CLPL3b: Creating Morphological Analyzers

Tafseer Ahmed
May 2020

Plan

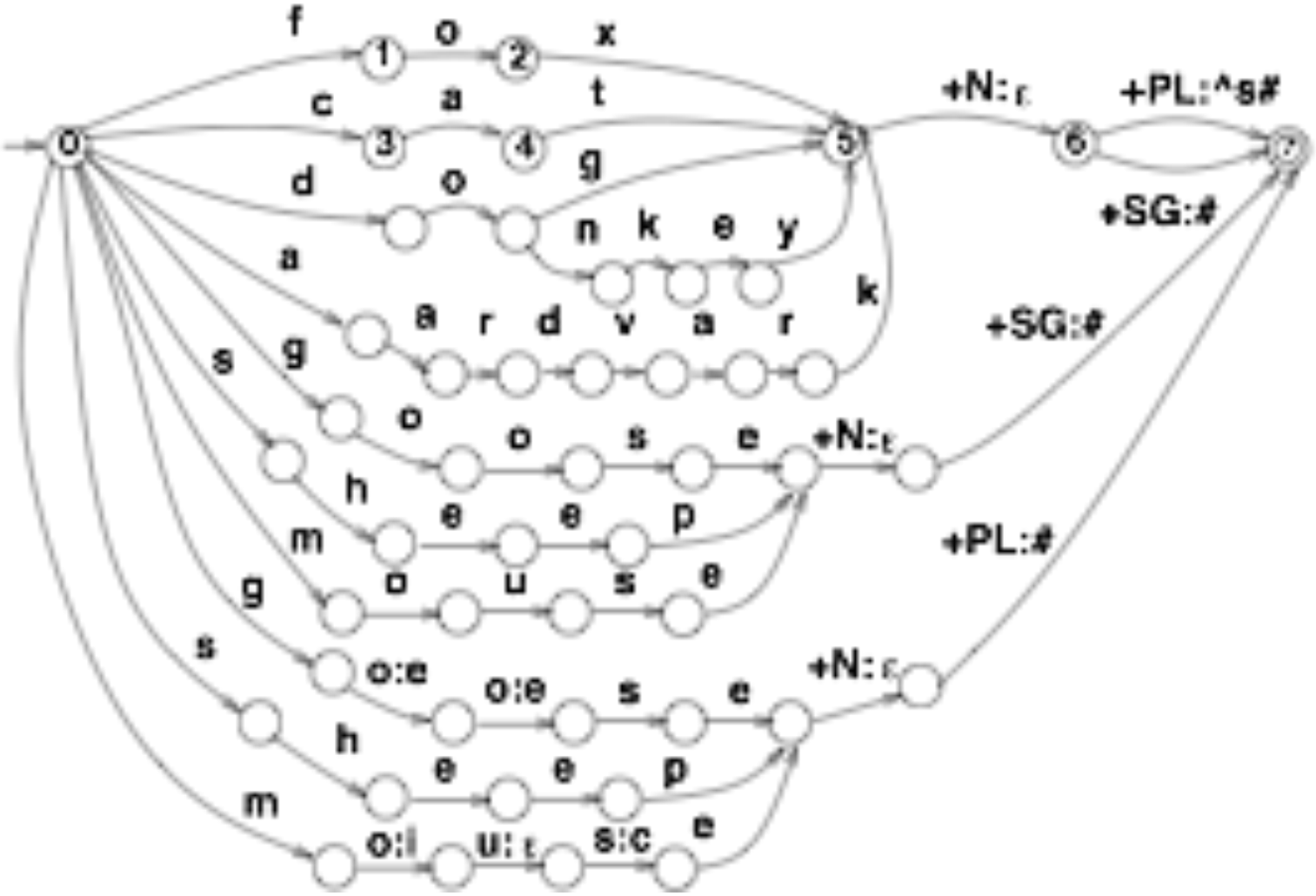
- Finite State Transducer
- How to use Foma
- Modeling Urdu Noun

Finite State Transduce

Why Finite State Approach?

- Finite state systems are mathematically well-understood, elegant, flexible.
- Finite state systems are computationally efficient.
- Finite state systems are inherently bidirectional.

A simple FST of nouns



Some Terms

- Lexical (upper) form
- Surface (down) form

Lexical:	s	u	n	+N	+PL
Surface:	s	u	n	s	

Lexical:	f	o	x	+N	+PL
Surface:	f	o	x	e	s

Two alternate approaches

There are two alternate approaches for the creation of FST for morphological analysis/generation.

- Parallel constraint approach
  - Two level morphology e.g. PC-Kimmo

- Sequential constraint approach
  - Cascade rules e.g. XFST and FOMA

Tools for sequential FST

- **XFST:** Xerox Finite State Transducer

  used by PARallel GRAMmmer (Pargram) project.

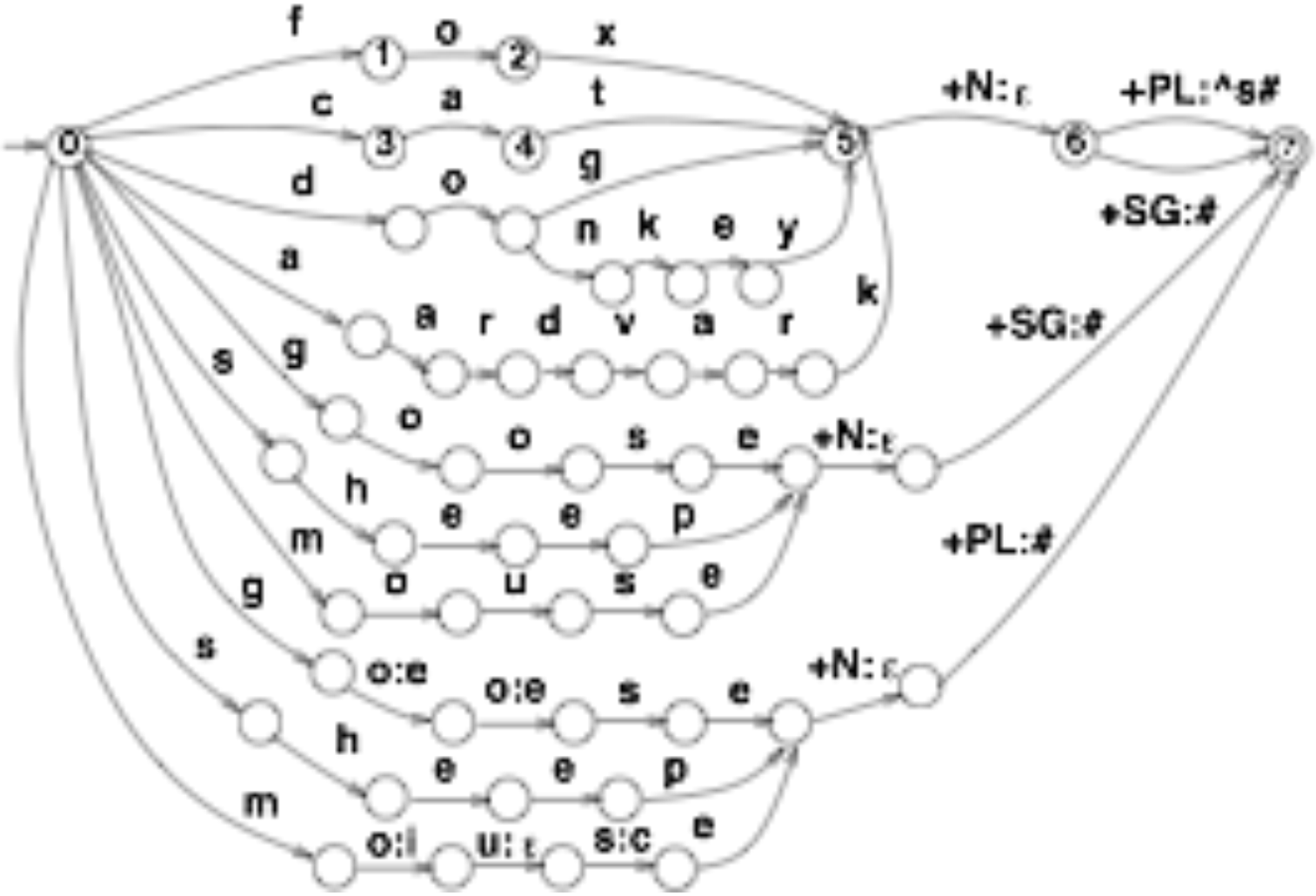
- **FOMA**: Open source equivalent of XFST

  https://fomafst.github.io/

  https://code.google.com/archive/p/foma/wikis/MorphologicalAnalysisTutorial.wiki

Foma

A simple FST of nouns

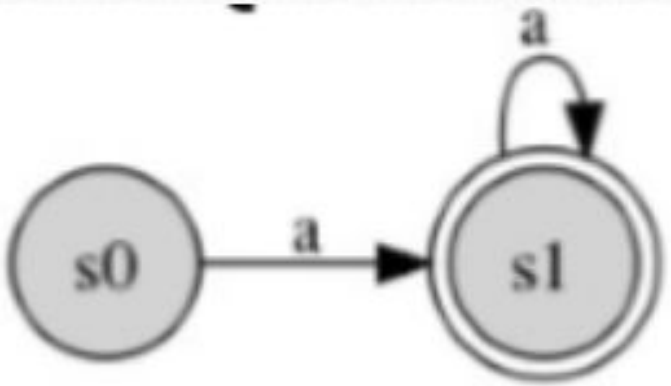


Foma Rules

- **Matching Rules**
- Replacement Rules

Regex (REGular EXpression)

- regex a+ ;
- down
- CTRL+D



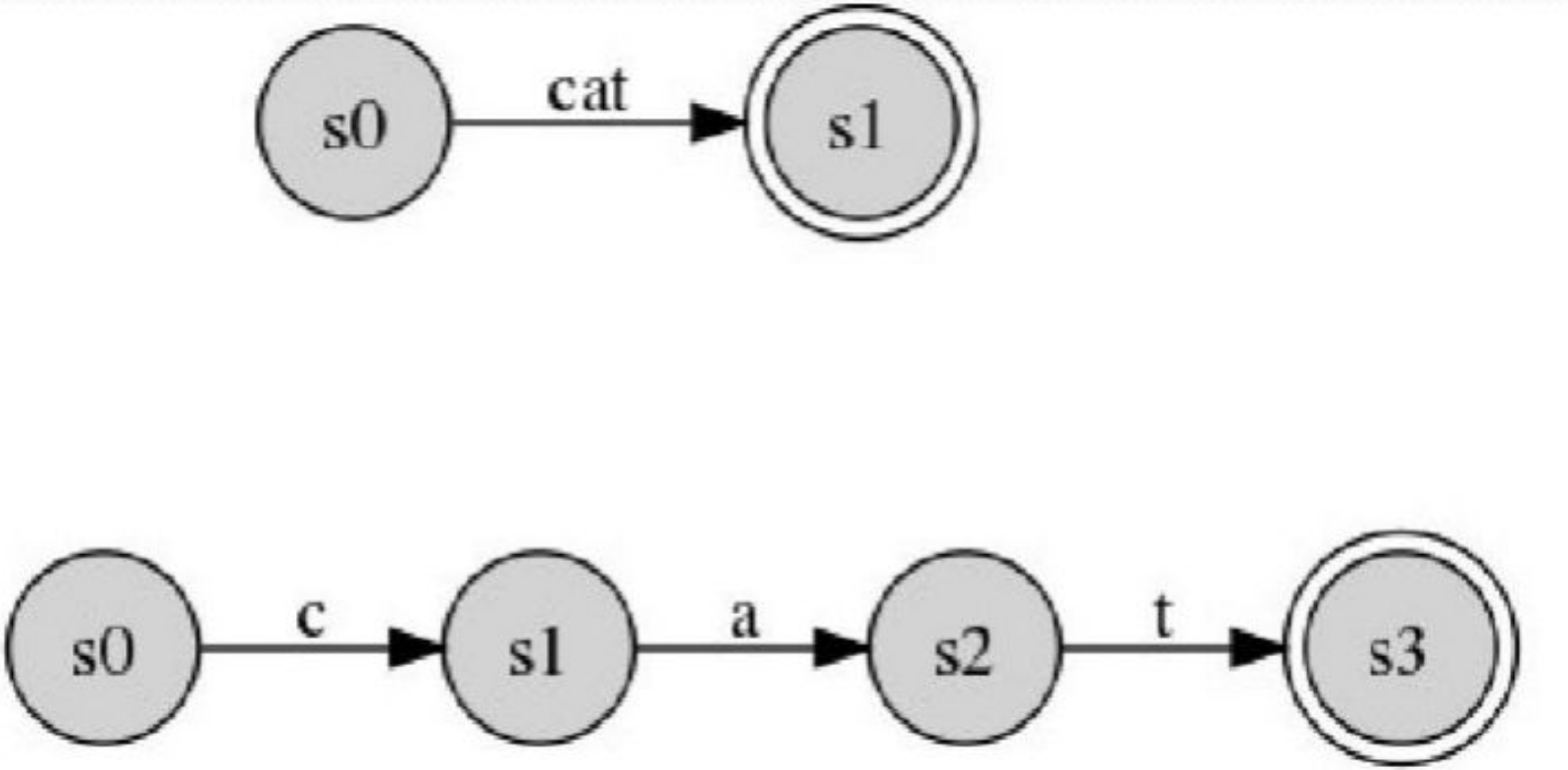
Matching Rules

- regex a* ;
- regex cat ;
- regex c a t ;
- regex c a [t|b] ;
- regex c a ? ;

```
foma[0]: regex a* ;
197 bytes. 1 state, 1 arc, Cyclic.
foma[1]: down aa
aa
foma[1]: down b
???
foma[1]:
foma[1]: regex cat ;
184 bytes. 2 states, 1 arc, 1 path.
foma[2]: down cat
cat
foma[2]: down cab
???
foma[2]:
foma[2]: regex c a t ;
273 bytes. 4 states, 3 arcs, 1 path.
foma[3]:
foma[3]: regex c a [t|b] ;
303 bytes. 4 states, 4 arcs, 2 paths.
foma[4]: down cat
cat
foma[4]: down cab
cab
foma[4]: down cap
???
foma[4]:
foma[4]: regex c a ? ;
323 bytes. 4 states, 5 arcs, 3 paths.
foma[5]: down cat
cat
foma[5]: down cap
cap
foma[5]: _
```

Matching Rules

cat vs c a t



Practice Questions

give Regex for the following

- The words start with *b* and end at *k*
- Four letter words that start with *b* or *c* and ends at *k*
- Days of the week

Practice Questions

```
foma[6]: regex b ?* k ;
339 bytes. 3 states, 7 arcs, Cyclic.
foma[7]: down book
book
foma[7]:
foma[7]:
foma[7]: regex [b|c] ? ? k ;
433 bytes. 5 states, 11 arcs, 32 paths.
foma[8]: down book
book
foma[8]: down cook
cook
foma[8]: down baeik
???
foma[8]:
foma[8]:
foma[8]: regex [ m o n | f r i | s u n ] d a y ;
513 bytes. 10 states, 11 arcs, 3 paths.
foma[9]: down monday
monday
foma[9]: print lowe
monday
friday
sunday
```

Matching Methods

- Regex

  regex .....

- Wordlist

  read text .....

- Lexc

  read lexc .....

read text filename

```
foma[12]: read text C:/fomacode/days.txt
606 bytes. 15 states, 17 arcs, 4 paths.
foma[13]: print lower-words
wednesday
sunday
monday
friday
```

Lexc (LEXical Compiler)

```
LEXICON Root
mon SUFF;
wednes SUFF;
fri SUFF;
sun SUFF;

LEXICON SUFF
day #;
```

```
foma[14]: read lexc C:/fomacode/days_c.txt
Root...4, SUFF...1
Building lexicon...
Determinizing...
Minimizing...
Done!
606 bytes. 15 states, 17 arcs, 4 paths.
foma[15]: print lower-words
sunday
friday
wednesday
monday
```

Lower and Upper Form

- regex c:k a:e t:T ;
  - down
  - up
- regex [c a t]:[k e T] ;
- regex [c a t]:[b i l l i];

Lower and Upper Form

```
foma[15]: regex [c a t]:[b i l l i] ;
347 bytes. 6 states, 5 arcs, 1 path.
foma[16]: down cat
billi
foma[16]: up billi
cat
foma[16]: up cat
???
```

Replacement Rules

- regex c -> k;
- regex c -> k || _ [a|o];

```
foma[16]: regex c -> k || _ [a|o] ;
478 bytes. 3 states, 12 arcs, Cyclic.
foma[17]: down catch
katch
foma[17]: down conference
konference
```

Combining the automata

- Compose
- Union
- Intersection
- Difference
- Priority Union
- ....

Modeling Urdu Noun

Urdu Nouns

Form/Case	Number	Gender	
Nominal	Singular	Masculine	لڑکا آیا
Nominal	Plural	Masculine	دو لڑکے آئے
Oblique	Singular	Masculine	لڑکے نے کہا
Oblique	Plural	Masculine	دو لڑکوں نے کہا

Urdu Nouns

Form/Case	Number	Gender	
Nominal	Singular	Masculine	سیب گرا
Nominal	Plural	Masculine	دو سیب گرے
Oblique	Singular	Masculine	سیب کو گرایا
Oblique	Plural	Masculine	دو سیبوں کو گرایا

Urdu Nouns

Case/Form	Number	Gender	
Nominal	Singular	Feminine	لڑکی آئی
Nominal	Plural	Feminine	دو لڑکیاں آئیں
Oblique	Singular	Feminine	لڑکی نے کہا
Oblique	Plural	Feminine	دو لڑکیوں نے کہا

Urdu Nouns

Case/Form	Number	Gender	
Nominal	Singular	Feminine	عورت آئی
Nominal	Plural	Feminine	دو عورتیں آئیں
Oblique	Singular	Feminine	عورت نے کہا
Oblique	Plural	Feminine	دو عورتوں نے کہا

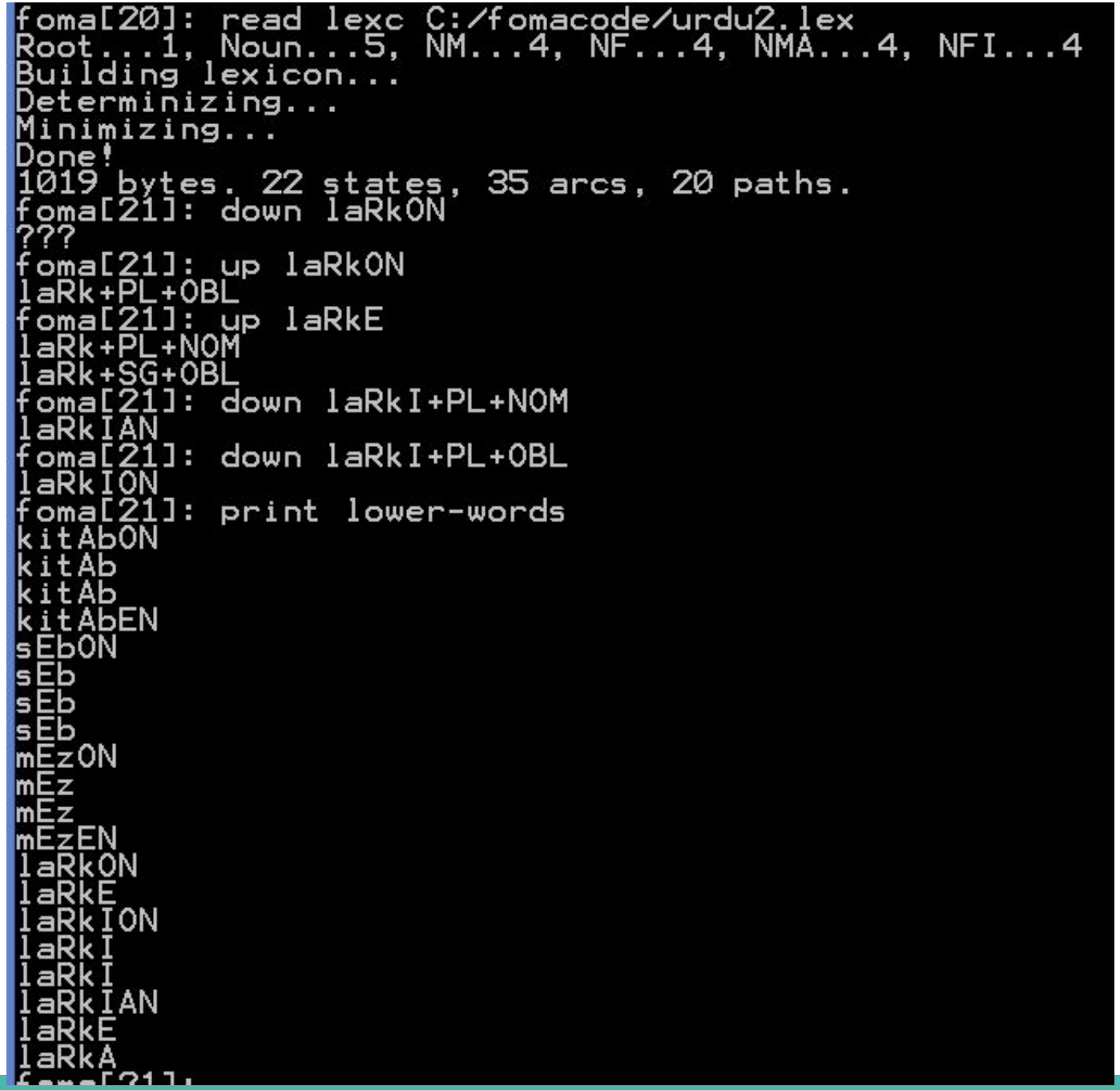
Lexc (LEXical Compiler)

```
LEXICON Root
mon SUFF;
wednes SUFF;
fri SUFF;
sun SUFF;

LEXICON SUFF
day #;
```

```
foma[14]: read lexc C:/fomacode/days_c.txt
Root...4, SUFF...1
Building lexicon...
Determinizing...
Minimizing...
Done!
606 bytes. 15 states, 17 arcs, 4 paths.
foma[15]: print lower-words
sunday
friday
wednesday
monday
```

Urdu Nouns

```
Multichar_Symbols +N +SG +PL +NOM +OBL

LEXICON Root
Noun ;

LEXICON Noun
laRk   NMA;
laRkI   NFI;
kitAb   NF;
mEz NF;
sEb   NM;

LEXICON NM
+SG+NOM:0    #;
+SG+OBL:0    #;
+PL+NOM:0    #;
+PL+OBL:ON   #;

LEXICON NF
+SG+NOM:0    #;
+SG+OBL:0    #;
+PL+NOM:EN   #;
+PL+OBL:ON   #;

LEXICON NMA
+SG+NOM:A    #;
```


foma[20]: read lexc C:/fomacode/urdu2.lex
Root...1, Noun...5, NM...4, NF...4, NMA...4, NFI...4
Building lexicon...
Determinizing...
Minimizing...
Done!
1019 bytes. 22 states, 35 arcs, 20 paths.
foma[21]: down laRkON
???
foma[21]: up laRkON
laRk+PL+OBL
foma[21]: up laRkE
laRk+PL+NOM
laRk+SG+OBL
foma[21]: down laRkI+PL+NOM
laRkIAN
foma[21]: down laRkI+PL+OBL
laRkION
foma[21]: print lower-words
kitAbON
kitAb
kitAb
kitAbEN
sEbON
sEb
sEb
sEb
mEzON
mEz
mEz
mEzEN
laRkON
laRkE
laRkION
laRkI
laRkI
laRkIAN
laRkE
laRkA

Urdu Nouns

- Using Urdu Script

```
foma[22]: read lexc C:/fomacode/urdu3.lex
ROOT...1, Noun...4, NM...4, NF...4, NMA...4, NFI...4
Building lexicon...
*Warning: no Root lexicon, using 'ROOT' as Root.
Determinizing...
Minimizing...
Done!
894 bytes. 18 states, 30 arcs, 16 paths.
foma[23]: print lower-words > c:/fomacode/urduwords.txt
Writing to c:/fomacode/urduwords.txt.
```

Non-Concatenative Morphology

Example from Arabic

Root	$C_1C_2C_3$		
Templates	$C_1AC_2iC_3$,	$mC_1C_2UC_3$	
Root	f?l	ktb	?lm
	فعل	كتب	علم
T1	fA?il	kAtib	?Alim
	فاعل	كاتب	عالم
T2	mf?Ul	mktUb	m?lUm

Soultions

- An Open-Source Finite State Morphological Transducer for Modern Standard Arabic, Mohammed Attia et al.,

  https://www.aclweb.org/anthology/W11-4417.pdf

- https://sourceforge.net/projects/aracomlex/