

Gebze Technical University
Department of Computer Engineering
CSE 654 Natural Language Processing
HW1 Report

H.Yavuz ERZURUMLU
181041004

Abstract

In this project, I will design and implement a **finite state transducer(FST)** using **foma** compiler. The fst must handle nouns and accusatives. In Turkish grammar, words ending in -P -Ç -T -K change to -B -C -D -Ğ when suffixed with a vowel.

You can access the whole source code and paper which used in this report from:

- <https://github.com/freeloki/CSE654-NLP-Homework01>

1 Introduction

The foma compiler is essentially a tool for converting regular expressions to finite automata and transducers. It supports a variety of operations (many more than are found in search-regex formalisms such as the Python re module). The interface also includes tools for performing various tests on automata and transducers, passing words through transducers (doing translations), and importing and exporting transducers in various formats.

2 Turkish Consonant Mutation

Nouns ending hard Un-Voiced Consonants **P, Ç, T, K** mutate to voiced **B, C, D, Ğ, Ğ** when a vowel suffix is added.

2.1 Examples of Turkish Consonant Changes

- kitap (book) → kitabın (your book)
- öğüt (advice) → öğüdüm (my advice)
- tat (taste) → tadı (its taste)
- ilaç (medicine) → ilacı (his medicine)
- ağaç (tree) → ağacın (the tree's)

3 The foma FST compiler

The *foma* program runs with a read-eval-print loop (REPL), like IPython/Jupyter. That means that each command given is executed and the output is printed, and a new prompt is displayed. You can run scripts of commands by either launching foma with the foma -l flag, or by typing source filename inside foma.

```

codemania@Codegenius-G550JK /mnt/Linux-Extended/DEVELOPMENT/GIT/MASTER/CSE654-NLP-Homework01 $ foma
Foma, version 0.9.18alpha
Copyright © 2008-2014 Mans Hulden
This is free software; see the source code for copying conditions.
There is ABSOLUTELY NO WARRANTY; for details, type "help license"

Type "help" to list all commands available.
Type "help <topic>" or help "<operator>" for further help.
of commands by either launching foma with the foma -l flag, or by typing source filename inside foma.

foma[0]: 
  63
  64
  65- lexicon[Theorph]
  66
  67 Theorph, a two-level morphological analyzer for Turkish. Theorph is a fairly complete and accurate
  morphological analyzer for Turkish. However, strength of Theorph is neither in its performance, nor in its
  novelty. The main feature of this analyzer is its availability. It has completely been implemented using
  freely available tools and resources, and the two-level description is also distributed with a license that
  allows others to use and modify it freely for different applications. To our knowledge, Theorph is the first
  freely available morphological analyzer for Turkish. This makes Theorph particularly suitable for
  applications where the analyzer has to be changed in some way, or as a starting point for morphological
  analyzers for similar languages. Theorph's specification of Turkish morphology is relatively complete, and
  it is distributed with a large lexicon. Along with the description of how the analyzer is implemented, this
  paper provides an evaluation of the analyzer on two large corpora.
  68
  69
  70- lexicon[Lexicon]
  71
  72
  73- lexicon[Accusative Noun ( Isin -l Hall )]
  74
  75
  76- lexicon[Genitive - Dative ( Isin -e Hall - Yemeye Hall)]
  77
  78
  79- lexicon[Deriving Rule]
  80
  81
  82- lexicon[Samples]
  83
  84- lexicon[Read()]]
  85
  86- lexicon[Print()]]
  87

```

Figure 1: Foma tool.

4 TRmorph[1]

TRmorph[2], a two-level morphological analyzer for Turkish. TRmorph is a fairly complete and accurate morphological analyzer for Turkish. However, strength of TRmorph is neither in its performance, nor in its novelty. The main feature of this analyzer is its availability. It has completely been implemented using freely available tools and resources, and the two-level description is also distributed with a license that allows others to use and modify it freely for different applications. To our knowledge, TRmorph is the first freely available morphological analyzer for Turkish. This makes TRmorph particularly suitable for applications where the analyzer has to be changed in some way, or as a starting point for morphological analyzers for similar languages. TRmorph's specification of Turkish morphology is relatively complete, and it is distributed with a large lexicon. Along with the description of how the analyzer is implemented, this paper provides an evaluation of the analyzer on two large corpora.

4.1 Lexicon

I have used all nouns in TRmorph's lexicon file. That's why I have 12189 states.

```

codemania@Codegenius-G550JK /mnt/Linux-Extended/DEVELOPMENT/GIT/MASTER/CSE654-NLP-Homework01 $ foma
Foma, version 0.9.18alpha
Copyright © 2008-2014 Mans Hulden
This is free software; see the source code for copying conditions.
There is ABSOLUTELY NO WARRANTY; for details, type "help license"

Type "help" to list all commands available.
Type "help <topic>" or help "<operator>" for further help.

foma[0]: regex @"yavuz.fst";
435.6 kB, 12189 states, 27640 arcs, 47223 paths.
foma[1]: up ağacı
ağaç<N><acc>
foma[1]: 

```

Figure 2: FST State

```

Multichar_Symbols ^I
                  ^(^s) ^(^y)
                  @RB @MB
                  @NCOMP

LEXICON N
%<N%>:@RB      Ncont;

LEXICON Ncont
  NcompCont; ! includes final

LEXICON NcompCont
  ENDLEX;
  NCaseNoGen;

LEXICON NCaseNoGen
%<acc%>:^(y)^I@MB  ENDLEX;
%<dat%>:^(y)^A@MB  ENDLEX;

LEXICON ENDLEX #;

```

Figure 3: Lexical States

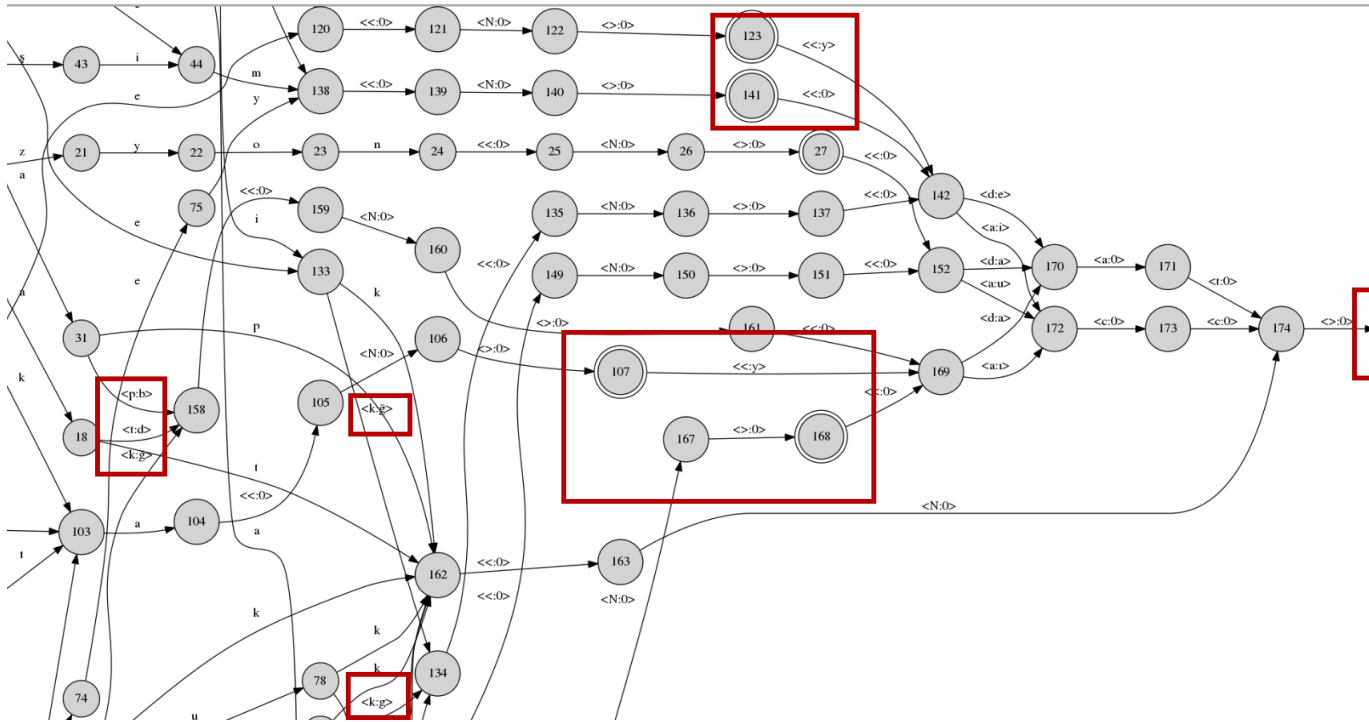


Figure 4: Dependency Graph

4.2 Accusative Noun (İsmi -i Hali

This simple regex handles accusatives:
 %<acc%>:^(y)^I@MB ENDLEX;

```

Foma, version 0.9.18alpha
Copyright © 2008-2014 Mans Hulden
This is free software; see the source code for copying conditions.
There is ABSOLUTELY NO WARRANTY; for details, type "help license"

Type "help" to list all commands available.
Type "help <topic>" or help "<operator>" for further help.

foma[0]: regex @"yavuz.fst";
435.6 kB. 12189 states, 27640 arcs, 47223 paths.
foma[1]: up ağacı
ağaç<N><acc>
foma[1]: down ağaç<N><acc>
ağacı
foma[1]:

```

Figure 5: Accusative with devoicing

```

foma[1]:
foma[1]: up aşk1
aşk<N><acc>
foma[1]: down aşk<N><acc>
aşk1
foma[1]:

```

Figure 6: Accusative without devoicing

4.3 Bonus - Dative (İsmi -e Hali - Yönelme Hali)

Using same method we can handle datives as well:

%<dat%>:(y)Â@MB ENDLEX;

```

foma[1]: up kağıda
kağıt<N><dat>
foma[1]:
foma[1]: down kağıt<N><dat>
kağıda
foma[1]:

```

Figure 7: Dative with devoicing

```

foma[1]:
foma[1]: up kaleme
kalem<N><dat>
foma[1]: down kalem<N><dat>
kaleme
foma[1]:
foma[1]:
foma[1]:

```

Figure 8: Dative without devoicing

4.4 Devoicing Rule

```
define FSDevoicing [%^c -> c,  
                    %^g -> ğ,  
                    %^K (->) ğ,  
                    %^K -> y,  
                    %^p -> b,  
                    %^t -> d || _ IntSym* [ Vowel | Vxx ] ]  
.o. %^k -> ğ || \n _ IntSym* [ Vowel | Vxx ]  
.o. %^k -> g || n _ IntSym* [ Vowel | Vxx ]  
.o. [%^c -> ç,  
    %^g -> g,  
    %^K -> k,  
    %^p -> p,  
    %^t -> t,  
    %^k -> k];
```

Figure 9: Devoicing Rule

4.5 Samples

```
???  
apply up> asidi  
asit<N><acc>  
apply up> aşkı  
aşk<N><acc>  
apply up> bayrağı  
bayrak<N><acc>  
apply up> bilişimi  
bilişim<N><acc>  
apply up> çadırı  
çadır<N><acc>  
apply up> camı  
cam<N><acc>  
apply up> çarığı  
çarık<N><acc>  
apply up> cengi  
cenk<N><acc>  
apply up> çırağı  
çırak<N><acc>  
apply up> denklemleri  
denklemler<N><acc>  
apply up> dolabı  
dolap<N><acc>  
apply up> esasları  
esaslar<N><acc>  
apply up> evladı  
evlat<N><acc>  
apply up> fayı  
fay<N><acc>  
apply up> fıncığı  
fındık<N><acc>  
apply up> gamı  
gam<N><acc>  
apply up> gömleği  
gömlek<N><acc>  
apply up> hızı  
hız<N><acc>  
apply up> hududu  
hudut<N><acc>  
apply up> icatları  
icatlar<N><acc>  
???  
apply up> icadları  
icadlar<N><acc>  
apply up> ırkı  
ırk<N><acc>  
apply up> ışığı  
ışık<N><acc>  
apply up> işlemi  
işlem<N><acc>  
apply up> jeli  
jel<N><acc>  
apply up> jeofiziği  
jeofizik<N><acc>  
apply up> kaçı  
kaçık<N><acc>  
apply up> kuzeyi  
kuzey<N><acc>  
apply up> leyleği  
leylek<N><acc>
```

Figure 10: Samples 1

```

111
apply up> maden ELY NO WARRANTY; for detail
maden<N>
apply up> mantık st all commands available
mantık<N> "topic" or help "<operator>" for
apply up> nam
nam<N> source yavuz analyzer.xfst
apply up> madeni uz analyzer.xfst
maden<N><acc> 15928, N...1, Ncont...1, Nco
apply up> mantiğı
mantık<N><acc> on 'Ncomp' used but never
apply up> nohudu
nohut<N><acc>
apply up> namı
nam<N><acc> 116 states, 26323 arcs, 47271 p
apply up> oğlağı 413.2 kB, 11676 states, 26
oğlak<N><acc> morph-phon-yavuz.xfst
apply up> ocağı bytes, 2 states, 12 arcs, 1
ocak<N><acc> 1 bytes, 2 states, 14 arcs, 1
apply up> önayağı tes, 2 states, 16 arcs, 1
önayak<N><acc> bytes, 2 states, 14 arcs, 1
apply up> paniğı bytes, 2 states, 5 arcs, 5
panik<N><acc> 1 bytes, 2 states, 7 arcs, 7
apply up> patiyi bytes, 2 states, 7 arcs, 7
pati<N><acc> 1 bytes, 2 states, 7 arcs, 7
apply up> diski 3 kB, 2 states, 26 arcs, 2
??? 193 bytes, 2 states, 2 arcs,
apply up> riski 9 bytes, 2 states, 8 arcs,
risk<N><acc> 89 bytes, 2 states, 2 arcs, 2
apply up> ruhsatı 87 bytes, 2 states, 14 arc
ruhsat<N><acc> 1 kB, 2 states, 36 arcs, 36
apply up> sapağı 5 kB, 2 states, 31 arcs, 3
sapak<N><acc> 124 bytes, 2 states, 17 arc
apply up> sarayı 1 kB, 2 states, 46 arcs, 46
saray<N><acc> 384 bytes, 2 states, 4 arc
apply up> şebeğı 91 bytes, 2 states, 9 arcs
şebek<N><acc> 115, 293 bytes, 2 states, 2
apply up> sonbahar 5 kB, 2 states, 55 arcs,
sonbahar<N> 115, 335 bytes, 2 states, 3 ar
apply up> sonbaharı bytes, 2 states, 9 arcs
sonbahar<N><acc> 12.5 kB, 2 states, 55 ar
apply up> terliğı 4.2 kB, 2 states, 138 a
terlik<N><acc> 2.6 kB, 2 states, 80 arc
apply up> timsahı er, 7.5 kB, 3 states, 297
timsah<N><acc> 187 bytes, 1 state, 17 arcs,
apply up> üstadı ony, 3.6 kB, 4 states, 152
üstat<N><acc> 11mg, 5.7 kB, 5 states, 256
apply up> uyağı er, 5.4 kB, 24 states, 297
uyak<N><acc> 320.5 kB, 195 states, 2028
apply up> vadeyi symbols, 1.5 kB, 10 states
vade<N><acc> boundary, 378 bytes, 1 state,
apply up> varlığı 435.6 kB, 12189 states, 2
varlık<N><acc> 1 states, 27640 arcs, 47223 p
apply up> yapımı yavuz.fst
yapım<N><acc> 115
apply up> yüreğı
yürek<N><acc> 115ac<N><acc>
apply up> zambağı
zambak<N><acc>
apply up> zili 115mg<N><acc> 115mg<N><acc>
zil<N><acc> 115mg<N><acc> 115mg<N><acc>
apply up> on 8.9.18alpha
foma[1]: 2008-2014 Hans Holden

```

Figure 11: Samples 2

5 Result

TRmorph is very useful tool for Turkish morphological analyzer. I have implemented very simple version of it which only handles **N** , **Accusative**, **Dative** one. We can easily define and implement regular expressions via **foma**.

References

- [1] Çağrı Çöltekin (2010) A Freely Available Morphological Analyzer for Turkish. In: Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC 2010)
- [2] <https://github.com/coltekin/TRmorph> free morphological analyzer repo.
- [3] <https://fomafst.github.io>
- [4] <http://sureksiz-sert-unsuzlerin-yumusamasi.nedir.org/>