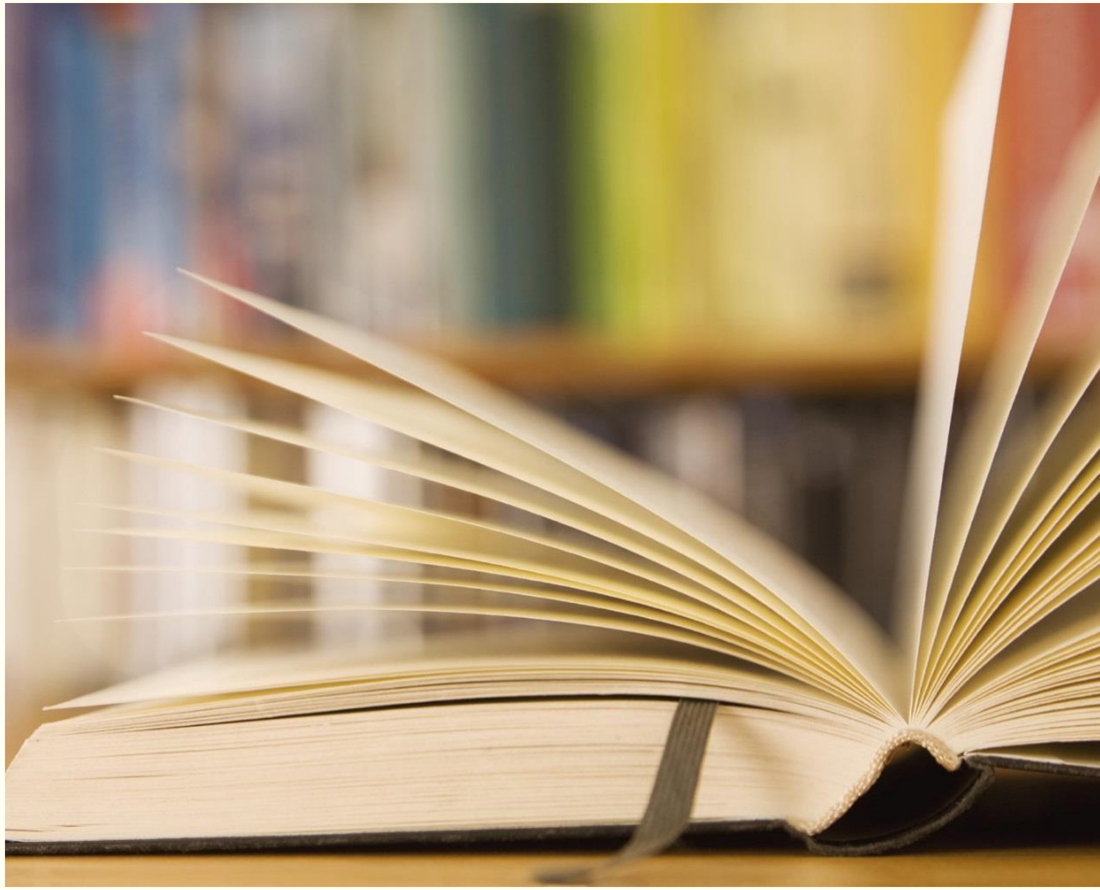
NATURAL LANGUAGE PROCESSING

Dr. G. Bharadwaja Kumar



What is Morphology?

- The study of how words are composed of morphemes (the smallest meaning-bearing units of a language)

- It analyzes the structure of words and parts of words, such as stems, root words, affixes.

- Morphology also looks at parts of speech, intonation and stress, and the ways context can change a word's pronunciation and meaning.

What is Morphology?

- Stems – core meaning units in a lexicon
- Affixes that combine with stems to modify their meanings and grammatical functions (can have multiple ones)
  - im-material
  - try-ing

Why is Morphology Important to the Lexicon?

Full listing versus Minimal Redundancy

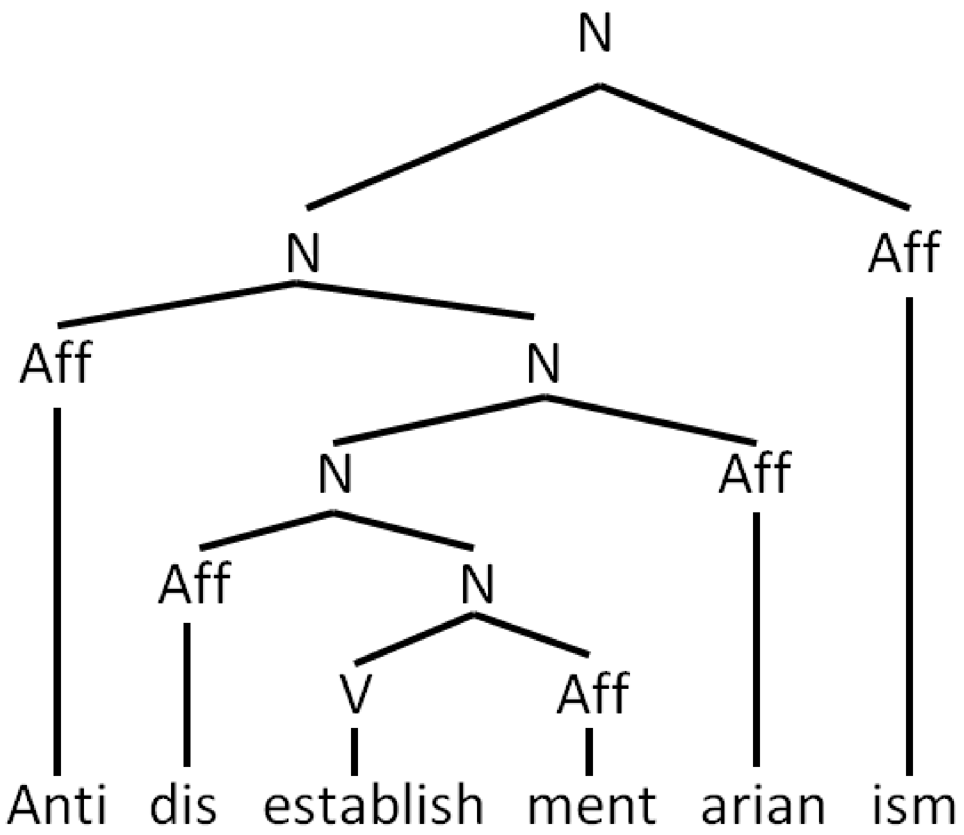
- true, truer, truest, truly, untrue, truth, truthful, truthfully, untruthfully, untruthfulness
- Untruthfulness = un- + true + -th + -ful + -ness
- These morphemes appear to be productive
- By representing knowledge about the internal structure of words and the rules of word formation, we can save space and search time.

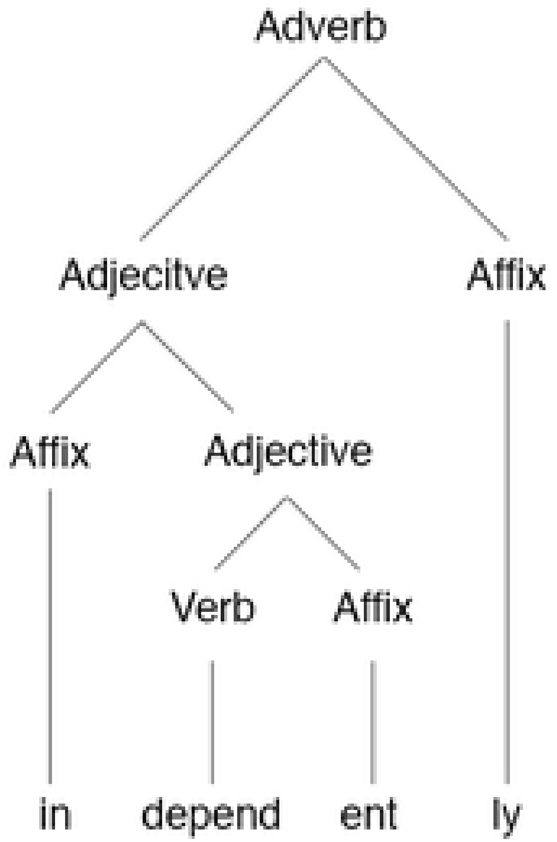
Morphological parsing contd…

- *It is quite inefficient to list all forms of noun and verb in the dictionary because the productivity of the forms.*
    - *Productive suffix*
    - *Applies to every verb*
    - *Example –ing*

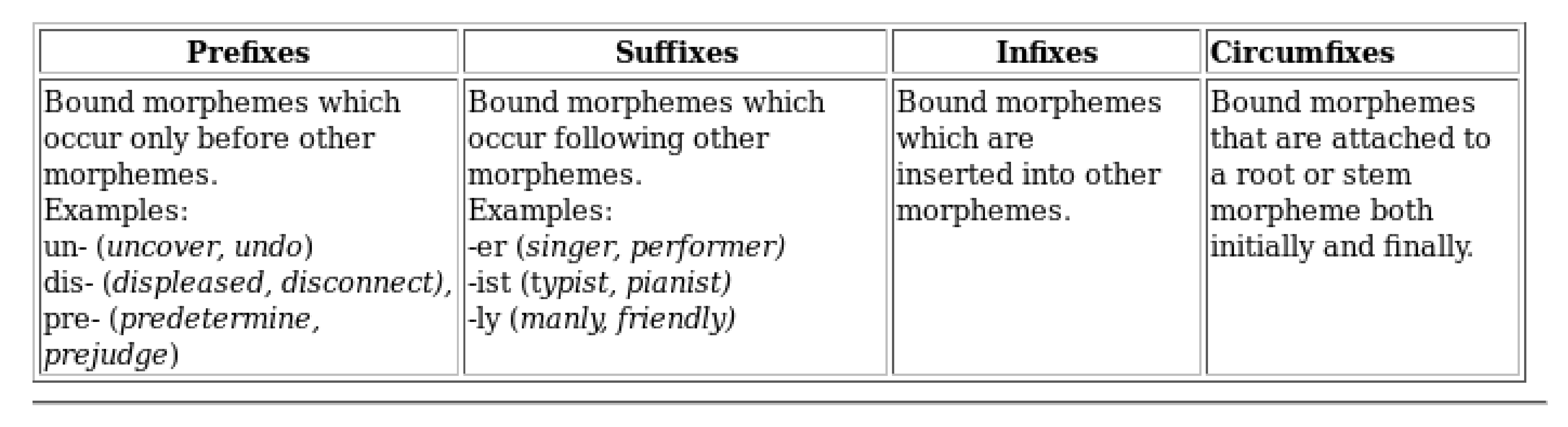
- *Morphological parsing is necessary more than just IR, but also*
    - *Machine translation*
    - *Spelling checking*
    - *Language Modeling*

Complex morphology Tree





- Free morpheme: a simple word, consisting of one morpheme
  - eg house, work, high, chair, wrap.
- They are words in themselves.
- Bound morpheme: morphemes that must be attached to another morpheme to receive meaning.
  - eg: UNKIND**NESS**
- UN- and -NESS are the bound morphemes, requiring the root KIND to form the word.



Prefixes	Suffixes	Infixes	Circumfixes
Bound morphemes which occur only before other morphemes. Examples: un- (*uncover, undo*) dis- (*displeased, disconnect*), pre- (*predetermine, prejudge*)	Bound morphemes which occur following other morphemes. Examples: -er (*singer, performer*) -ist (*typist, pianist*) -ly (*manly, friendly*)	Bound morphemes which are inserted into other morphemes.	Bound morphemes that are attached to a root or stem morpheme both initially and finally.

uncover
disconnect
singer
manly

abso-*bloody*-lutely
agoing
unreadable
fan-*bloody*-tastic

- Not all morphology is about a linear string of morphemes put together
    - sing, sang, sung and song.

Two Broad Classes of Morphology

- Inflectional Morphology
  - Combination of stem and morpheme resulting in word of same class
  - Usually fills a syntactic feature such as agreement
  - E.g., plural –s, past tense -ed

- Derivational Morphology
  - Combination of stem and morpheme usually results in a word of a different class
  - Meaning of the new word may be hard to predict
  - E.g., +ation in words such as computerization

Another classification of morphology

- *Prefixes and suffixes are often called* ***concatenative morphology.***
  - *A word is composed of a number of morphemes concatenated together*

- *A number of languages have extensive* ***non-concatenative morphology***
  - *Morphemes are combined in a more complex way*
    - *Indian Languages (Complex Sandhi Rules)*

Isolating languages

- They're composed of isolated, or free, morphemes. Free morphemes can be words on their own, such as cat or happy. Languages that are purely analytic in structure don't use any prefixes or suffixes, ever.

- Mandarin Chinese and Vietnamese are good examples of Isolating languages.

Analytic Languages

- Analytic languages use relatively little morphology

- English is not exactly isolating, but it is analytic. We have relatively little inflectional morphology, but rely on word order for a lot of information:

Synthetic language

- Synthetic languages allow affixation such that words may include two or more morphemes. These languages have bound morphemes, meaning they must be attached to another word (whereas analytic languages only have free morphemes)

- Synthetic languages include three subcategories: agglutinative, fusional, and polysynthetic.

Agglutinative Languages

- Agglutination, a grammatical process in which words are composed of a sequence of morphemes
- The key characteristic is that morphemes within words are easily parsed i.e. morpheme boundaries are easy to identify. 1:many word to morpheme ratio; 1:1 morpheme to meaning
  - Turkish, Finnish, and Japanese are among the languages that form words by agglutination.

- Fusional languages, like other synthetic languages, may have more than one morpheme per word
- However, fusional languages may have morphemes that combine multiple pieces of grammatical information; that is, there is not a clear 1 to 1 relationship between grammatical information and morphemes
- Spanish, Dravidian languages

- Polysynthetic languages often display a high degree of affixation and typically have very few free morphemes
- Additionally, words may encapsulate whole sentences

- Native American languages

Need to do Morphological Parsing

Morphological Parsing (or Stemming)

- Taking a surface input and breaking it down into its morphemes
- foxes breaks down into the morphemes fox (noun stem) and –es (plural suffix)
- rewrites breaks down into re- (prefix) and write (stem) and –s (suffix)

English Inflectional Morphology

- Word stem combines with grammatical morpheme
  - Usually produces word of same class
  - Usually serves a syntactic function (e.g., agreement)

    like → likes or liked

    bird → birds
- Nominal morphology
  - Plural forms
    - s or es
    - Irregular forms
    - Mass vs. count nouns (email or emails)
  - Possessives

Complication in Morphology

- The terms regular and irregular will be used to refer to words that follow the rules and those that don’t.

Regular (Nouns)

- Singular (cat, thrush)
- Plural (cats, thrushes)
- Possessive (cat’s thrushes’)

Irregular (Nouns)

- Singular (mouse, ox)
- Plural (mice, oxen)

- Verbal inflection
  - *Main* verbs (sleep, like, fear) are relatively regular
    - -s, ing, ed
    - And productive: Emailed, instant-messaged, faxed, homered
    - But eat/ate/eaten, catch/caught/caught
  - *Primary* (be, have, do) and *modal* verbs (can, will, must) are often irregular and not productive
    - Be: am/is/are/were/was/been/being
  - Irregular verbs few (~250) but frequently occurring
  - English verbal inflection is much simpler than e.g. Latin

Regular and Irregular Verbs

- Regulars…
  - Walk, walks, walking, walked, walked
- Irregulars
  - Eat, eats, eating, ate, eaten
  - Catch, catches, catching, caught, caught
  - Cut, cuts, cutting, cut, cut

Derivational Morphology

- Derivational morphology is the messy stuff that no one ever taught you.
  - Quasi-systematicity
  - Irregular meaning change
  - Changes of word class

English Derivational Morphology

- Word stem combines with grammatical morpheme
  - Usually produces a word of a **different** class
  - More complicated than inflectional
- Example: nominalization
  - -ize verbs → -ation nouns
  - generalize, realize → generalization, realization
  - verb → -er nouns
  - Murder, spell → murderer, speller
- Example: verbs, nouns → adjectives
  - embrace, pity→ embraceable, pitiable
  - care, wit → careless, witless

- Example: adjective → adverb
  - happy → happily
- More complicated to model than inflection
  - Less productive: *science-less, *concern-less, *go-able, *sleep-able
  - Meanings of derived terms harder to predict by rule
    - clueless, careless, nerveless

Derivational Examples

- Verb/Adj to Noun

-ation	computerize	computerization
-ee	appoint	appointee
-er	kill	killer
-ness	fuzzy	fuzziness

Derivational Examples

- Noun/Verb to Adj

-al	Computation	Computational
-able	Embrace	Embraceable
-less	Clue	Clueless

Compute

- Many paths are possible…
- Start with compute
  - Computer -> computerize -> computerization
  - Computation -> computational
  - Computer -> computerize -> computerizable
  - Compute -> computee

Parsing

- Taking a surface input and identifying its components and underlying structure
- Morphological parsing: parsing a word into stem and affixes and identifying the parts and their relationships
    - Stem and **features**:
        - goose → goose +N +SG or goose +V
        - geese → goose +N +PL
        - gooses → goose +V +3SG
    - Bracketing: indecipherable → [in [[de [cipher]] able]]

- A is a Greek prefix for "not", or "without".

- Theist / Atheist

- Non

- Non is basically a special case of "un" that gets used, erm, for no obvious pattern-based reason I've been able to work out whatsoever. You just have to learn which words take "non".

- Nonsense
- Noncompliance
- Nonpartisan

- "in"changes to "im" if the next letter is going to be "m", or "p", or "b"
- "in" changes to "il" if the next letter is going to be "l"
- "in" changes to "ir" if the next letter is going to be "r"

- So, you have:
- impossible / impolite (instead of inpossible / impolite)
- illegal (instead of inlegal)
- irregular (instead of regular

- It tends to be used with verbs, adjectives and adverbs, and abstract nouns, but not concrete nouns.

- Some examples:

- Unhappy (not happy)
- Untie (not tied)
- Unarmed (not carrying weapons)

- De / Dis

- De / dis   tend to be used with a verb to mean the "reversal".

- Dis tends to come with Latin words.

- Decode reverses what you did when you (en)code
- Deactivate reverses what you did when you activated.
- Defrost is the removal of frost, not just its absence
- Disarm is the removal of weapons, not their absence

What do we need to build a morphological parser?

- Lexicon: stems and affixes (w/corresponding pos)

- Morphotactics of the language: model of how morphemes can be affixed to a stem

- Orthographic rules: spelling modifications that occur when affixation occurs
  - in → il in context of l (in- + legal)

Morphotactics

Tells us the sequence in which morphemes may be combined.

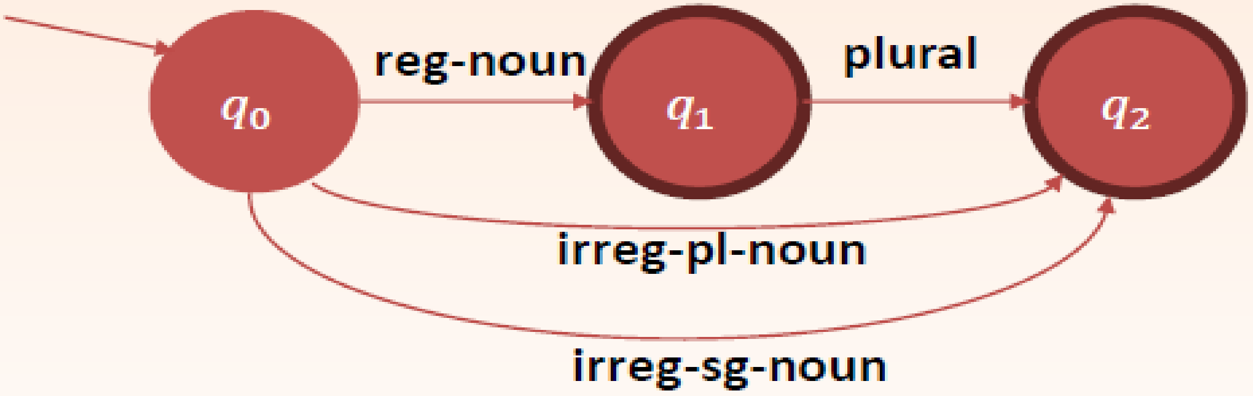
- e.g., English nouns:

reg-noun

reg-noun + -s

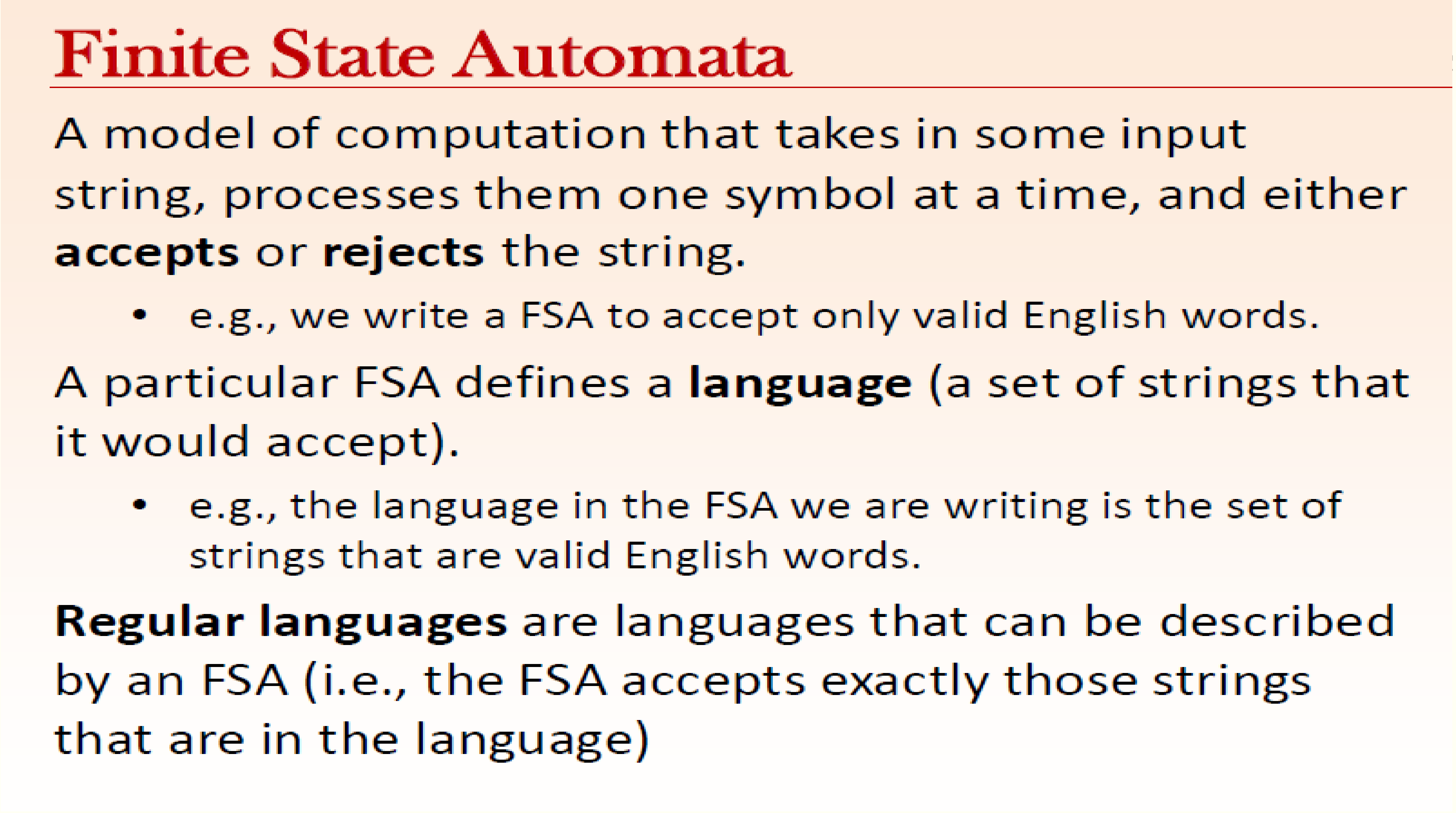
irreg-sg-noun

irreg-pl-noun



Nodes with outline represent an **accepting state**. (This is a word)

This is a representation of a **finite state automaton (FSA)**.



Finite State Automata

A model of computation that takes in some input string, processes them one symbol at a time, and either **accepts** or **rejects** the string.

- e.g., we write a FSA to accept only valid English words.

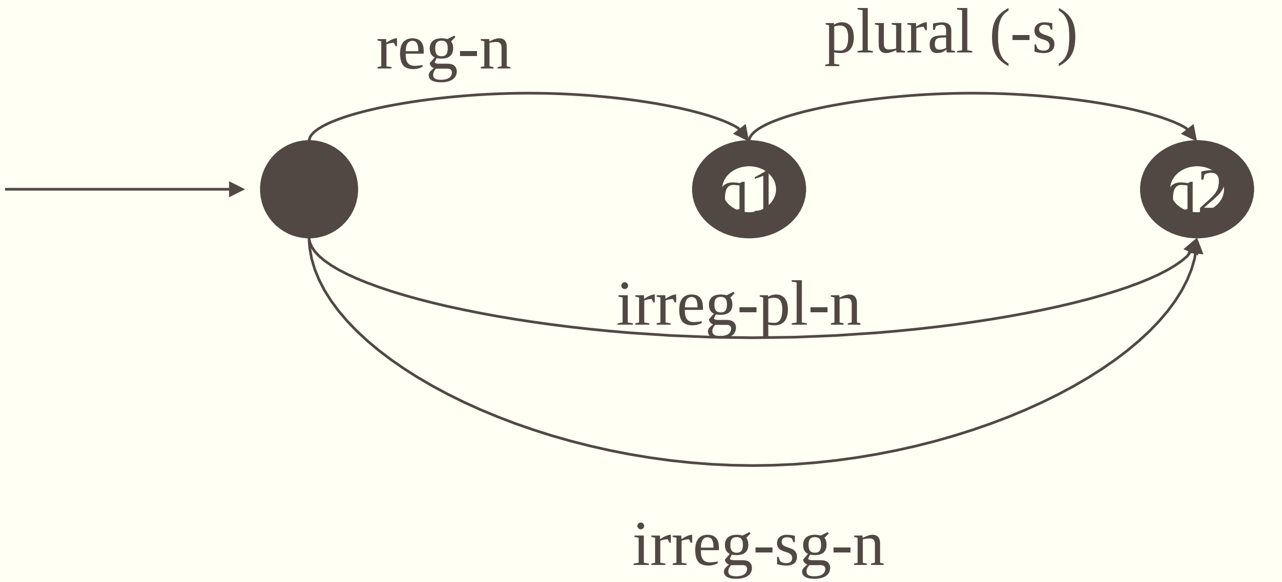
A particular FSA defines a **language** (a set of strings that it would accept).

- e.g., the language in the FSA we are writing is the set of strings that are valid English words.

Regular languages are languages that can be described by an FSA (i.e., the FSA accepts exactly those strings that are in the language)

Morphotactic Models

- English nominal inflection

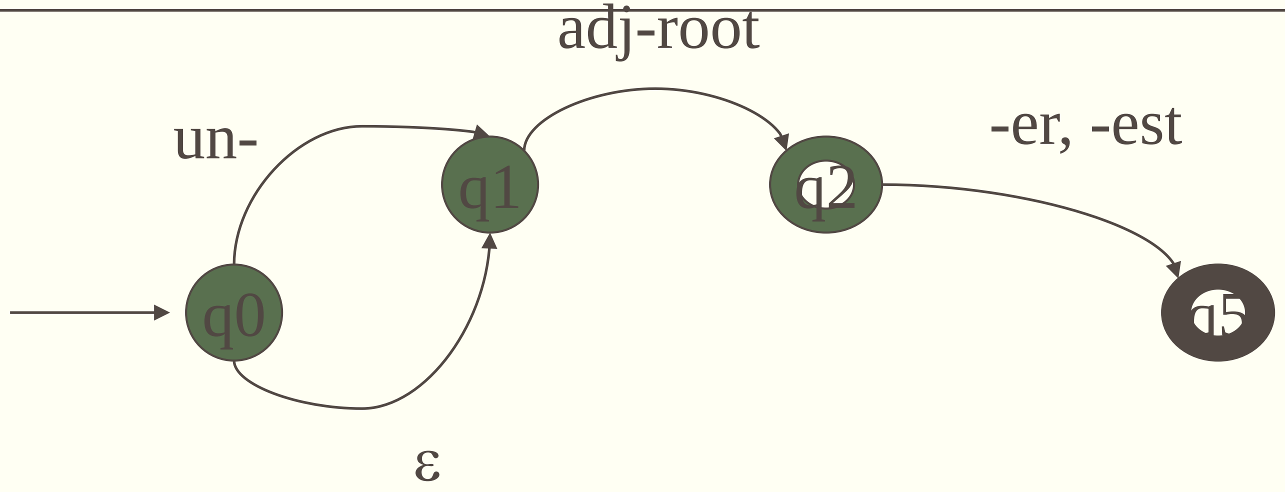


- Inputs: cats, goose, geese

English Adjectives

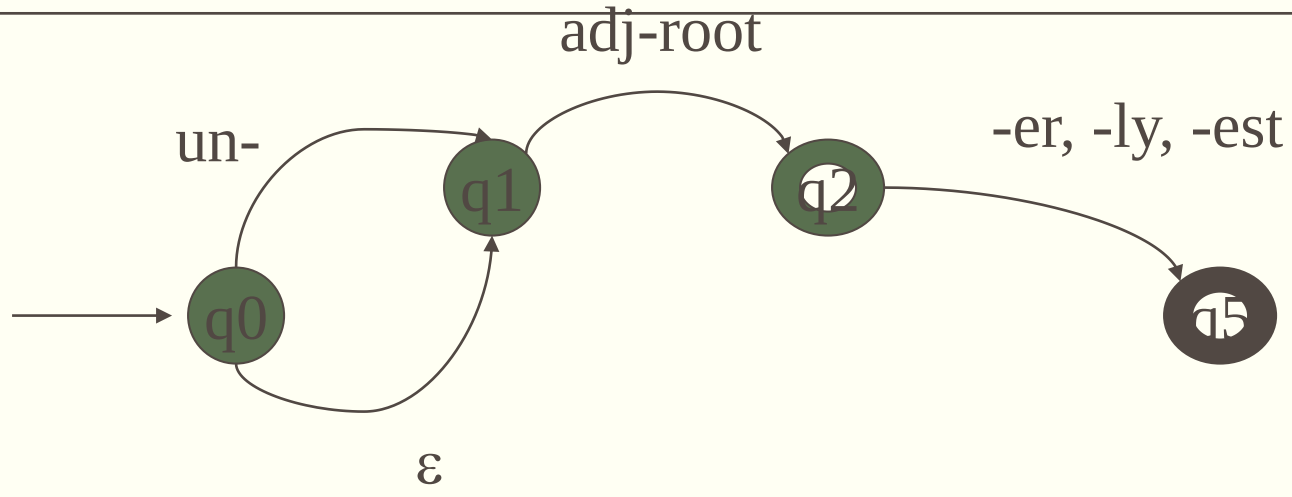
- Big, bigger, biggest
- Cool, cooler, coolest
- Red, redder, reddest
- Clear, clearer, clearest, clearly, unclear, unclearly
- Happy, happier, happiest, happily
- Unhappy, unhappier, unhappiest, unhappily
- Real, unreal, really

- Derivational morphology: adjective fragment



- Adj-root: clear, happy, real, big, red

Derivational morphology: adjective fragment



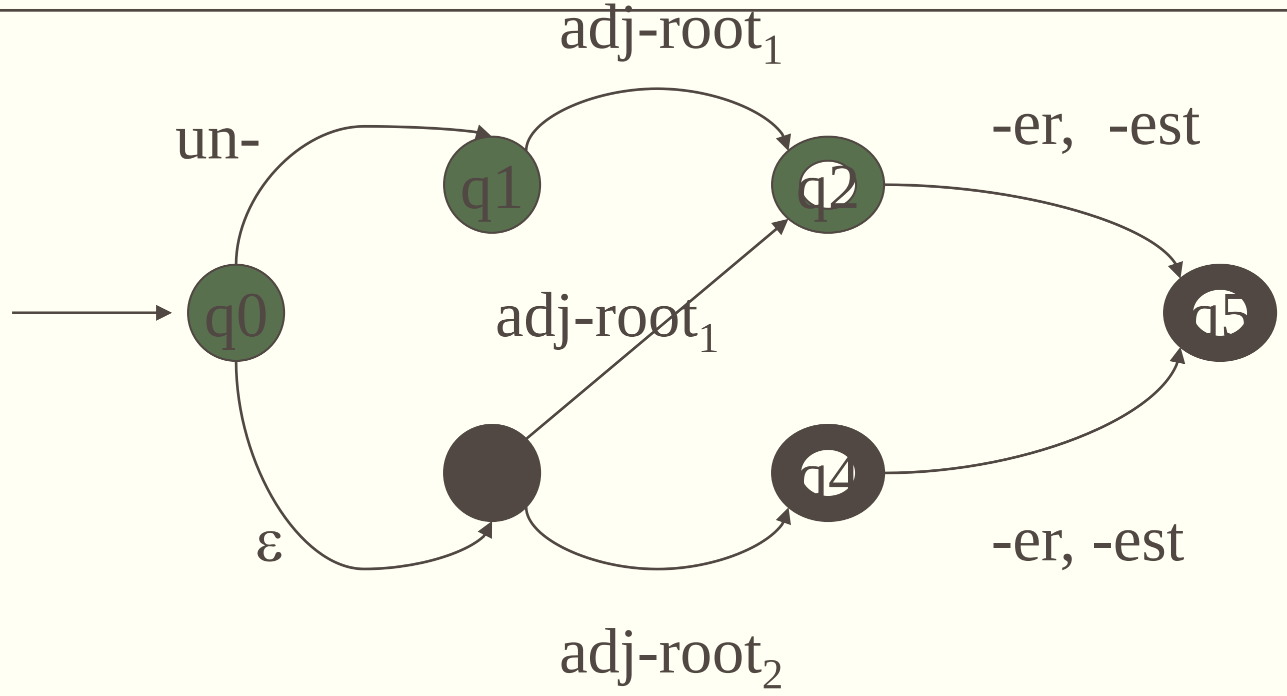
- Adj-root: clear, happy, real, big, red

•BUT: unbig, redly, realest

Antworth data on English Adjectives

- Big, bigger, biggest
- Cool, cooler, coolest, cooly
- Red, redder, reddest
- Clear, clearer, clearest, clearly, unclear, unclearly
- Happy, happier, happiest, happily
- Unhappy, unhappier, unhappiest, unhappily
- Real, unreal, really

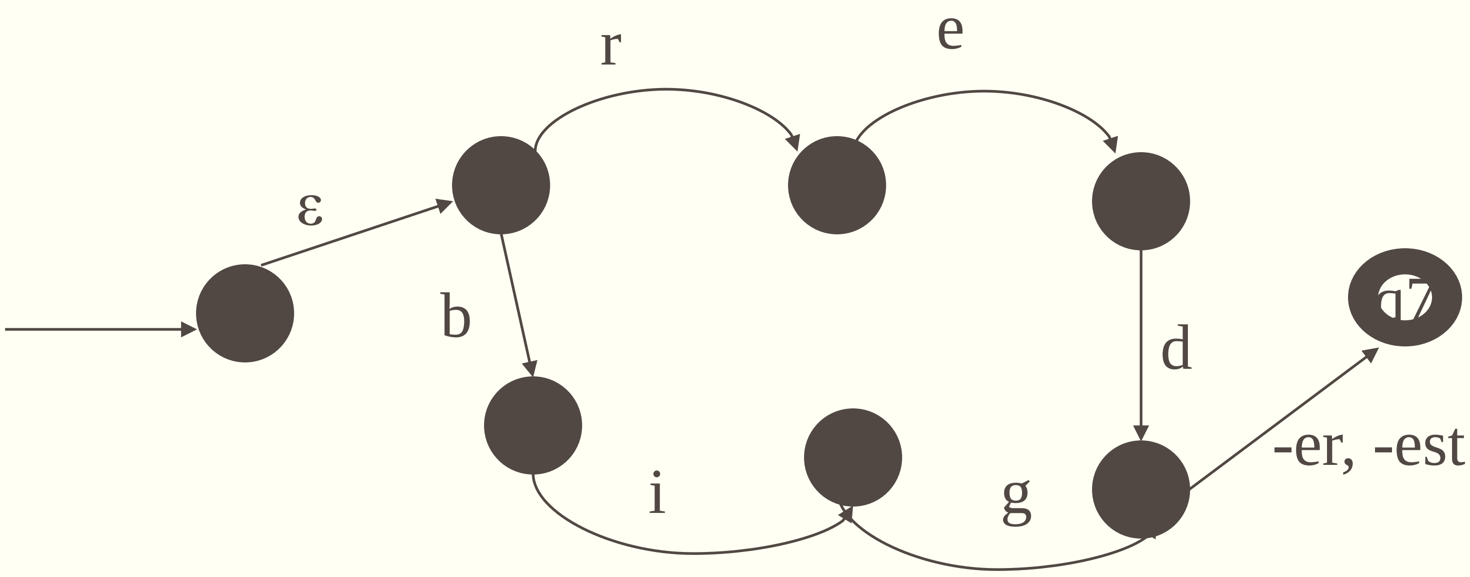
- Derivational morphology: adjective fragment



- Adj-root$_1$: clear, happy, real
- Adj-root$_2$: big, red

Using FSAs to Represent the Lexicon and Do Morphological Recognition

- Lexicon: We can expand each non-terminal in our NFSA into each stem in its class (e.g. adj_root$_2$ = {big, red}) and expand each such stem to the letters it includes (e.g. red → r e d, big → b i g)



Limitations

- To cover all of e.g. English will require very large FSAs with consequent search problems
  - Adding new items to the lexicon means recomputing the FSA
  - Non-determinism
- FSAs can only tell us whether a word is in the language or not – what if we want to know more?
  - What is the stem?
  - What are the affixes and what sort are they?
  - We used this information to build our FSA: can we get it back?

Parsing/Generation vs. Recognition

- Recognition is usually not quite what we need.
  - Usually if we find some string in the language we need to find the structure in it (parsing)
  - Or we have some structure and we want to produce a surface form (production/generation)
- Example
  - From "`cats`" to "`cat +N +PL`"

Finite State Transducers

- The simple story
  - Add another tape
  - Add extra symbols to the transitions
  - On one tape we read "`cats`", on the other we write "`cat +N +PL`"
- A finite-state transducer is a finite automaton whose state transitions are labeled with both input and output symbols. Therefore, a path through the transducer encodes a mapping from an input symbol sequence to an output symbol sequence.

Parsing with Finite State Transducers

- cats →cat +N +PL
- Kimmo Koskenniemi's two-level morphology
  - Words represented as correspondences between **lexical** level (the morphemes) and **surface** level (the orthographic word)
  - Morphological parsing :building **mappings** between the lexical and surface levels

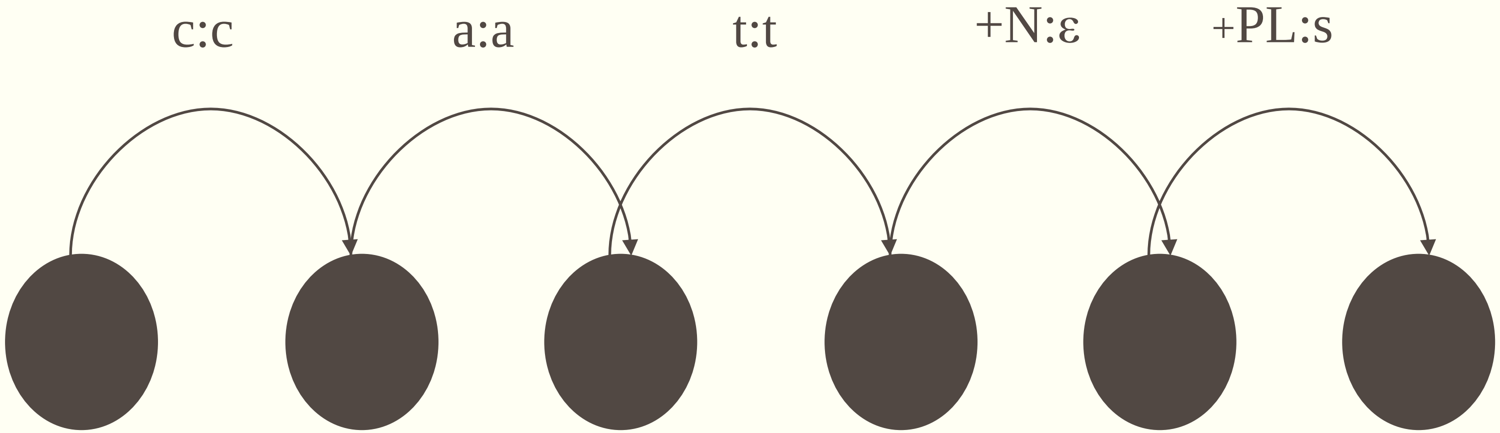
	c	a	t	+N	+PL	
	c	a	t	s		

Finite State Transducers

- FSTs map between one set of symbols and another using an FSA whose alphabet Σ is composed of pairs of symbols from input and output alphabets
- In general, FSTs can be used for
  - Translator
  - Parser/generator
  - To map between the lexical and surface levels of Kimmo's 2-level morphology

- FST is a 5-tuple consisting of
    - Q: set of states {q0,q1,q2,q3,q4}
    - Σ: an alphabet of complex symbols, each an i/o pair s.t. i ∈ I (an input alphabet) and o ∈ O (an output alphabet) and Σ is in I x O
    - q0: a start state
    - F: a set of final states in Q {q4}
    - δ(q,i:o): a transition function mapping Q x Σ to Q

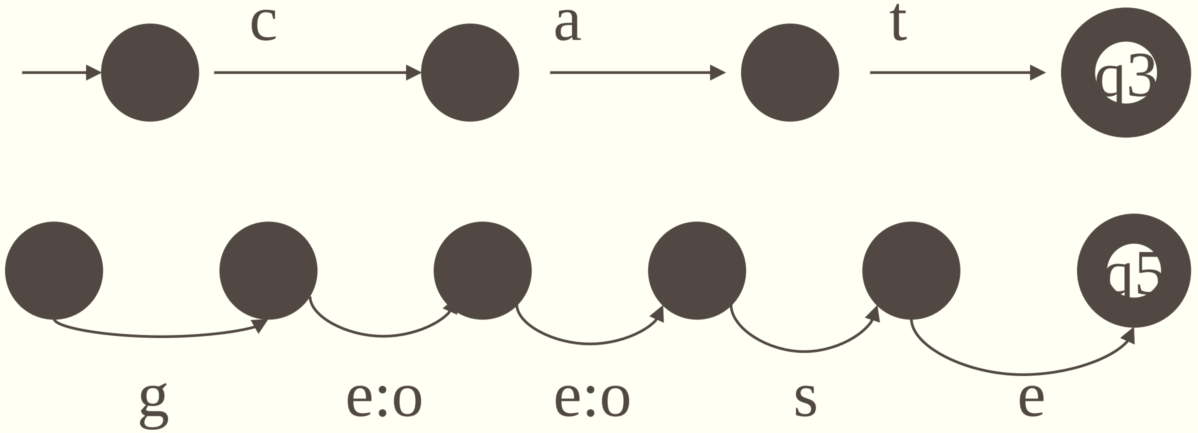
Transitions



- c:c means read a c on one tape and write a c on the other
- +N:ε means read a +N symbol on one tape and write nothing on the other
- +PL:s means read +PL and write an s

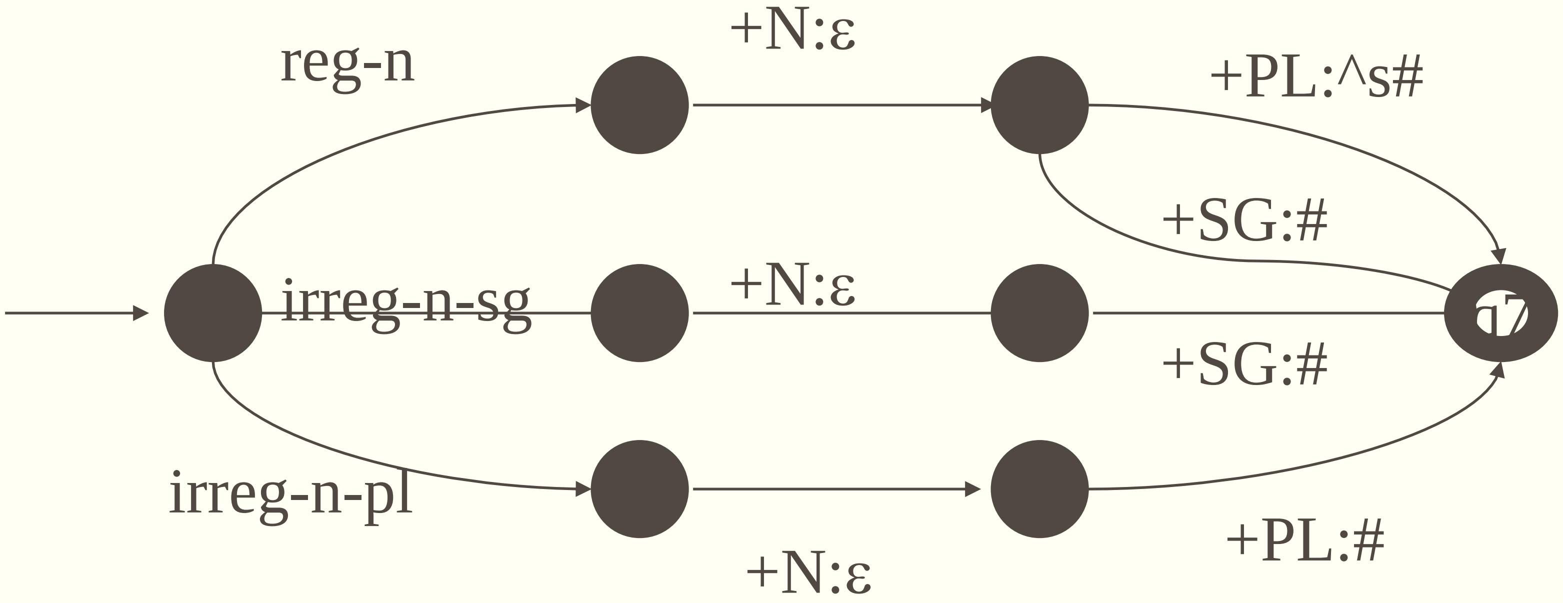
FST for a 2-level Lexicon

- E.g.



Reg-n	Irreg-pl-n	Irreg-sg-n
c a t	g o:e o:e s e	g o o s e

FST for English Nominal Inflection



Combining (cascade or composition) this FSA with FSAs for each noun type replaces e.g. reg-n with every regular noun representation in the lexicon

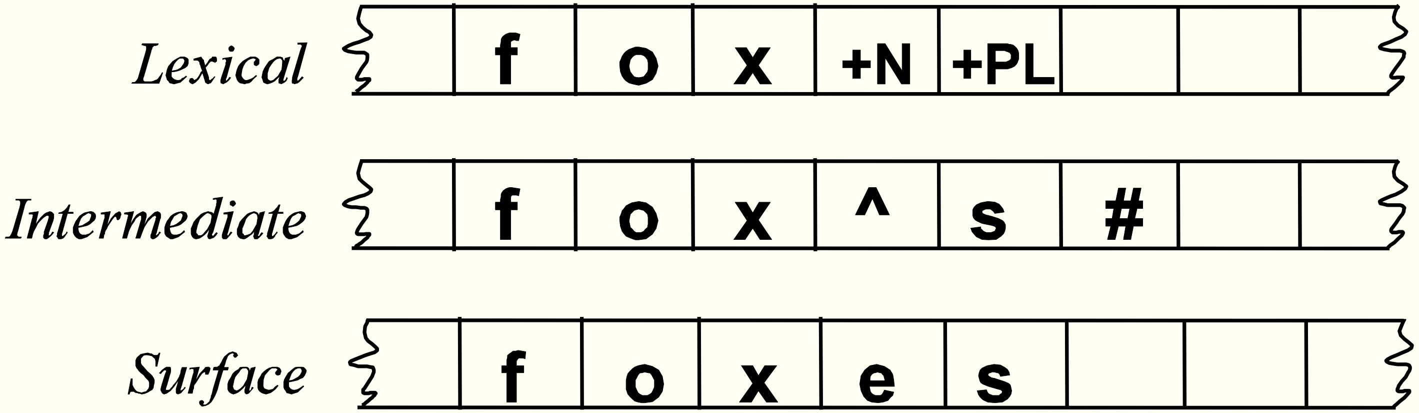
The Gory Details

- Of course, its not as easy as
  - `“cat +N +PL” <-> “cats”`
- Or even dealing with the irregulars geese, mice and oxen
- But there are also a whole host of spelling/pronunciation changes that go along with inflectional changes

Multi-Tape Machines

- To deal with this we can simply add more tapes and use the output of one tape machine as the input to the next
- So to handle irregular spelling changes we'll add intermediate tapes with intermediate symbols

Multi-Level Tape Machines



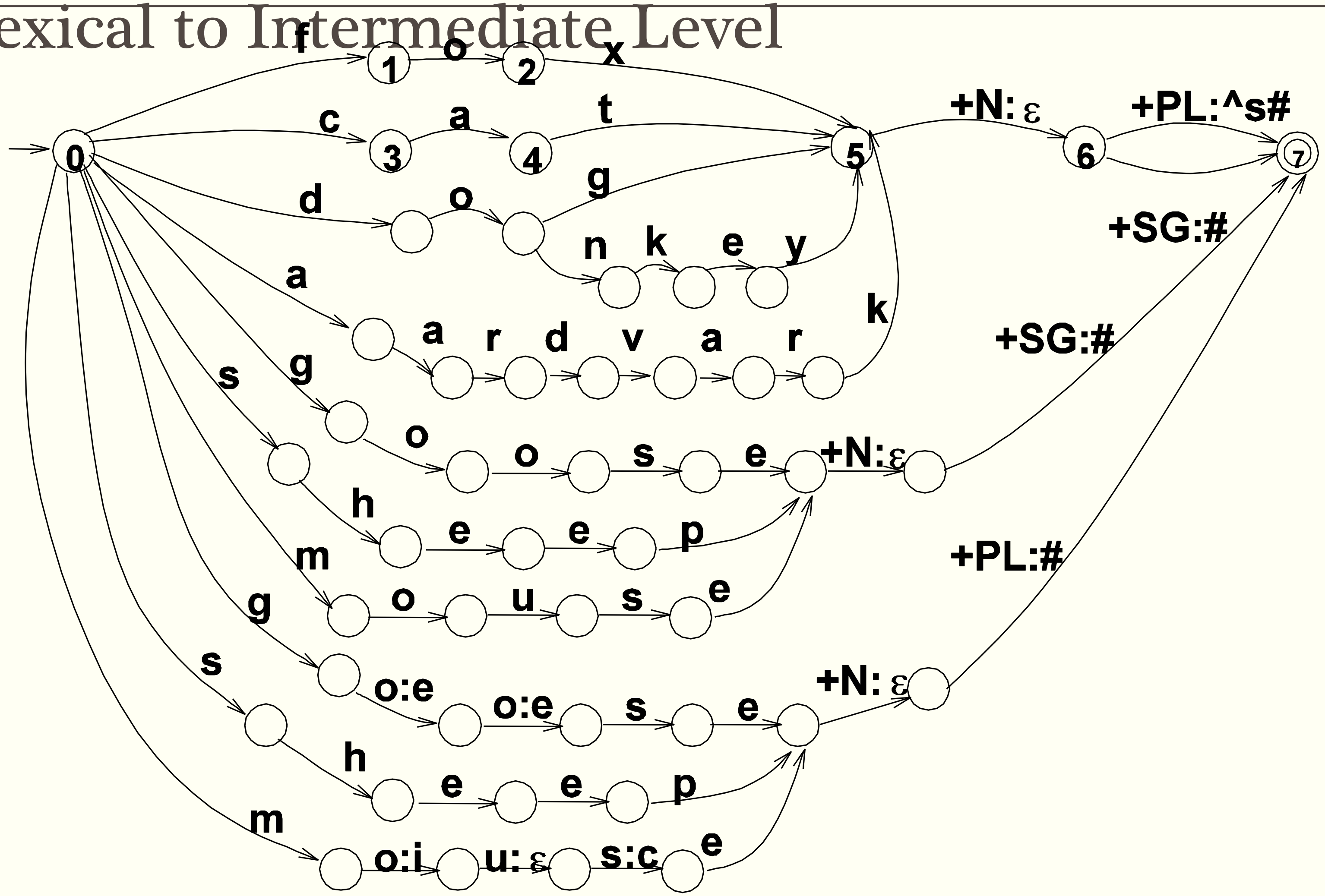
- We use one machine to transduce between the lexical and the intermediate level, and another to handle the spelling changes to the surface tape

Orthographic Rules and FSTs

- Define additional FSTs to implement rules such as consonant doubling (beg → begging), ‘e’ deletion (make → making), ‘e’ insertion (watch → watches), etc.

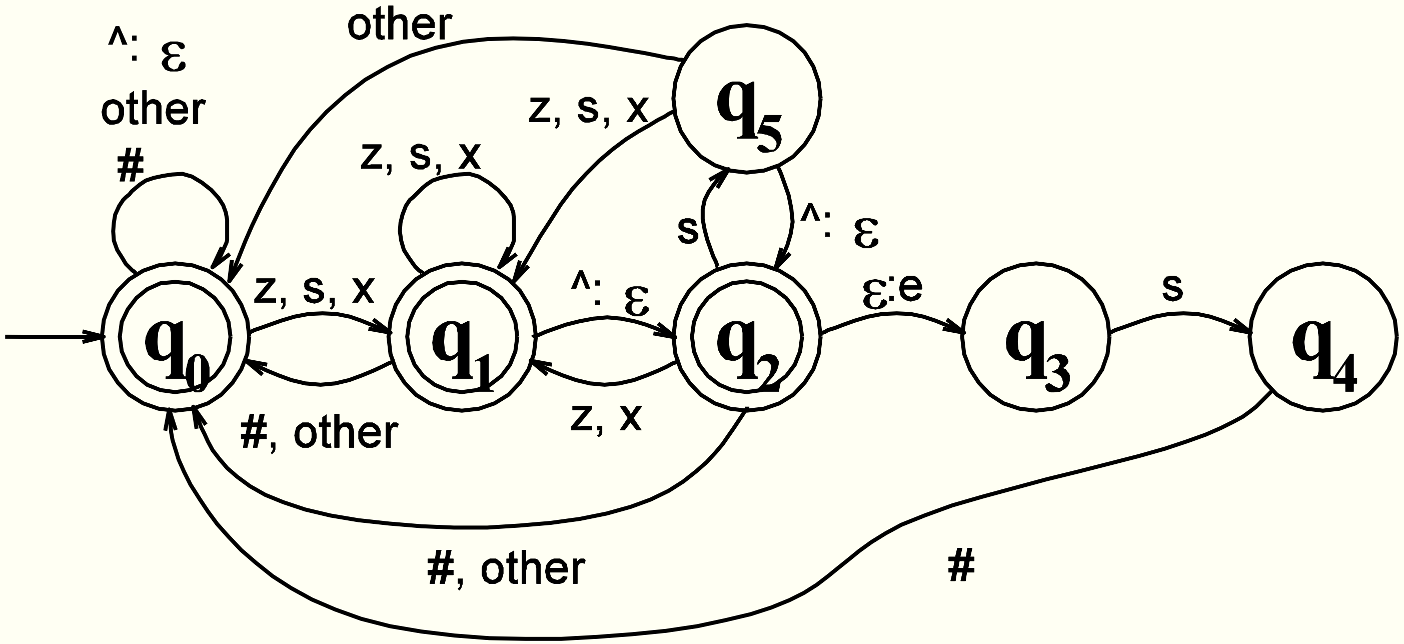
Lexical	f	o	x	+N	+PL	
Intermediate	f	o	x	^	s	#
Surface	f	o	x	e	s	

Lexical to Intermediate Level



Intermediate to Surface

- The add an “e” rule as in `fox^s# <-> foxes`



Note

- A key feature of this machine is that it doesn't do anything to inputs to which it doesn't apply.
- Meaning that they are written out unchanged to the output tape.

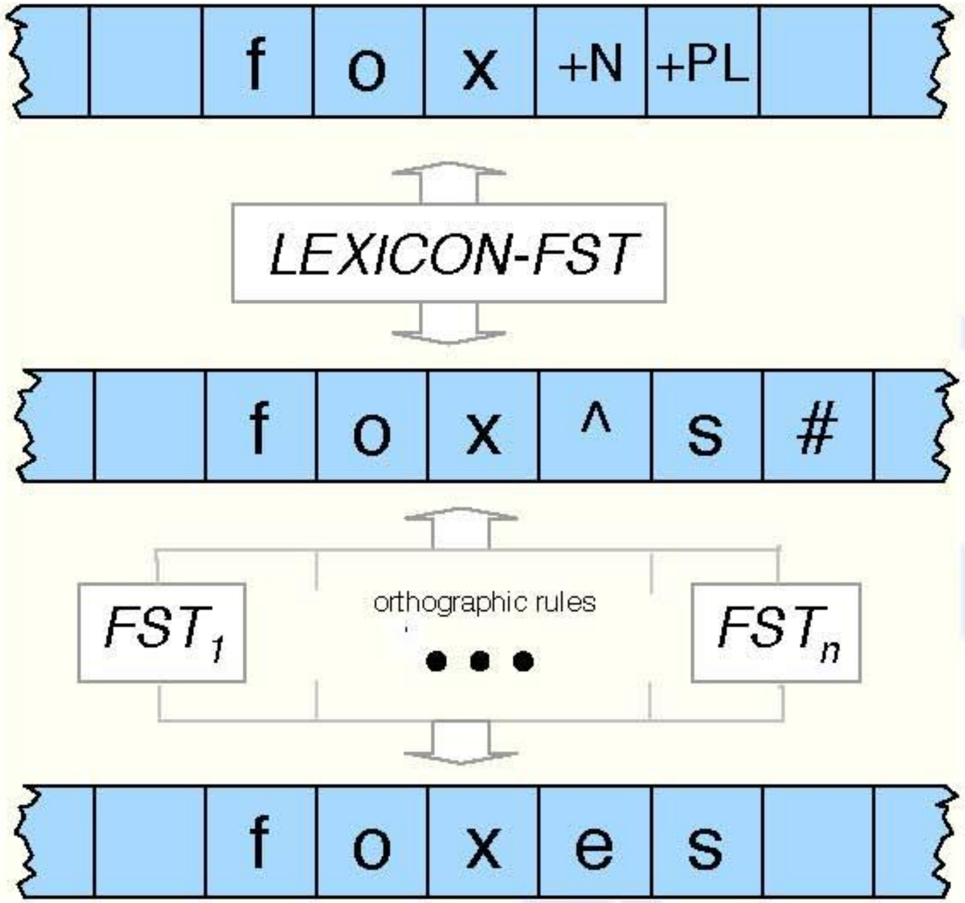


Figure 3.19 Generating or parsing with FST lexicon and rules

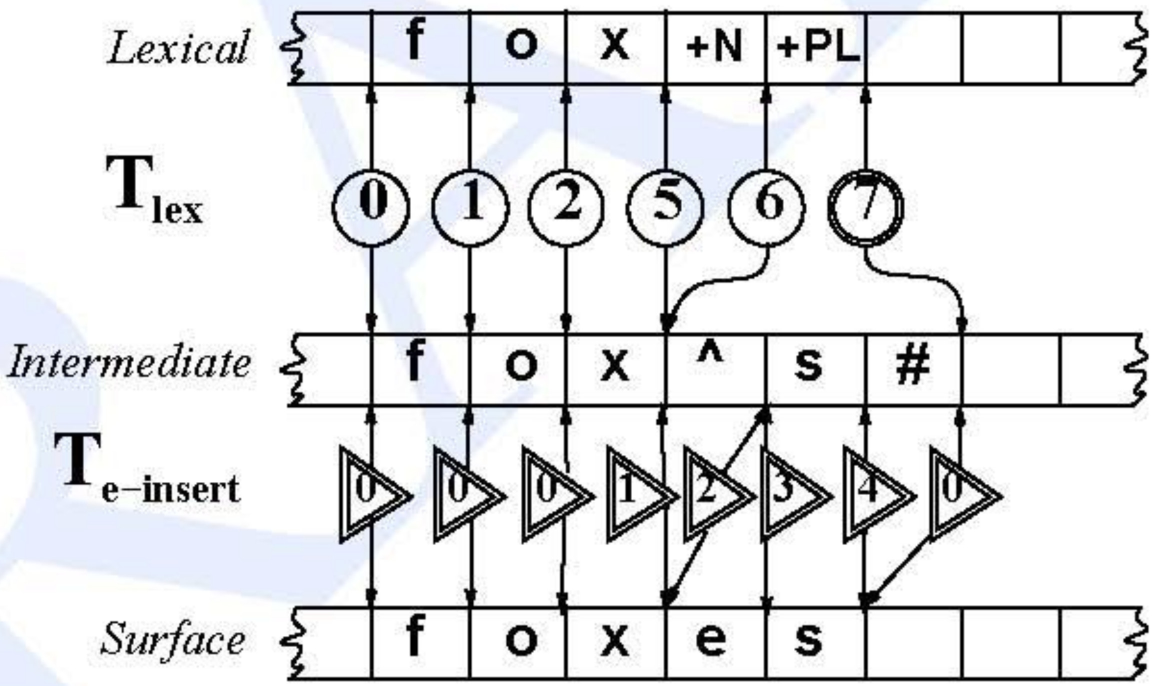


Figure 3.20 Accepting *foxes*: The lexicon transducer T_{lex} from Fig. 3.14 cascaded with the E-insertion transducer in Fig. 3.17.

- Note: These FSTs can be used for generation as well as recognition by simply exchanging the input and output alphabets (e.g. ^s#:+PL)

Summing Up

- FSTs provide a useful tool for implementing a standard model of morphological analysis, Kimmo's two-level morphology
  - Key is to provide an FST for each of multiple levels of representation and then to combine those FSTs using a variety of operators (cf AT&T FSM Toolkit)
  - Other (older) approaches are still widely used, e.g. the rule-based Porter Stemmer