to expand to dimensions, which are uncomfortable, or sometimes impossible, to manage, there is strong motivation to condense the lexicon wherever possible. This report describes the method, where verbs, and words derived from them, such as adjectives and nouns, are listed as underspecified entities. In a later phase these entities are then processed into separate readings, so that the readings can be disambiguated on the basis of context. The method condenses the lexicon considerably and it is easier to maintain, when the stems are in one place.

Key Words: *morphology, word analysis, machine translation.*

1 Introduction

In this report I try to demonstrate the method, how the morphological lexicon can be condensed and easier to manage. In the method we use under-specification of lexical entries. After analysis, the underspecified entries are 'written out' into non-ambiguous readings, so that the disambiguation rules can be applied to them.

The method suits to such cases, where the wordform has more than one interpretation. The ambiguity may concern the POS-internal ambiguity or ambiguity between POS categories. The POS-internal ambiguity is fairly easy to implement, because the POS tag remains the same in all cases. The ambiguity between POS categories is more difficult to implement, because the original tag must be changed into a new one.

The approach proposed here is related to the sign theory. In it, words are considered to be derived from more basic elements called signs. For example, the sign act would produce such words as *act_N act_V, acted_A, acting_A, acting_N,* and *actor_N*.

In theory, it is possible to list into the lexicon only the sign act and derive all other forms from it, using relevant sub-lexicons and entries in them.

If this method is used for all signs, there is a danger of overproduction, which causes difficulties in disambiguation. Therefore, we must consider carefully when to use the approach described here. For example, we can assume that the gerund form of a verb can generally function also as a noun or adjective. Also, the participial form of the verb can occur as an adjective. On the other hand, the infinitive form of the verb can often occur as a noun, but in many cases not. The noun corresponding to the verb has often a different form, although the root is the same. Compare, for example, the following: *grow* > *growth*, *see* > *sight*, *think* > *thought*. It seems plausible that the nouns corresponding to the infinitive form of the verb must be listed separately.

Below we see how the finite-state lexicon is implemented on the lines described above.

2 English verbs described as underspecified entries

In the morphological lexicon, there is currently about 3900 verbs with regular inflection, In addition, there are the irregularly inflecting verbs. Both of the verb types require different description types. First, we take a look at regular verbs

2.1 Regular verbs

When considering verb inflection, the basic verb type is the one, where the verb ends in a consonant. In it, the suffixes are *-s*, *-ed*, and *-ing*. Another common group consists of verbs ending in the vowel *e*. The third group are the verbs ending in *y*. Then there are verbs ending in *s*, *sh*, or *x*, which require their own treatment. The more difficult group of verbs to implement are such verbs, where the consonant is reduplicated in some forms.

Below we show how each verb type is implemented in the finite state lexicon. First, I will describe the structure of the lexicon and the structure of lexical entries. The example verbs are act, *assume*, *hurry*, and *confess*.

(1)
```
LEXICON Verb
VerbStem " V";

LEXICON VerbStem
act V "= vt vi";
assum V-e "= vi";
hurr V-y "= vt";
confess V-ss "= vt vi";
```

We see above that each sub-lexicon has the reserved word LEXICON, and after it is the name of the sub-lexicon. Under the lexicon name there is the list of lexical entries. In the first sub-lexicon there is only one entry. Normally each entry has three sections.

The first section is the lexical content of the word.

The second section is the address (that is, the name of a sub-lexicon), to which the process can proceed. If the address is #, the word-form is interpreted to be ready and the string is accepted as a valid word.

The third section is surrounded by double quotes. This section contains the analysis tags. In this section is also described the lemma of the word. If and equal mark = is typed, the string in the first section is copied. If the copy method cannot be used, then the lemma form must be typed immediately after the first double quot.

Also other alternatives for writing the entry exist. The first and third slot can be left out. Only the second slot is compulsory. If the first slot is left out, as in the first sub-lexicon, the first word in the entry is interpreted as an address.

Each entry must end in a semicolon.

We see in the second sub-lexicon that each entry has a different address. Below we see each of the four sub-lexicons and what they contain.

(2)

```
LEXICON V
 # " INF";
 # " PRES SG1/SG2";
 # " PRES PL1/PL2/PL3";
s # " PRES SG3";
ed # " PAST/EN";
ed # "= A ";
ing # " ING";
ing # "= N/A SG";
ings # "ing N PL";
```

The verb *act* may have all the forms and interpretations described in the sub-lexicon above. The first three entries have no suffixes, and also their lemma is the one described earlier. The second and third entry are underspecified. The fourth entry corresponds to the form *acts*, and also its lemma remains the same as it was defined earlier. The fifth entry corresponds to the form *acted*, and it is underspecified. The sixth entry has the same surface form as the fifth one. It is here listed separately, because its lemma form is different. The gerund suffix *ing* is also listed twice, so that the correct lemma can be produced in each case. If the gerund form has a plural suffix, it must be a noun, and not an adjective. Therefore, it is listed separately.

Next, we will make a test with all the forms of the verb *act* (3).

(3)

```
"<act>"
        "act" N SG
        "act" V vt vi INF
        "act" V vt vi PRES SG1/SG2
        "act" V vt vi PRES PL1/PL2/PL3
"<acts>"
        "act" N PL
        "act" V vt vi PRES SG3
"<acted>"
        "act" V vt vi PAST/EN
        "acted" V vt vi A
```

```
"<acting>"
        "act" V vt vi ING
        "acting" V vt vi N/A SG
"<actings>"
        "acting" V vt vi N PL
```

We see that each form was analysed, and all required interpretations were given. There are several underspecified readings, which must be further processed. There are also readings, which have more than one POS tag. Readings, which are obviously nouns or adjectives, have tags on verb transitivity. These tags must be removed.

In the post-processing phase, we modify the readings as shown in (4).

(4)

```
"<act>"
        "act" N SG
        "act" V vt vi INF
        "act" V vt vi PRES SG1
        "act" V vt vi PRES SG2
        "act" V vt vi PRES PL1
        "act" V vt vi PRES PL2
        "act" V vt vi PRES PL3
"<acts>"
        "act" N PL
        "act" V vt vi PRES SG3
"<acted>"
        "act" V vt vi PAST
        "act" V vt vi EN
        "acted" A
"<acting>"
        "act" V vt vi ING
        "acting" N SG
        "acting" A
"<actings>"
        "acting" N PL
```

Now the readings in each cohort are fully specified. If the form is interpreted as a verb, it has transitivity tags. Also, each person tag has a separate entry. Each entry has also the correct lemma. The words *act* and *acts* are also interpreted as nouns, because the word *act* is listed in the noun lexicon.

Next, we take a look at the verb *assume*, which ends in the vowel *e*. The sub-lexicon for endings of this verb type is in (5).

(5)

```
LEXICON V-e
e # "= INF";
e # "= PRES SG1/SG2";
```

```
e # "= PRES PL1/PL2/PL3";
es # "e PRES SG3";
ed # "e PAST/EN";
ed # "= A ";
ing # "e ING";
ing # "= N/A SG";
ings # "ing N PL";
```

We must drop the final vowel of the verb in the stem lexicon, because the gerund form does not have it. Therefore, the entries in the suffix lexicon ere different compared with the lexicon in (2). When we analyse the forms of the verb *assume*, we get the result as in (6).

(6)

```
"<assume>"
        "assume" V vt INF
        "assume" V vt PRES SG1/SG2
        "assume" V vt PRES PL1/PL2/PL3
"<assumes>"
        "assume" V vt PRES SG3
"<assumed>"
        "assume" V vt PAST/EN
        "assumed" V vt A
"<assuming>"
        "assume" V vt ING
        "assuming" V vt N/A SG
"<assumings>"
        "assuming" V vt N PL
```

After this basic analysis we modify the result further (7).

(7)

```
"<assume>"
        "assume" V vt INF
        "assume" V vt PRES SG1
        "assume" V vt PRES SG2
        "assume" V vt PRES PL1
        "assume" V vt PRES PL2
        "assume" V vt PRES PL3
"<assumes>"
        "assume" V vt PRES SG3
"<assumed>"
        "assume" V vt PAST
        "assume" V vt EN
        "assumed" A
"<assuming>"
```

```
        "assume" V vt ING
        "assuming" N SG
        "assuming" A
"<assumings>"
        "assuming" N PL
```

We see that the result is similar to that in (4). The difference is that the basic form does not have the noun interpretation, because the noun is *assumption*.

The next verb type is the one that ends in *y*, such as *hurry*. The inflection lexicon of this verb type is in (8).

(8)

```
LEXICON V-y
y # "= INF";
y # "= PRES SG1/SG2";
y # "= PRES PL1/PL2/PL3";
ies # "y PRES SG3";
ied # "y PAST/EN";
ied # "= A ";
ying # "y ING";
ying # "= N/A SG";
yings # "ying N PL";
```

The final *y* is dropped in the stem lexicon, because some forms do not have it. The correct endings are listed in the inflection lexicon. The basic analysis of the verb forms is in (9).

(9)

```
"<hurry>"
        "hurry" N SG
        "hurry" V vi INF
        "hurry" V vi PRES SG1/SG2
        "hurry" V vi PRES PL1/PL2/PL3
"<hurries>"
        "hurry" N PL
        "hurry" V vi PRES SG3
"<hurried>"
        "hurried" A
        "hurry" V vi PAST/EN
        "hurried" V vi A
"<hurrying>"
        "hurry" V vi ING
        "hurrying" V vi N/A SG
"<hurryings>"
        "hurrying" V vi N PL
```

Further processing cleans the underspecified readings as in (10).

(10)
```
"<hurry>"
        "hurry" N SG
        "hurry" V vi INF
        "hurry" V vi PRES SG1
        "hurry" V vi PRES SG2
        "hurry" V vi PRES PL1
        "hurry" V vi PRES PL2
        "hurry" V vi PRES PL3
"<hurries>"
        "hurry" N PL
        "hurry" V vi PRES SG3
"<hurried>"
        "hurried" A
        "hurry" V vi PAST
        "hurry" V vi EN
        "hurried" A
"<hurrying>"
        "hurry" V vi ING
        "hurrying" N SG
        "hurrying" A
"<hurryings>"
        "hurrying" N PL
```

Also here the base form *hurry* is interpreted also as a noun, because it is listed in the noun lexicon.

The fourth verb type is the one, which ends in *ss*, *s*, *sh*, or *x*. Its inflection lexicon is in (11).

(11)
```
LEXICON V-ss
 # " INF";
 # " PRES SG1/SG2";
 # " PRES PL1/PL2/PL3";
es # " PRES SG3";
ed # " PAST/EN";
ed # "= A ";
ing # " ING";
ing # "= N/A SG";
ings # "ing N PL";
```

This lexicon is very similar to the one in (2). The only difference is that here the form of present tense singular has the suffix *es*, while in the former one it is simple *s*.

One could consider joining these two sub-lexicons together and handle the deviant forms in the post-processing phase. This is, however, prone to mistakes, and it is safer to implement in the way as is shown here.

The basic analysis of the verb forms is in (12).

(12)
```
"<confess>"
        "confess" V vt vi INF
        "confess" V vt vi PRES SG1/SG2
        "confess" V vt vi PRES PL1/PL2/PL3
"<confesses>"
        "confess" V vt vi PRES SG3
"<confessed>"
        "confess" V vt vi PAST/EN
        "confessed" V vt vi A
"<confessing>"
        "confess" V vt vi ING
        "confessing" V vt vi N/A SG
"<confessings>"
        "confessing" V vt vi N PL
```

The further processing produces the specified readings (13).

(13)
```
"<confess>"
        "confess" V vt vi INF
        "confess" V vt vi PRES SG1
        "confess" V vt vi PRES SG2
        "confess" V vt vi PRES PL1
        "confess" V vt vi PRES PL2
        "confess" V vt vi PRES PL3
"<confesses>"
        "confess" V vt vi PRES SG3
"<confessed>"
        "confess" V vt vi PAST
        "confess" V vt vi EN
        "confessed" A
"<confessing>"
        "confess" V vt vi ING
        "confessing" N SG
        "confessing" A
"<confessings>"
        "confessing" N PL
```

The base form does not have a noun interpretation, because the noun form is *confession*.

2.2 Verbs with consonant duplication

Such verbs, which duplicate the final consonant in some forms, are a more difficult case, because there are several consonant types, which are duplicated. One solution would be that for each type a separate inflection lexicon is constructed. A more elegant solution is that these verbs are listed twice. One type allows only the basic inflection, and the other type allows only the duplicated inflection. The inflection lexicons are constructed so that each one has only the permitted forms.

Let us take three verbs with duplicating inflection, *drag*, *emit*, and *map*. In the verb stem lexicon, they have the entries as in (14).

(14)
```
dragg V-dub "= ";
emitt V-dub "= ";
mapp V-dub  "= ";
---
drag V-s "= ";
emit V-s "= ";
map V-s  "= ";
```

If the verb form has doubled the final consonant, it is directed to the sub-lexicon *V-dub*. In other cases, it is directed to the sub-lexicon *V-s*.

The basic analysis of all forms of the three verbs is in (15).

(15)
```
"<drag>"
        "drag" V vt INF
        "drag" V vt PRES SG1/SG2
        "drag" V vt PRES PL1/PL2/PL3
"<drags>"
        "drag" V vt PRES SG3
"<dragged>"
        "dragg" V vt DUP PAST/EN
        "dragged" V vt A
"<dragging>"
        "dragg" V vt DUP ING
        "dragging" V vt N/A SG
"<draggings>"
        "dragging" V vt N PL

"<emit>"
        "emit" V vt INF
        "emit" V vt PRES SG1/SG2
        "emit" V vt PRES PL1/PL2/PL3
"<emits>"
        "emit" V vt PRES SG3
```

```
"<emitted>"
        "emitt" V vt DUP PAST/EN
        "emitted" V vt A
"<emitting>"
        "emitt" V vt DUP ING
        "emitting" V vt N/A SG
"<emittings>"
        "emitting" V vt N PL

"<map>"
        "map" N SG
        "map" V INF
        "map" V PRES SG1/SG2
        "map" V PRES PL1/PL2/PL3
"<maps>"
        "map" N PL
        "map" V PRES SG3
"<mapped>"
        "mapp" V vt DUP PAST/EN
        "mapped" V vt A
"<mapping>"
        "mapp" V vt DUP ING
        "mapping" V vt N/A SG
"<mappings>"
        "mapping" V vt N PL
```

We see that most of the forms are correct. The problem is that in some readings the lemma form has the double consonant. In order to correct this problem, we have added the tag DUP to such readings, where correction is needed, and the correction is done later.

We see this when we proceed with the process (16).

(16)

```
"<drag>"
        "drag" V vt INF
        "drag" V vt PRES SG1
        "drag" V vt PRES SG2
        "drag" V vt PRES PL1
        "drag" V vt PRES PL2
        "drag" V vt PRES PL3
"<drags>"
        "drag" V vt PRES SG3
"<dragged>"
        "drag" V vt DUP PAST
        "drag" V vt DUP EN
        "dragged" A
```

```
"<dragging>"
        "drag" V vt DUP ING
        "dragging" N  SG
        "dragging" A
"<draggings>"
        "dragging" N PL

"<emit>"
        "emit" V vt INF
        "emit" V vt PRES SG1
        "emit" V vt PRES SG2
        "emit" V vt PRES PL1
        "emit" V vt PRES PL2
        "emit" V vt PRES PL3
"<emits>"
        "emit" V vt PRES SG3
"<emitted>"
        "emit" V vt DUP PAST
        "emit" V vt DUP EN
        "emitted" A
"<emitting>"
        "emit" V vt DUP ING
        "emitting" N  SG
        "emitting" A
"<emittings>"
        "emitting" N PL

"<map>"
        "map" N SG
        "map" V INF
        "map" V PRES SG1
        "map" V PRES SG2
        "map" V PRES PL1
        "map" V PRES PL2
        "map" V PRES PL3
"<maps>"
        "map" N PL
        "map" V PRES SG3
"<mapped>"
        "map" V vt DUP PAST
        "map" V vt DUP EN
        "mapped" A
"<mapping>"
        "map" V vt DUP ING
        "mapping" N  SG
        "mapping" A
```

```
"<mappings>"
        "mapping" N PL
```

We see that now all the forms are correct. The word map has also the noun interpretation, while the other two base forms do not.

When we describe the verbs with consonant duplication in this way, we need only two inflection lexicons.

2.3 Irregular verbs

Irregular verbs require such a structure that part of the verb inflection is described directly in the stem lexicon, and some forms are directed to the corresponding inflection lexicons. In (17) is an example of the verb *draw*.

(17)
```
draw V-i "= vt vi ";
drew # "draw vt vi PAST";
drawn # "draw vt vi EN";
```

We see that the deviant past and participial forms are listed as such, and the regular forms are directed to the sub-lexicon *V-i*. The sub-lexicon is in (18).

(18)
```
LEXICON V-i
 # " INF";
 # " PRES SG1/SG2";
 # " PRES PL1/PL2/PL3";
s # " PRES SG3";
ing # " ING";
ing # "= N/A SG";
ings # "ing N PL";
```

When we do the basic analysis of all the forms of the verb *draw*, the result is as in (19).

(19)
```
"<draw>"
        "draw" V vt vi INF
        "draw" V vt vi PRES SG1/SG2
        "draw" V vt vi PRES PL1/PL2/PL3
"<draws>"
        "draw" V vt vi PRES SG3
"<drew>"
        "draw" V vt vi PAST
"<drawn>"
        "draw" V vt vi EN
"<drawing>"
```

```
        "draw" V vt vi ING
        "drawing" V vt vi N/A SG
"<drawings>"
        "drawing" V vt vi N PL
```

The further processing brings a clean result (20).

(20)
```
"<draw>"
        "draw" V vt vi INF
        "draw" V vt vi PRES SG1
        "draw" V vt vi PRES SG2
        "draw" V vt vi PRES PL1
        "draw" V vt vi PRES PL2
        "draw" V vt vi PRES PL3
"<draws>"
        "draw" V vt vi PRES SG3
"<drew>"
        "draw" V vt vi PAST
"<drawn>"
        "draw" V vt vi EN
"<drawing>"
        "draw" V vt vi ING
        "drawing" N SG
        "drawing" A
"<drawings>"
        "drawing" N PL
```

The example (20) above was a consonant-final verb. Next, we take a look at the e-final irregular verbs, such as hide, lose, and awake. These kinds of verbs are listed in the stem lexicon as in (21).

(21)
```
hid V-e2 "= vt vi ";
hid # "hide vt vi PAST";
hidden # "hide vt vi EN";
hidden # "= A";

los V-e2 "= vt vi ";
lost # "lose vt vi PAST";
lost # "lose vt vi EN";
lost # "= A";

awak V-e2 "= vt vi ";
awoke # "awake vt vi PAST";
awoken # "awake vt vi EN";
```

awoken # "= A";

The analysis of all three verb stems is directed to the sub-lexicon *V-e2*, which is in (22).

(22)
```
LEXICON V-e2
e # "= INF";
e # "= PRES SG1/SG2";
e # "= PRES PL1/PL2/PL3";
es # "e PRES SG3";
ing # "e ING";
ing # "= N/A SG";
ings # "ing N PL";
```

When all forms of the three verbs are initially analysed, the result is a mixture of specified and underspecified readings (23).

(23)
```
"<hide>"
        "hide" V vt vi INF
        "hide" V vt vi PRES SG1/SG2
        "hide" V vt vi PRES PL1/PL2/PL3
"<hides>"
        "hide" V vt vi PRES SG3
"<hid>"
        "hide" V vt vi PAST
"<hidden>"
        "hide" V vt vi EN
        "hidden" V A
"<hiding>"
        "hide" V vt vi ING
        "hiding" V vt vi N/A SG
"<hidings>"
        "hiding" V vt vi N PL

"<lose>"
        "lose" V vt vi INF
        "lose" V vt vi PRES SG1/SG2
        "lose" V vt vi PRES PL1/PL2/PL3
"<loses>"
        "lose" V vt vi PRES SG3
"<lost>"
        "lose" V vt vi PAST
        "lose" V vt vi EN
        "lost" V A
"<lost>"
```

```
        "lose" V vt vi PAST
        "lose" V vt vi EN
        "lost" V A
"<losing>"
        "lose" V vt vi ING
        "losing" V vt vi N/A SG
"<losings>"
        "losing" V vt vi N PL

"<awake>"
        "awake" V vt vi INF
        "awake" V vt vi PRES SG1/SG2
        "awake" V vt vi PRES PL1/PL2/PL3
"<awakes>"
        "awake" V vt vi PRES SG3
"<awoke>"
        "awake" V vt vi PAST
"<awoken>"
        "awake" V vt vi EN
        "awoken" V A
"<awaking>"
        "awake" V vt vi ING
        "awaking" V vt vi N/A SG
"<awakings>"
        "awaking" V vt vi N PL
```

We see that all readings have the verb tag V, which was added to the output already before the process was directed to the stem lexicon. If the reading has also another POS tag, such as A or N, this will be retained, and the verb tag removed. This can be done, however, only after we have rewritten the underspecified readings into non-ambiguous readings (24).

(24)

```
"<hide>"
        "hide" V vt vi INF
        "hide" V vt vi PRES SG1
        "hide" V vt vi PRES SG2
        "hide" V vt vi PRES PL1
        "hide" V vt vi PRES PL2
        "hide" V vt vi PRES PL3
"<hides>"
        "hide" V vt vi PRES SG3
"<hid>"
        "hide" V vt vi PAST
"<hidden>"
        "hide" V vt vi EN
```

```
        "hidden" A
"<hiding>"
        "hide" V vt vi ING
        "hiding" N SG
        "hiding" A
"<hidings>"
        "hiding" N PL

"<lose>"
        "lose" V vt vi INF
        "lose" V vt vi PRES SG1
        "lose" V vt vi PRES SG2
        "lose" V vt vi PRES PL1
        "lose" V vt vi PRES PL2
        "lose" V vt vi PRES PL3
"<loses>"
        "lose" V vt vi PRES SG3
"<lost>"
        "lose" V vt vi PAST
        "lose" V vt vi EN
        "lost" A
"<lost>"
        "lose" V vt vi PAST
        "lose" V vt vi EN
        "lost" A
"<losing>"
        "lose" V vt vi ING
        "losing" N SG
        "losing" A
"<losings>"
        "losing" N PL

"<awake>"
        "awake" V vt vi INF
        "awake" V vt vi PRES SG1
        "awake" V vt vi PRES SG2
        "awake" V vt vi PRES PL1
        "awake" V vt vi PRES PL2
        "awake" V vt vi PRES PL3
"<awakes>"
        "awake" V vt vi PRES SG3
"<awoke>"
        "awake" V vt vi PAST
"<awoken>"
        "awake" V vt vi EN
        "awoken" A
```

```
"<awaking>"
        "awake" V vt vi ING
        "awaking" N SG
        "awaking" A
"<awakings>"
        "awaking" N PL
```

Each reading is now correctly formatted, and the result can be sent to the disambiguation process.

3 Discussion

We have seen that it is possible to handle also nouns and adjectives derived from verbs directly in the verb stem lexicon, without the need to list them separately into the noun lexicon and adjective lexicon. The method reduces the size of the total lexicon by more than 10000 lines. At the same time, it also simplifies the maintenance of the system, because a verb and the other POS categories derived from it need to be edited only in one place.

Although the solution adds ambiguity to readings, the added ambiguity is useful in disambiguation, especially when also person tags are added into relevant places. Such tags are useful in machine translation to languages, where each person has a different form.

The method allows also such derived forms, which probably never occur in text. If the forms are grammatically acceptable in the language, they should be allowed. In case some forms cause unnecessary ambiguity, such cases can be handled individually. My view is that the advantages of the proposed method are vastly bigger than its possible disadvantages.