Recognizers vs Transducers

- A finite state recognizer is an automaton which accepts strings (yes/no decisions):
  - recognize Zip Codes, Proper Names, Syllables, ...

- A finite state transducer is an automaton which maps one string onto another string:
  - Map Letters onto Phonemes, Inflected words onto Base Forms, Words onto Part of Speech Tags, ....

Stemming

- Translate a word into its base form,
- For information retrieval:
  - Given a query, find relevant documents
  - A query with republican, can lead to a document with republicans.

Stemming

Georgia  georgia
Republicans  republican
are  be
getting  get
strong  strong
encouragement  encouragement
to  to
enter  enter
da  a
candidate  candidate
Part of Speech Tagging

- Translate a sequence of words into a sequence of Part of Speech Tags
- Useful as a first step towards full parsing or to support searching for linguistic patterns,

Grapheme to Phoneme Conversion

- Translate a sequence of letters into a sequence of phonemes
- Required for Text to Speech applications
- Each letter or sequence of letters is translated into a phoneme

Encoding a Rule

- e → I / \{t,d\} _ d #
- abbreviated# → abbreviatId#
**Regex Notation for Transducers**

- \([a:b, c^*]\) translates, among others, `accc` in `bccc`.

- `:` is the ‘pair’-operator: it translates a symbol A in a symbol B.

- \([a:b, c^*]\) is short for \([a:b, (c;c)^*]\)

By default, a regular expression without `:` is read as the identity-transducer: every symbol in the input is mapped onto itself.

**Dutch Dimunitives**

<table>
<thead>
<tr>
<th>input</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>huis+je</td>
<td>huisje</td>
</tr>
<tr>
<td>haan+je</td>
<td>haantje</td>
</tr>
<tr>
<td>man+je</td>
<td>mannnetje</td>
</tr>
</tbody>
</table>

```
? *,[[s,+ :[]],
    [a,a,n,+ :t],
    [~a,a,n,[]:n,[]:e,+ :t]
},
    j,e
]
```
An transducer is deterministic if for every state and input symbol, at most a single transduction to a new state is possible.

Non-deterministic transducers can sometimes be made deterministic, but not always.

Non-deterministic recognizers can always be made deterministic.

Two Sources of Non-determinism

- Unbounded Look-ahead
  - \{a:b, c*, b\}, \{a:d, c*, d\}

- Multiple outputs
  - bloem+je → bloempje
  - bloem+je → bloemetje
  - [?*, o, e, m, {:+p, +:[e,t]}, j, e]
Deterministic Transducers

- Deterministic transducers are more efficient than non-deterministic transducers (because no choice-points/backtracking/search is required).
- But deterministic transducers can be much larger than corresponding non-deterministic transducer.
- \( \text{t.determinize} \) option in FSA.

From English to Dutch Numbers

- Automatic translation of (spoken) English into Dutch requires translation of number words,
- twentyone → eenentwintig,
- twentyone → 21 → eenentwintig

Making a Transducer Deterministic

\[
\begin{align*}
\text{acb} & \rightarrow \text{bcb} \\
\text{acd} & \rightarrow \text{dcd}
\end{align*}
\]

From Number Words to Numbers

macro(one, \{one:1, two:2, ..., nine:9 \}).
macro(twenty, \{twenty:2, thirty:3, ..., ninety:9 \}).
macro(eng2num, \{ one, ten: [1,0], eleven: [1,1], ..., nineteen: [1,9], [twenty, one] \}).
From English to Dutch Numbers

- Transducer $T_1$ for translating English Number Words into Numbers,
- Transducer $T_2$ for translating Numbers into Dutch Number Words
- The output of $T_1$ is used as input by $T_2$.

Composition

- The composition of transducers $T_1$ and $T_2$ is a new transducer $T_3$, which is equivalent to passing the input through $T_1$, taking the output of $T_1$ as input for $T_2$, and taking the output of $T_2$ as output.
- $T_1 \circ T_2$ denotes the composition of $T_1$ and $T_2$.

Number Translation by Composition

```
macro(eng2num, 
   {{one,ten:[1,0],..}}).
macro(num2dut, 
   {1:een,2:twee, ....}).
macro(eng2dut, 
   eng2num o num2dut).
```

Input/Output reversal

- The inverse of a transducer $T$ is a transducer which takes as input the output of $T$, and produces as output the input of $T$.
- In FSA: $\text{inverse}(T)$.
- Translating Numbers into English expressions:
  - $\text{eng2num}$
    - twelve $\rightarrow$ 12
  - $\text{inverse(eng2num)}$
    - 12 $\rightarrow$ twelve
Finite State POS Tagging

- Assign Part of Speech tags to words,
- but many words have more than one POS:
  - The/det report/n was/aux written/v
  - The/det police/n has/aux to/aux report/v
  - all/det problems/n

A Solution:
- A non-deterministic $T$ which assigns a word all possible POS tags,
- Recognizers $R$ which filter the output of $T$,
- Compose $T$ and (the identity transducer for) $R$.

```
macro(lexicon, 
  { all:det, has:aux, police:n, problems:n, 
    report:{v,n}, the:det, to:v, was:aux, 
    written:v}* ).
macro(no_det_v, 
macro(tagger, 
  lexicon o no_det_v ).
```