Natuurlijke Taalverwerking II 2006/2007

Gosse Bouma

www.let.rug.nl/~gosse/ntv2

Overview

- Spelling Rules
  - Context-sensitive, Obligatory
- Replace,
  - Longest-match, left-to-right, obligatory replacement
- Applications
  - Grapheme to Phoneme Conversion

Spelling Reform

- (Dutch) Words should be spelled the way they are pronounced,
- In the new Dutch Spelling,
  - x will be written as ks
  - qu will be written as kw,
  - c will be written as s or k (depending on pronunciation),
  - ch will be written as g,
  - isch will be written as ies.

Spelling Reform

extra	→	ekstra
frequentie	→	frekwentie
centraal	→	sentraal
camera	→	kamera
lach	→	lag
automatisch	→	automaties
exotoxine	→	eksotoksine
accu	→	akku
accent	→	aksent
acquit	→	akkwit

Rules for Spelling Changes

- Replace x by ks,
- First Attempt (wrong):
  - `[[?* , x:[k,s] ]*, ?*]`
  - xerox → kseroks, kserox, xeroks, xerox
  - Replacement is obligatory!
- Second Attempt (ok):
  - `[[(? -x) * , x:[k,s] ]*, (? -x) *]`
  - `{ (? -x), x:[k,s]}*` (shorter)

Context-Sensitive Rules

- Replace c with s if followed by e or i
  - `cent, politici`
- First Attempt:
  - `{ ? - c, [c:s, {e,i}] }*`
  - cent → sent,
  - politicus → no output

Context-Sensitive Rules 2

- Replace c followed by e or i with s, elsewhere with k

```
{ ? -c,
  [c:s, {e,i}],
  [c:k, ? -{e,i}] }*
```

- cent → sent,
- politicus → politikus,
- accu → akcu

Third Attempt, Double C

```
{ ? - c,
  [c:s,{e,i}],
  [c:k,{ ? -{e,i,c},
         [c:s,{e,i}],
         [c:k,? -{e,i}]  }  ]  }*
```

- accu → akku,
- accent → aksent,
- accccu → akkcku

Fourth Attempt, Double C

```
{? -c, [c:s, {e,i}], [c:k, ? -{e,i}]}*
                      o
{? -c, [c:s, {e,i}], [c:k, ? -{e,i}]}*
```

- accu → akku,
- accent → aksent,
- accccu → akkkku

Fifth Attempt, Double C

```
{? - c,
 [c:k*,{[c:s,{e,i}],
        [c:k,? -{e,i,c}] } ] }*
```

- accu → akku,
- accent → aksent,
- accccu → akkkku

Multiple C Rules

- isch → ies,
  - ⋆ Must obligatory replace a string of 4 characters, but leave untouched all other 1-4 sequences ???
- ch → g
- c/k/s rule
- Order specific rules before more general rules

Multiple C Rules

```
[[ ?*, [i,s,c,h]:[i,e,s]]*, ?*]
                o
      ~ $[i,s,c,h]
                o
  [[ ?*, [c,h]:g]*, ?*]
                o
          ~ $[c,h]
                o
          cks_rule
```

Replace

- Phonological rules as optional replacement composed with a transducer filtering unreplaced patterns. (Karttunen 1997, Kaplan & Kay, 1994)
- `replace(A x B, LC, RC)`
- Obligatory replace all occurences of A by B, in the context of LC _ RC.

The Replace Operator

- `replace(A x B, LftContext, RghtContext)`:
  - Obligatorily replace all A's between LftContext and RghtContext, by B.
  - A x B is a RegEx defining an arbitrary transducer,
  - LftContext and RghtContext are RegEx's for a recognizer

Example of replace

- `replace({a,e}* x 'V',{b,d},{b,d})`
- robbed → robbVd
- dead → dVd

C-rules with Replace

```
replace([s,c,h]:[e,s],[i],[])
                o
  replace([c,h]:g,[],[])
                o
  replace(c:s,[],{i,e})
                o
   replace(c:k,[],[])
```

Replace Longest Match

- `replace({a,e,i,u}* x 'V',[],[])`
- beard → bVrd *bVVrd

Replace Left to Right

- Replace works from left to right,

```
replace(a:b,a,[])
```

- aa → ab
- aaa → aba

Hyphenation

- Insert a hyphen between two syllables,
- Maximizing the onset of the second syllable

```
replace([]:-, syllable, syllable)
```

- alfabet → al-fa-bet, *alf-a-bet
- aap → a-ap

Replace Longest Match

- Replace performs longest match:
  - ⋆ It replaces the longest substring in the input matching the target
- replace([[]:@, nucleus, []:@],[],[])
- aap → @aa@p , * @a@@a@p

Hyphenation

```
replace([[]:@, nucleus, []:@],[],[])
                o
 replace([]:-, [@,coda^],[onset^,@]
                o
      replace(@:[],[],[])
```

- alfabet → @a@lf@a@b@e@t → al-fa-bet
- aap → @aa@p → aap

Verbal Inflection

Root		werk	raad
1st pers sing	(ik)	werk	raad
3rd pers sing	(hij,zij)	werkt	raadt
plural	(wij, jullie)	werken	raden
sing past tense	(ik, hij, zij)	werkte	raadde
plur past tense	(wij, jullie)	werkten	raadden

Verbal Inflection

- A regular (weak) verbal root in Dutch can be inflected with
  - ⋆ +t (3rd person singular form),
  - ⋆ +en (plural and infinitive),
  - ⋆ +Te (singular past tense),
  - ⋆ +Ten (plural past tense)

Examples

Lexical	Surface
loop+t	loopt
brand+t	brandt
ga+t	gaat
zet+t	zet
bof+t	boft
leev+t	leeft

Lexical	Surface
werk+en	werken
maak+en	maken
zie+en	zien
werk+Te	werkte
ren+Te	rende

Grapheme to Phoneme Conversion

- Convert sequences of characters (words) into sequences of phonemes,
- For (unknown words in) text-to-speech synthesis,
- For pronunciation-based spelling correction,
  - abbreviate → @brivI1t
  - zucchinis → zUkinIz
  - aankruiend → aNkrL@nt
  - zygoten → ziGot@

Finite State Grapheme to Phoneme Conversion

- Leftmost, longest-match replacement,
- Segment character strings into graphemes,
- Convert graphemes into phonemes.

Finite-state G2P

```
macro(graph2phon,
  segmentation     %% segment the input
o mark_begin_end   %% add '#'
o conversion       %% apply rules
o clean_up    ).   %% remove markers
```

input:	aanknopingspunt
segmentation:	aa-n-k-n-o-p-i-ng-s-p-u-n-t-
mark_begin_end:	#-aa-n-k-n-o-p-i-ng-s-p-u-n-t-#
conversion:	#-a+N+k-n-o-p-I+N+s-p-}+n-t-#
clean_up:	aNknopINsp}nt

Segmentation

one hour dull → o-n-e h-ou-r d-u-ll

- Going from left to right, mark longest-matching substring as grapheme:

```
macro(graph, {o,u,ou,ll,h,r,..}).
macro(segmentation,
  replace([ graph, []:-], [],[])).
```

Conversion

```
macro( conversion,
          special_vowel_rules
       o  short_vowel_rules
       o  special_consonant_rules
       o  default_rules ).
```

G2P operator

- Conversion Rules:
  - ⋆ At most one rule applies to each grapheme,
  - ⋆ Each rule applies to a complete grapheme (not some substring).

```
macro( g2p(Target,LC,RC),
       replace([Target,- x +],
               [ignore(LC,{+,-}),{-,+}],
               ignore(RC,{+,-}) ) ).
```

Conversion Rules

```
macro(special_vowel_rules,
  %% museum
  g2p([e,u] x [e,j,}], [],  m )
  %% mogelijkheid
o g2p([i,j] x @, l, k )
    .... ).
macro(short_vowel_rules,
  g2p(a x 'A', [], [cons, {cons , #}] )
o g2p(e x 'E', [], {[t,t],[k,k],x,...})
    .... ).
```

Conversion Rules

```
macro(special_consonant_rules,
   g2p(b x p, [], {s,t,#} )
   ...).
macro(default_rules,
  g2p({ [a,a] x a, [a,a,i] x [a,j],...,
        [b,b] x b, ...
      }, [], [] )
   ).
```

Some Results

- 77 Graphemes,
- $\approx$ 40 non-default (contexted) rules,
- $\approx$ 40 default rules,
- Automaton :
  - 750 states,
  - 20K transitions.

Edit distance

- Edit distance of strings S1 and S2:
  - number of substitutions, deletions, and insertions required to make S1 equal to S2.
- ZIGot@n ZiGot@
- Edit Distance = 2

Accuracy

- Phoneme Accuracy $= \frac{\text{Edit Distance}}{\text{System String Length}}$
- Tested on Celex data (10K words):
  - 93.6% phoneme accuracy,
  - 60.6% word accuracy
  - (avg. word length = 10).

Improving Accuracy

- Further manual improvements require:
  - A large number of exceptional cases,
  - Heuristic rules (i.e. not 100% correct).