Normalization of dialects and variants using FST technology
Grammars and examples

Iñaki Alegria
(University of The Basque Country)

Overview

Basic ideas

Designing a rewrite-grammar

Compiling & testing a complete grammar

Examples

Some basic ideas

Rewrite rules: when optional?

→ When the changes can be produced or not

i.e. OCR

```
m (->) r n ;
```

→ NOT when changes are necessary

i.e. Portuguese/Galician

```
n h -> ñ ;
```

Some basic ideas II

Composition or parallel rules?

→ In composition (sequential rules, rewrite rules)
the order of the rules is important
and new changes are applied on the previous ones

→ Composition is more flexible, powerful and habitual

→ But sometimes parallel rules are more adequate
i.e. OCR

```
define PARAL0 [ m (->) r n , n (->) r i ] ;
define SEQ0   [ m (->) r n .o. n (->) r i ] ;
regex PARAL0 ;
down ama;  # result: ama/arna
regex SEQ0 ;
down ama;  # result: ama/arna/arria
```

First grammar: OCR

File: ***ocr/ocr_simple***

```
define PARAL  [ c (->) e , e (->) c , c (->) o ,
 o (->) c , l (->) i , i (->) l , l (->) 1 ,
 1 (->) l , r n (->) m , m (->) r n ,
 r i (->) n , n (->) r i , c l (->) d ,
 d (->) c l , o (->) "0" , "0" (->) o ,
 r t (->) n , r t (->) i t , r m (->) n n ,
 r r i (->) m , m (->) r r i , n i (->) m ,
 l i (->) h , u (->) i r ] ;

# very generative, using word-list/dictionaries
define OCR  DICTIONARY .o. PARAL ;
# for correcting
up pcrtuguesc ;
```

Modelling

Always from dialect/variant to standard/pivot?

- Both possibilities
- Depending on the aim/grammar/examples
- Bidirectional transducers
  - `down / up`
  - `flookup / flookup -i`
  - `.i` operator
- `.l` and `.u` also (lower and upper languages)

Compounding with a lexicon (wordlist/morphology)

- more precision
- coverage?

Second grammar: gal2por

- Describing changes for writing Portuguese into Galician
- Based on http://gramatica.usc.es/~gamallo/port2gal.htm
  - Original perl file: *por2gal/port2gal.perl*
- File: ***por2gal/por2gal***

por2gal (1)

```
define Border [" "|"."|","|";"|"!"|"?"|"-"] ;

define R1 l h -> l l ;
        #coelho:coello
define R2 n h -> ñ ;
        #minha:miña
define R3 m -> n || _ [ Border | .#. ] ;
        #um:un
define R4 s s -> s ;
        #passar:pasar
define R5 ã o -> [ó|á] n || _ [ Border | .#. ]  ;
        #alemão:alemán
define R6 ã e  -> á n || _ s [ Border | .#. ]  ;
        #alemães:alemans
define R7 õ e  -> ó n || _ s [ Border | .#. ]  ;
        #camiões:camións
```

por2gal (2)

```
define R8 m -> n h || [Border|.#.] (a l g) u _ a
                                    [ Border | .#. ];
        #no general
        #uma:unha
        #alguma:algunha
define R91 j -> x ;
define R92 g -> x || _ [ e | i ] ;
        #hoje:hoxe
        #gigante:xigante
        #geral:xeral
define R9 R91 .o. R92 ;
define PhonChang R1 .o. R2 .o. R3 .o. R4 .o. R5 .o.
        R6 .o. R7 .o. R8 .o. R9 ;
```

por2gal (lexicon)

```
# general lex
define LexComm ?+ ;
# exceptions
define LexChang [{açúcar}:{sucre}|{cena}:{escena}] ;
# priority union
define Lex LexChang .P. LexComm ;
# composition
define Normal Lex .o. PhonChang ;
regex Normal;
down alemão   # result: alemán/alemón
# lexicon from wordlist (galician)
read text lemmas_gl.txt
define LEXGL;
# composition (target language)
regex Normal .o. LEXGL;
down alemão   # result: alemán
```

por2gal (test using a file)

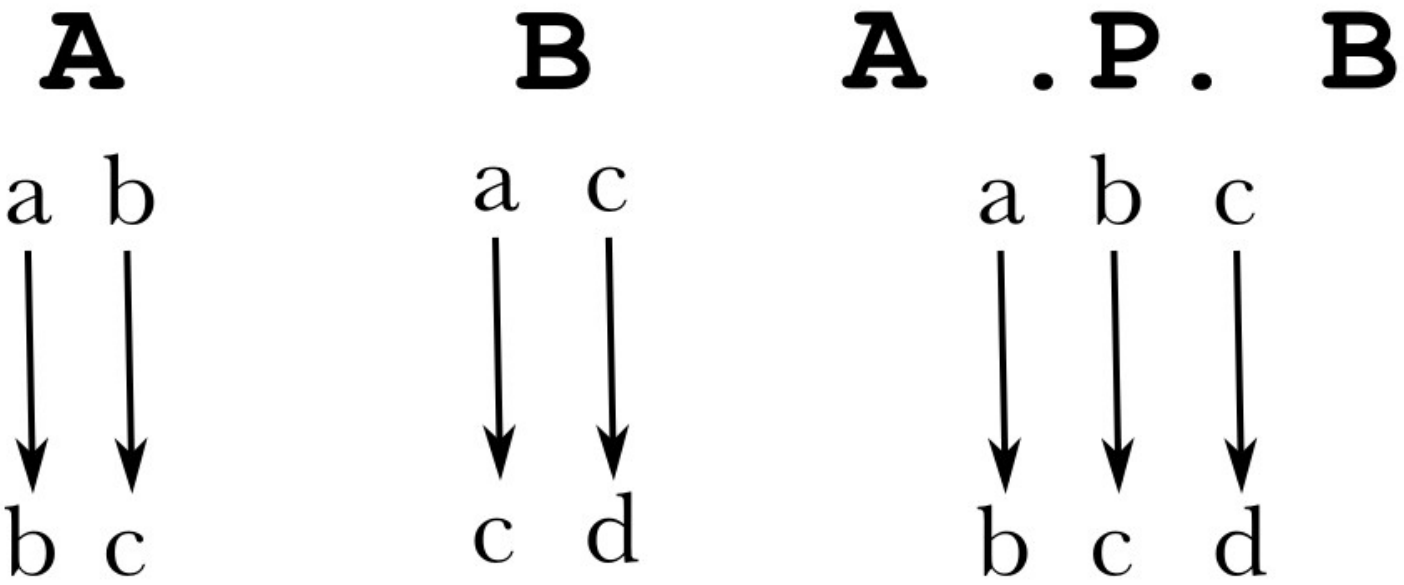
```
# ambiguous output
flookup -ib por2gal.fst <text_pt | more

# not enough dictionary-coverage (plurals...)
flookup -ib por2galdict.fst <text_pt | more

# test word-by-word (using a tokenizer)
./tokenize text_pt | flookup -ib por2galdict.fst
```

Combining with priority union

Priority union operation for overriding regular with irregular forms



In this case, we need to calculate the transducer [Exceptions .P. Grammar]

Exercise: improving gal2por

- *pessoa/persoa* (small dictionary)
- *dele/del aquele/aquel* ....
- Some verbs: *-obri- / -ubri-*
- Named entities (persons, places, companies...)

Bigger dictionaries

- Using Regular expressions

```
define LexChang [{açúcar}:{sucre}|{cena}:{escena}] ;
```

- Using text file of pairs

```
read text por2gal.txt
define LEX1;
```

- Using text file with morpholtactics (paradigms)

```
read lexc por2gal.lexc
define LEX2;
```

More examples...

Tweet normalization in Spanish
rules from standard to variants

```
# examples: graaaaaaaande, acostao

define R1 ?* [ a:[a+] | e:[e+] | i:[i+] | o:[o+] | u:
[u+] ] ?*;

define R2 d (->) 0 || [a|i] _ [o|a] .#. ;

define RNorm R2 .o. R1 ;
```

Phonological vs. morphological rules

- Sometimes phonology is not enough
- Morphological information is necessary
    - at least morpheme-boundary ("+" or "^")
- But in this case a morphological analyser/guesser is necessary
    - for the standard/pivot language

More examples...

Rule order in morphology

Simple Spanish (pluralization) rules

```
# examples: papel+s:papeles; pez+s:peces
# sequential (ordered)
(1)    z -> c || _ "+" s .#. ;
(2) [..] -> e || Cons "+" _ s .#. ;
```

Compare:

```
pez+s (1) → pec+s  (2) → pec+es
pez+s (2) → pez+es (1) → *pez+es
```

More examples...

One Basque morphological rule

```
# (morpho)phonology with r (two special R)
define MM "+" ;      # morpheme-boundary
## hard r (R)
define R2 R -> r r || _ MM (Q) Vowel ;
       # zakuR+a:zakurra
       # itziaR+Qen:itziarr+Qen:itziarren
define RR R -> r ;
       # ekaR+tzen:ekartzen
## epenthetical r (Q)
define Q0 Q -> 0 || Cons MM _ ;
       #ur+Qen:uren
define QR Q -> r ;
       # amA+Qen:amA+ren:amaren
define RRule R2 .o. RR .o. Q0 .o. QR;
```

Example of morphology: lexc+rules

General strategy:

Create lexc-grammar, load in foma, define:

```
read lexc english.lexc
define Lexicon;
```

Replacement rules in foma:

```
define Rule1 x -> y ...
```

Combine with composition:

```
define Grammar Lexicon .o. Rule1 .o. ... .o. RuleN;
regex Grammar;
```

Files: ***examples/english***

```
more english.lexc
more english.foma
foma -l english.foma
```

Third grammar: Basque dialect

- Real example based on morphology
- Working in some applications (spelling checking, "machine translation")
- New morphemes and rules are integrated in the description
  - source: operator in foma for integration of sections

```
read lexc lexicon-st-dial.lexc
source standard-rules.txt
source dialectal-rules.txt
...composition, save...
```

- Simplified version
- File: ***basque-biscayan/normalize-basque.txt***

New operators

Forbidden for some combinations

against overgeneration

~ operator: negation

$ operator: substring

$A is equivalent to ?* A ?*

```
define ZETADEBEKU2 ~$[ [ e | o ] MM t z e ];
        # ibiL+tzea: ibiltea
        # jaso+ten:jasoten
        # jaso+tzen:------
        # paga+ten:pagaten
        # paga+tzen:pagatzen
        # bete+tzen:----
```

foma for other tasks

Surface syntax:

named entities, numbers, codes, URLs, hashtags...

Files: ***examples/other***

```
foma -l simple_enti
foma -l simple_date
foma -l transl_numbers
```

Xerox, examples:

http://web.stanford.edu/~laurik/fsmbook