Lecture 2: Morphology and Finite State Techniques

1. A brief introduction to morphology  
2. Using morphology in NLP  
3. Aspects of morphological processing  
4. Finite state techniques
Morphemes are the smallest meaningful units of language. Words are composed of morpheme(s).
Morphemes are the smallest meaningful units of language. Words are composed of morpheme(s).

**Affix**: morpheme which only occurs in conjunction with other morphemes.
- suffix (*units*), prefix (*in*complete), infix, circumfix
Infix Tagalog (Philippines)

**Infix**

- **Infix:** -um-

- **Tagalog (Philippines)**

- **Infix:** -um-

- **basa** (read)
- **b-um-asa** (read.PAST)
- **sulat** (write)
- **s-um-ulat** (wrote)

Source: J Hana & A Feldman. ESSLLI 2013: Computational Morphology. 
Infix Tagalog (Philippines)

Infix: -um-

basa → read
b-um-asa → read.PAST
sulat → write
s-um-ulat → wrote

Circumfix: occur on both sides

circumfix: ge-X-te

berg → mountain
g-e-berg-te → mountains

*Dutch collectives

*ge-berg
*berg-te

Source: J Hana & A Feldman. ESSLII 2013: Computational Morphology.
Productivity

*Productivity*: whether affix applies generally, whether it applies to new words

- *sing*, *sang*, *sung*
- *ring*, *rang*, *rung*
Productivity: whether affix applies generally, whether it applies to new words

- *sing*, *sang*, *sung*
- *ring*, *rang*, *rung*

- But, *ping*, *pinged*, *pinged*

This infixation pattern is not productive: *sing*, *ring* are *irregular*
Inflection and derivation

**Inflection** creates new forms of the same word
- e.g. *bring*, *brought*, *brings*, *bringing*
- generally fully productive (modulo irregular forms)
- tends to affect only its *syntactic function*

**Derivation** creates new words
- e.g. *logic*, *logical*, *illogical*, *illogicality*, *logician*, etc.
- generally semi-productive: e.g., *escapee*, *textee*, *?dropee*, *?snoree*, *cricketee* (* and ?)
- tends to be more irregular; the meaning is more idiosyncratic and less compositional.
- tends to affects the *meaning* of the word, and may change part-of-speech
**Root**: nucleus of the word that affixes attach too.

**Compounds** contain more than one root.

*bookshopped*
**Root**: nucleus of the word that affixes attach too.

**Compounds** contain more than one root.

*bookshopped*

**Stem**: word without its inflectional affixes = root + all derivational affixes.
**Root**: nucleus of the word that affixes attach too.

**Compounds** contain more than one root.

*bookshopped*

**Stem**: word without its inflectional affixes = root + all derivational affixes.

**Lexeme**: the set of all forms related by inflection (but not derivation).

\{bookshops, bookshopped, bookshopping, ...\}

**Lemma**: the *canonical/base/dictionary/citation* form of a lexeme chosen by convention.

bookshop (cf. the stem—bookshopp)
## Compound and multiword expression

### Compound

1. **beam-width**
2. **sunset**

### Multiword expression

Combinations of two or more words that exhibit syntactic and semantic idiosyncratic behavior.

<table>
<thead>
<tr>
<th>Fixed</th>
<th>(Syntactically) flexible</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>by and large</em></td>
<td><em>put on the clothes</em></td>
</tr>
<tr>
<td></td>
<td><em>put the clothes on</em></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Non-compositional</th>
<th>Semi-compositional</th>
<th>Compositional</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>kick the bucket</em></td>
<td><em>spill the beans</em></td>
<td><em>strong tea</em></td>
</tr>
<tr>
<td>(reveal the secret)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Etymology

*slither, slide, slip* etc have somewhat similar meanings; but *sl-* is not a morpheme.

*slith, slid* and *slip* are historically related.

See www.etymonline.com/word/slide
Internal structure: order

The order of morphemes matters

- talk-ed ≠ *ed-talk
- re-write ≠ *write-re
- un-kind-ly ≠ *kind-un-ly

Suffixing is more frequent than prefixing and far more frequent than infixing/circumfixing

- Postpositional and head-final languages use suffixes and no prefixes. cf. harmonic order: ⟨VO, PO⟩, ⟨OV, OP⟩
- Prepositional and head-initial languages use not only prefixes but also suffixes.
- Many languages use exclusively suffixes and no prefixes
- Very few languages use only prefixes and no suffixes
Internal structure: ambiguity

*dog*

Source of photo: commons.wikimedia.org/w/index.php?curid=73851814

*Morpheme ambiguity*: stems and affixes may be individually ambiguous.

*dogs*
Internal structure: ambiguity

**Structural ambiguity:** different combinations of morphemes

- **unlockable**
  - un- + lock
  - un- + lock + -able

  Capable of being unlocked. Not capable of being locked.

**Cross word boundaries:** syntax all the way down

- **beautiful dancer**
  - beautiful + dance + -er
  - beautiful + dance
  - beautiful + -er


semantics.uchicago.edu/kennedy/classes/f11/na/docs/larson08.pdf

More about *unlockable*: en.wiktionary.org/wiki/unlockable
Abstraction

Surface form → Abstraction

- Indefinite article: *an* orange, *a* building
- Negation: *un*happy, *in*complete, *im*possible, *ir*rational
- Irregular: *sing*, *sang*, *sung*

The same morpheme may have different variants, which are called *allomorphs*. Allomorphs have the same function but different forms.
Computational tasks

**LEMMA**

natural language expression $\underbrace{\text{word}}_{\text{saw}} \rightarrow \text{lexeme} \quad \{\text{see, saw}\}

**TAGGING**

contextualized word $\underbrace{\text{contextualized word}}_{\text{saw @ J saw M}} \rightarrow \text{contextualized tag} \quad \langle\text{see, VERB.PAST}\rangle$

**SEGMENTATION**

word $\underbrace{\text{word}}_{\text{meaningful}} \rightarrow \text{morphemes (subwords)} \quad \text{mean+ing+ful}$

**GENERATION**

word $\underbrace{\text{word}}_{\text{saw}} \rightarrow \text{abstract word} \quad \langle\text{see, VERB.PAST}\rangle$

compiling a full-form lexicon, stemming for Information Retrieval, preprocessing for parsing, . . .
Segmentation

antidisestablishmentarianism ⇒ anti-dis-e-stabl-ish-ment-arian-ism

antidisestablishmentarianism

anti dis establish ment arian ism

en.wikipedia.org/wiki/Antidisestablishmentarianism
www.etymonline.com/word/antidisestablishmentarianism

Bioinformatics
Text normalization

- Not using any punctuation at all
  Eh speak english mi malay not tt good (Eh, speak English! My Ma-lay is not that good.)

- Using spell-ing/punctuation for emphasis
  goooooood Sunday morning !!!!!!! (Good Sunday morning!)

- Using phonetic spelling
  dat iz enuf (That is enough)

- Dropping vowel
  i hv cm to c my luv. (I have come to see my love.)

- Introducing local flavor
  yar lor where u go juz now (yes, where did you go just now?)

- Dropping verb
  l hv 2 go. Dinner w parents. (I have to go. Have dinner with parents.)

More: noisy-text.github.io/norm-shared-task.html
Cross-lingual variants

- The phones making up a morpheme don’t have to be contiguous, e.g. in Hebrew,

<table>
<thead>
<tr>
<th>Root</th>
<th>Pattern</th>
<th>PoS</th>
<th>Phonological Form</th>
<th>Gloss</th>
</tr>
</thead>
<tbody>
<tr>
<td>ktb</td>
<td>CaCaC</td>
<td>v</td>
<td>katav</td>
<td>‘wrote’</td>
</tr>
<tr>
<td>ktb</td>
<td>hiCCiC</td>
<td>v</td>
<td>hixtiv</td>
<td>‘dictated’</td>
</tr>
<tr>
<td>ktb</td>
<td>miCCaC</td>
<td>n</td>
<td>mixtav</td>
<td>‘a letter’</td>
</tr>
<tr>
<td>ktb</td>
<td>CCaC</td>
<td>n</td>
<td>ktav</td>
<td>‘writing, alphabet’</td>
</tr>
</tbody>
</table>

from E. Bender’s tutorial ([faculty.washington.edu/ebender/papers/100things.pdf](faculty.washington.edu/ebender/papers/100things.pdf))
Cross-lingual variants

- The phones making up a morpheme don’t have to be contiguous, e.g. in Hebrew,

<table>
<thead>
<tr>
<th>Root</th>
<th>Pattern</th>
<th>PoS</th>
<th>Phonological Form</th>
<th>Gloss</th>
</tr>
</thead>
<tbody>
<tr>
<td>ktb</td>
<td>CaCaC</td>
<td>v</td>
<td>katav</td>
<td>‘wrote’</td>
</tr>
<tr>
<td>ktb</td>
<td>hiCCiC</td>
<td>v</td>
<td>hixtiv</td>
<td>‘dictated’</td>
</tr>
<tr>
<td>ktb</td>
<td>miCCaC</td>
<td>n</td>
<td>mixtav</td>
<td>‘a letter’</td>
</tr>
<tr>
<td>ktb</td>
<td>CCaC</td>
<td>n</td>
<td>ktav</td>
<td>‘writing, alphabet’</td>
</tr>
</tbody>
</table>

from E. Bender’s tutorial (faculty.washington.edu/ebender/papers/100things.pdf)

- **English** morphology is essentially concatenative
cf. duplication in **Chinese**, e.g.

```
高兴 → 高高兴兴
happy → 高兴高兴
```
Spelling rules

- Irregular morphology — inflectional forms have to be listed
- Regular phonological and spelling changes associated with affixation, e.g.
  - *-s* is pronounced differently with stem ending in *s*, *x* or *z*
  - spelling reflects this with the addition of an *e* (*boxes* etc)

*morphophonology*

- In English, description is independent of particular stems/affixes
Lexical requirements for morphological processing

Knowledge

affixes, plus the associated information conveyed by the affix

- ed VERB.PAST
- ed VERB.PSP
- s NOUN.PLURAL

irregular forms, with associated information similar to that for affixes
began VERB.PAST begin
begun VERB.PSP begin
Automata

- Circles are **states** of the automaton.
- Arrows are called **transitions**.
- The automaton changes states by following transitions.
- The double circle indicates that this state is an **accepting state**. The automaton accepts the string if it ends in an accepting state.
Automata

- Circles are states of the automaton.
- Arrows are called transitions.
- The automaton changes states by following transitions.
- The double circle indicates that this state is an accepting state. The automaton accepts the string if it ends in an accepting state.

- **Form Transformation**: augmenting transitions
  
  $\text{input} \rightarrow \text{input:output}$
Finite state transducer

- cakes → cake#s
- boxes → box#s
Analysing boxes

INPUT

OUTPUT

| b | o | x | e | s |

```
q1 -> q4
q4 -> q1
q1 -> q2
q2 -> q3
q3 -> q4
```

Transitions:
- $\epsilon$: # from $q_1$ to $q_2$
- $s$: $s$ from $q_2$ to $q_3$
- $e$: $e$ from $q_1$ to $q_4$
- $e$: $e$ from $q_4$ to $q_1$
- $s$: $s$ from $q_1$ to $q_2$
- $s$: $s$ from $q_2$ to $q_3$
- $x$: $x$ from $q_4$ to $q_1$
- $z$: $z$ from $q_4$ to $q_1$

States:
- $q_1$
- $q_2$
- $q_3$
- $q_4$
Analysing boxes

OUTPUT
INPUT

$q1$ → $q2$: $\epsilon:\#$
$q2$ → $q3$: $s:s$
$q4$ → $q1$: $e:e$
$q1$ → $q4$: $\epsilon:e$
$q4$ → $q1$: $s:s$
$q1$ → $q4$: $x:x$
$q1$ → $q4$: $z:z$
$q1$ → $q4$: $e:other$
$q1$ → $q4$: $other:other$
$q1$ → $q4$: $s:s$
$q1$ → $q4$: $x:x$
$q1$ → $q4$: $z:z$
$q1$ → $q4$: $e:other$
$q1$ → $q4$: $other:other$
$q1$ → $q4$: $s:s$
$q1$ → $q4$: $x:x$
$q1$ → $q4$: $z:z$
$q1$ → $q4$: $e:other$
$q1$ → $q4$: $other:other$
$q1$ → $q4$: $s:s$
$q1$ → $q4$: $x:x$
$q1$ → $q4$: $z:z$
$q1$ → $q4$: $e:other$
$q1$ → $q4$: $other:other$
$q1$ → $q4$: $s:s$
$q1$ → $q4$: $x:x$
$q1$ → $q4$: $z:z$
$q1$ → $q4$: $e:other$
$q1$ → $q4$: $other:other$
$q1$ → $q4$: $s:s$
$q1$ → $q4$: $x:x$
$q1$ → $q4$: $z:z$
$q1$ → $q4$: $e:other$
$q1$ → $q4$: $other:other$
$q1$ → $q4$: $s:s$
$q1$ → $q4$: $x:x$
$q1$ → $q4$: $z:z$
$q1$ → $q4$: $e:other$
$q1$ → $q4$: $other:other$
$q1$ → $q4$: $s:s$
$q1$ → $q4$: $x:x$
$q1$ → $q4$: $z:z$
$q1$ → $q4$: $e:other$
$q1$ → $q4$: $other:other$
$q1$ → $q4$: $s:s$
$q1$ → $q4$: $x:x$
$q1$ → $q4$: $z:z$
$q1$ → $q4$: $e:other$
$q1$ → $q4$: $other:other$
$q1$ → $q4$: $s:s$
$q1$ → $q4$: $x:x$
$q1$ → $q4$: $z:z$
$q1$ → $q4$: $e:other$
$q1$ → $q4$: $other:other$
$q1$ → $q4$: $s:s$
$q1$ → $q4$: $x:x$
$q1$ → $q4$: $z:z$
$q1$ → $q4$: $e:other$
$q1$ → $q4$: $other:other$
$q1$ → $q4$: $s:s$
$q1$ → $q4$: $x:x$
$q1$ → $q4$: $z:z$
Analysing boxes

Output

Input

$\begin{array}{c}
\text{b} \\
\text{b} \\
\text{o} \\
\text{x} \\
\text{e} \\
\text{s}
\end{array}$

---

$q_1 \xrightarrow{\varepsilon: \#} q_2 \xrightarrow{s:s} q_3$

$q_2 \xrightarrow{\varepsilon: \#} q_1$

$q_4 \xrightarrow{s:s} q_2$
Analysing boxes

INPUT

b
boxes

OUTPUT

b

\[ q_1 \xrightarrow{\text{e:e}} \]
\[ q_2 \xrightarrow{\text{s:s}} \]
\[ q_3 \]

\[ q_4 \xrightarrow{\text{s:s}} \]
\[ x:x \]
\[ z:z \]

\[ q_1 \xrightarrow{\text{e:e}} \]
\[ \text{other:other} \]

\[ q_2 \xrightarrow{\text{e:#}} \]

\[ q_3 \xrightarrow{\text{s:#}} \]

\[ q_4 \xrightarrow{\text{e:#}} \]
\[ \text{other:other} \]
Analysing boxes

OUTPUT

INPUT

| b | o | x | e | s |

\[ q_1 \quad \epsilon: \# \quad q_2 \quad s:s \quad q_3 \]

\[ q_4 \quad e:e \quad q_3 \quad e:e \quad q_1 \quad e:e \]

other:other

s:s
x:x
z:z

e:?

s:s
x:x
z:z

\[ q_4 \quad e:e \quad q_4 \quad s:s \quad e:e \]

other:other

\[ q_4 \quad e:e \quad q_4 \quad s:s \quad e:e \]

other:other
Analysing boxes

INPUT

OUTPUT

\[
\begin{array}{c|cc}
\text{b} & \text{o} & \text{x} \\
\hline
\text{b} & \text{o} & \text{x} & \text{e} & \text{s} \\
\end{array}
\]

\[
\begin{array}{c}
q_1 \\
q_2 \\
q_3 \\
q_4 \\
\end{array}
\]

\[
\begin{array}{cc}
\text{e:e} & \text{other:other} \\
\text{e:e} & \text{other:other} \\
\text{e:e} & \text{other:other} \\
\text{e:e} & \text{other:other} \\
\end{array}
\]

\[
\begin{array}{cc}
\text{s:s} \\
\text{x:x} \\
\text{z:z} \\
\end{array}
\]

\[
\begin{array}{cc}
\text{e:}\# \\
\text{s:}\# \\
\text{e:}\# \\
\text{e:}\# \\
\end{array}
\]
Analysing boxes
Analysing boxes
Finite-state machine

• A symbolic system that can recognize or transform forms.
Finite-state machine

- A symbolic system that can recognize or transform forms.
- An automaton remembers only a finite amount of information.
Finite-state machine

- A symbolic system that can recognize or **transform forms**.
- An automaton remembers only a finite amount of information.
- Information is represented by its states.
Finite-state machine

- A symbolic system that can recognize or **transform forms**.
- An automaton remembers only a finite amount of information.
- Information is represented by its states.
- State changes in response to inputs and may trigger outputs.
Finite-state machine

- A symbolic system that can recognize or **transform forms**.
- An automaton remembers only a finite amount of information.
- Information is represented by its states.
- State changes in response to inputs and may trigger outputs.
- **Transition rules define how the state changes in response to inputs.**
A symbolic system that can recognize or **transform forms**.

An automaton remembers only a finite amount of information.

Information is represented by its states.

State changes in response to inputs and may trigger outputs.

Transition rules define how the state changes in response to inputs.

Given a sequence of input symbols, a recognition process starts in the start state and follow the transitions in turn. Input is accepted if this process ends up in an accepting state.
Finite-state machine

- A symbolic system that can recognize or **transform forms**.
- An automaton remembers only a finite amount of information.
- Information is represented by its states.
- State changes in response to inputs and may trigger outputs.
- Transition rules define how the state changes in response to inputs.
- Given a sequence of input symbols, a recognition process starts in the start state and follow the transitions in turn. Input is accepted if this process ends up in an accepting state.
- **Partial grammars for text preprocessing, tokenization, named entity recognition etc.**
Readings

- Ann’s lecture notes
- E. Bender. 100 Things You Always Wanted to Know about Linguistics But Were Afraid to Ask. NAACL-HLT 2012 tutorial. faculty.washington.edu/ebender/papers/100things.pdf