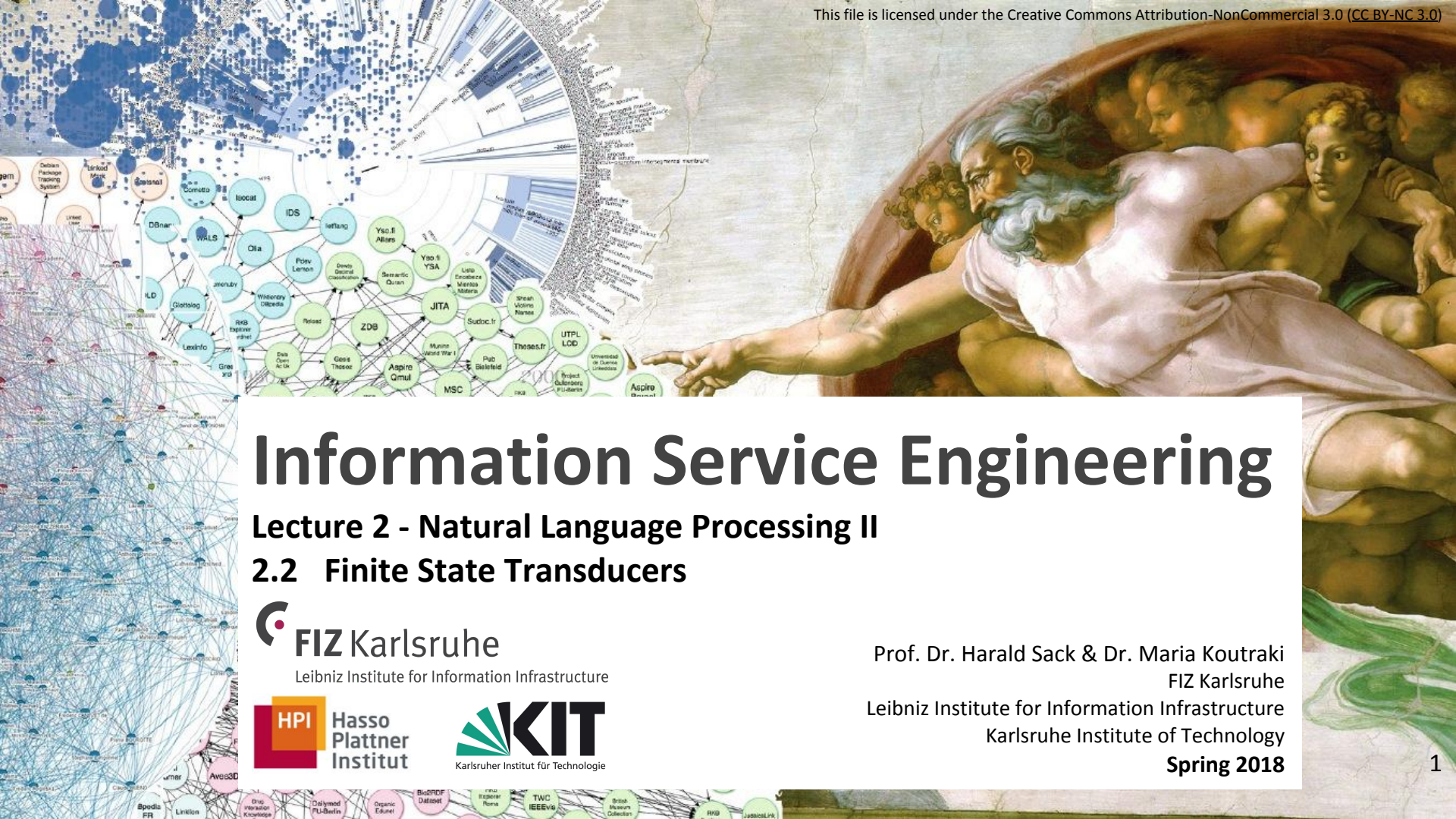




Information Service Engineering

Lecture 2 - Natural Language Processing II

2.2 Finite State Transducers



FIZ Karlsruhe

Leibniz Institute for Information Infrastructure



Hasso
Plattner
Institut



Karlsruher Institut für Technologie

Prof. Dr. Harald Sack & Dr. Maria Koutraki
FIZ Karlsruhe

Leibniz Institute for Information Infrastructure
Karlsruhe Institute of Technology
Spring 2018

Information Service Engineering

Lecture 2: Natural Language Processing II

2.1 Finite State Automata

2.2 Finite State Transducers

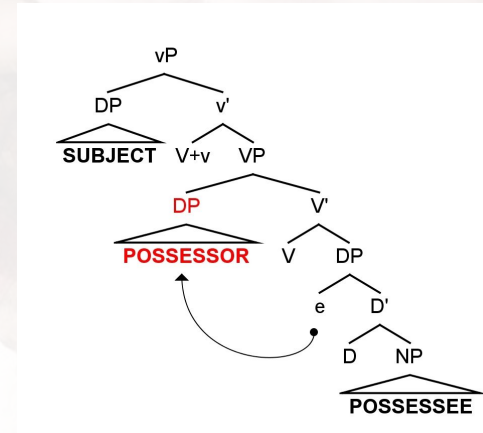
2.3 Language Model and N-Grams

2.4 Language Model and N-Grams II

2.5 Language Model and N-Grams III

2.6 Part-of-Speech Tagging

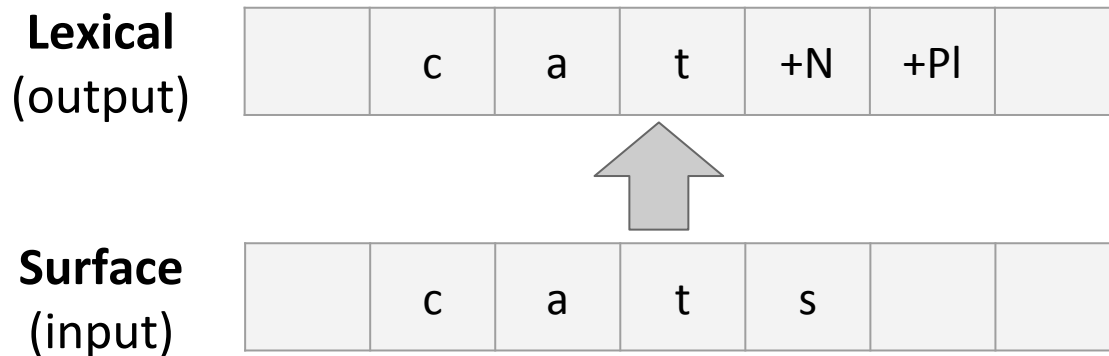
2.7 Part-of-Speech Tagging II



Recognition vs. Analysis

Information Service Engineering - Lecture 2 Natural Language Processing II - 2.2 Finite State Transducers

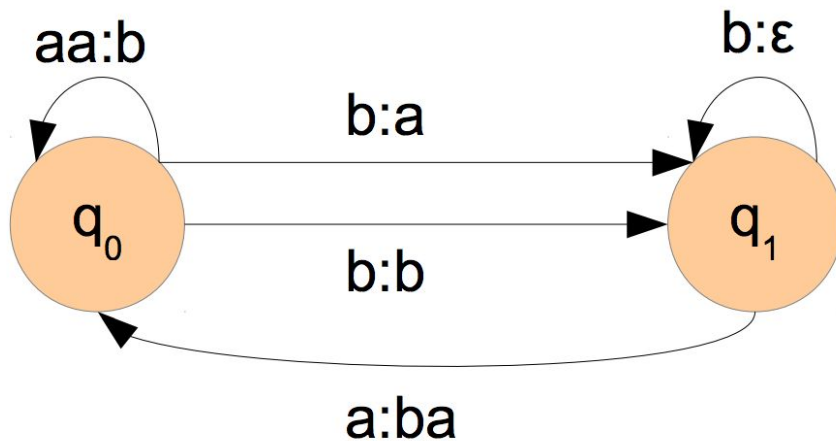
- FSAs can recognize (**accept**) a string, but they don't tell us its internal structure.
- We need a machine that maps (**transduces**) the input string into an output string that encodes its structure:



Finite State Transducer (FST)

Information Service Engineering - Lecture 2 Natural Language Processing II - 2.2 Finite State Transducers

- A Finite State Transducer (FST) maps between **two sets of symbols**
- **2-tape FSA** that **recognizes** or **generates** pairs of strings
- A FST defines relations between sets of strings



Formalization of the FST

Information Service Engineering - Lecture 2 Natural Language Processing II - 2.2 Finite State Transducers

- A finite state transducer $T = (Q, \Sigma, \Delta, q_0, F, \delta, \sigma)$ consists of:
 - Q : finite set $\{q_0, q_1, q_2, \dots, q_{N-1}\}$ of N states
 - Σ : finite set of input symbols
 - Δ : **finite set of output symbols**
 - q_0 : the designated start state
 - F : the set of final states, $F \subseteq Q$
 - $\delta(q, i)$: the transition function
 $\delta: Q \times \Sigma \rightarrow 2^Q, \delta(q, i) = Q' \text{ for } q \in Q, Q' \subseteq Q, i \in \Sigma$
 - $\sigma(q, i)$: **the output function**
 $\sigma: Q \times \Sigma \rightarrow \Delta^*, \sigma(q, i) = \omega \text{ for } q \in Q, i \in \Sigma, \omega \in \Delta^*$

Formalization of the FST

Information Service Engineering - Lecture 2 Natural Language Processing II - 2.2 Finite State Transducers

- A **finite state transducer** $T = L_{in} \times L_{out}$ defines a relation between two regular languages L_{in} and L_{out} :
- $L_{in} = \{\text{cat, cats, fox, foxes, ...}\}$
- $L_{out} = \{\text{cat+N+sg, cat+N+pl, fox+N+sg, fox+N+PL ...}\}$
- $T = \{ \langle \text{cat, cat+N+sg} \rangle, \langle \text{cats, cat+N+pl} \rangle, \langle \text{fox, fox+N+sg} \rangle, \langle \text{foxes, fox+N+pl} \rangle \}$

- FST as **recognizer**
 - Takes a pair of strings and accepts or rejects them
- FST as **generator**
 - Outputs a pair of strings for a language (i.e., switch L_{in} and L_{out})
- FST as **translator**
 - Reads a string and outputs another string
 - **Morphological parsing:** letters (input); morphemes (output)
- FST as **relater**
 - Computes relations between sets

FSTs for Morphological Parsing

Information Service Engineering - Lecture 2 Natural Language Processing II - 2.2 Finite State Transducers

- Two tapes
 - Upper (**lexical**) tape: output alphabet Δ
 - cat +N +Pl
 - Lower (**surface**) tape: Input alphabet Σ
 - cats

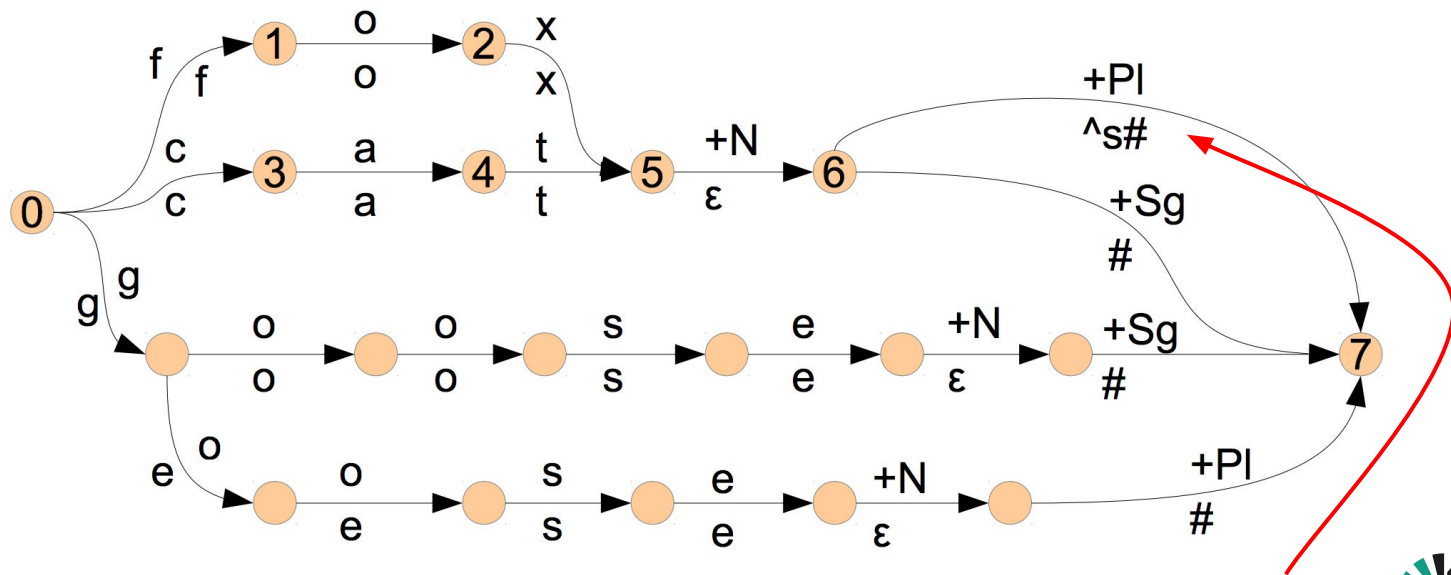
Lexical		c	a	t	+N	+Pl	
---------	--	---	---	---	----	-----	--

Surface		c	a	t	s		
---------	--	---	---	---	---	--	--

FSTs for Morphological Parsing

Information Service Engineering - Lecture 2 Natural Language Processing II - 2.2 Finite State Transducers

- goose/**geese**: g:g o:e o:e s:s e:e
 - Feasible pairs (e.g., o:e) vs. default pairs (g:g)



- We indicate morpheme boundaries (^) and word boundaries (#)

FSTs and Orthographic Rules

Information Service Engineering - Lecture 2 Natural Language Processing II - 2.2 Finite State Transducers

- English often requires spelling changes at morpheme boundaries
- Introduction of **orthographic rules**, as e.g.

Name	Orthographic Rule	Example
Consonant doubling	Consonant doubled before -ing/-ed	beg / begging
E deletion	Silent e dropped before -ing and -ed	make / making
E insertion	E added after -s, -z, -x, -cg, -sh before -s	fox / foxes
Y replacement	-y changes to -ie before -s, -i before -ed	try / tries
K insertion	Verbs ending with vowel + -c ad -k	panic / panicked

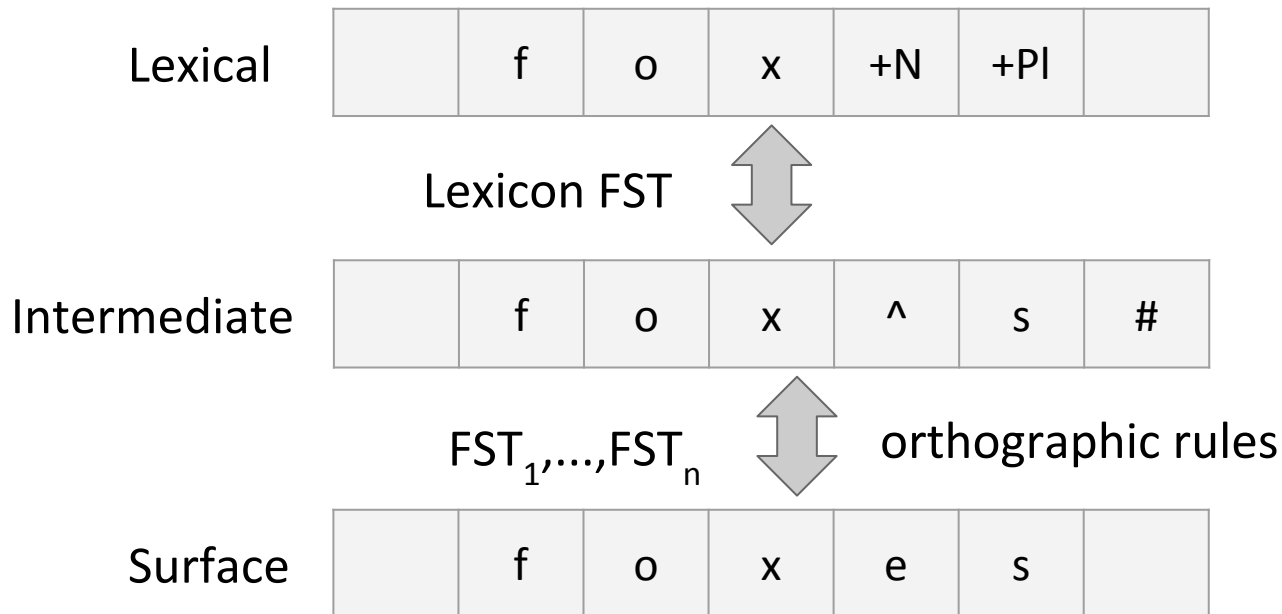
Intermediate Representations

Information Service Engineering - Lecture 2 Natural Language Processing II - 2.2 Finite State Transducers

- English plural -s:
 - cat $\Rightarrow$ cats, dog $\Rightarrow$ dogs
 - but: fox $\Rightarrow$ foxes, buzz $\Rightarrow$ buzzes
- We define an **intermediate representation** which captures **morpheme boundaries (^)** and **word boundaries (#)**:
 - **Lexicon:** cat+N+PL fox+N+PL
 - **Intermediate representation:** cat^s# fox^s#
 - **Surface string:** cats foxes
- **Intermediate-to-Surface Spelling Rule:**
If plural 's' follows a morpheme ending in 'x','z' or 's', *insert 'e'*.

FST for Orthographic Rules

Information Service Engineering - Lecture 2 Natural Language Processing II - 2.2 Finite State Transducers



Porter Stemmer

Information Service Engineering - Lecture 2 Natural Language Processing II - 2.2 Finite State Transducers

- Popular for information retrieval and text categorization tasks based on a **series of simple cascade rules**
 - **ATIONAL** → **ATE** (e.g. relational → relate)
 - **ING** → **ε** (e.g. motoring → motor)
 - **SSES** → **SS** (e.g. grasses → grass)
- But it commits many errors:
 - ORGANIZATION → ORGAN
 - DOING → DOE
 - POLICY → POLICE
 - ...

<https://tartarus.org/martin/PorterStemmer/>

A simple ELIZA Dialogue

Information Service Engineering - Lecture 2 Natural Language Processing II - 2.2 Finite State Transducers

- Computer: TELL ME MORE ABOUT YOUR FAMILY
- Human: My mother takes care of me.
- Computer: WHO ELSE IN YOUR FAMILY TAKES CARE OF YOU
- Human: My father.
- Computer: YOUR FATHER
- Human: You are like my father in some ways.
- Computer: WHAT RESEMBLANCE DO YOU SEE
- Human: You are not very aggressive but I think you don't want me to notice that.
- Computer: WHAT MAKES YOU THINK I AM NOT VERY AGGRESSIVE
- Human: You don't argue with me.
- Computer: WHY DO YOU THINK I DON'T ARGUE WITH YOU
- Human: You are afraid of me.

<http://www.masswerk.at/elizabot/>

ELIZA as an FST Cascade

Information Service Engineering - Lecture 2 Natural Language Processing II - 2.2 Finite State Transducers

<http://www.masswerk.at/elizabot/>

- Human: You don't argue with me.
- Computer: WHY DO YOU THINK I DON'T ARGUE WITH YOU
- 1. Replace **you** with **I** and **me** with **you**:
 - I DON'T ARGUE WITH **YOU**.
- 2. Replace <...> with **Why do you think <...>**:
 - **WHY DO YOU THINK I DON'T ARGUE WITH YOU**

**Language Model and
N-Grams**

Next Lecture...

16

Next Lecture...

Picture References:

- P.16: Pieter Brueghel the Elder, Das Schlaraffenland (1567),
[https://commons.wikimedia.org/wiki/File:Pieter Bruegel d. %C3%84. 037.jpg](https://commons.wikimedia.org/wiki/File:Pieter_Bruegel_d._%C3%84.037.jpg)