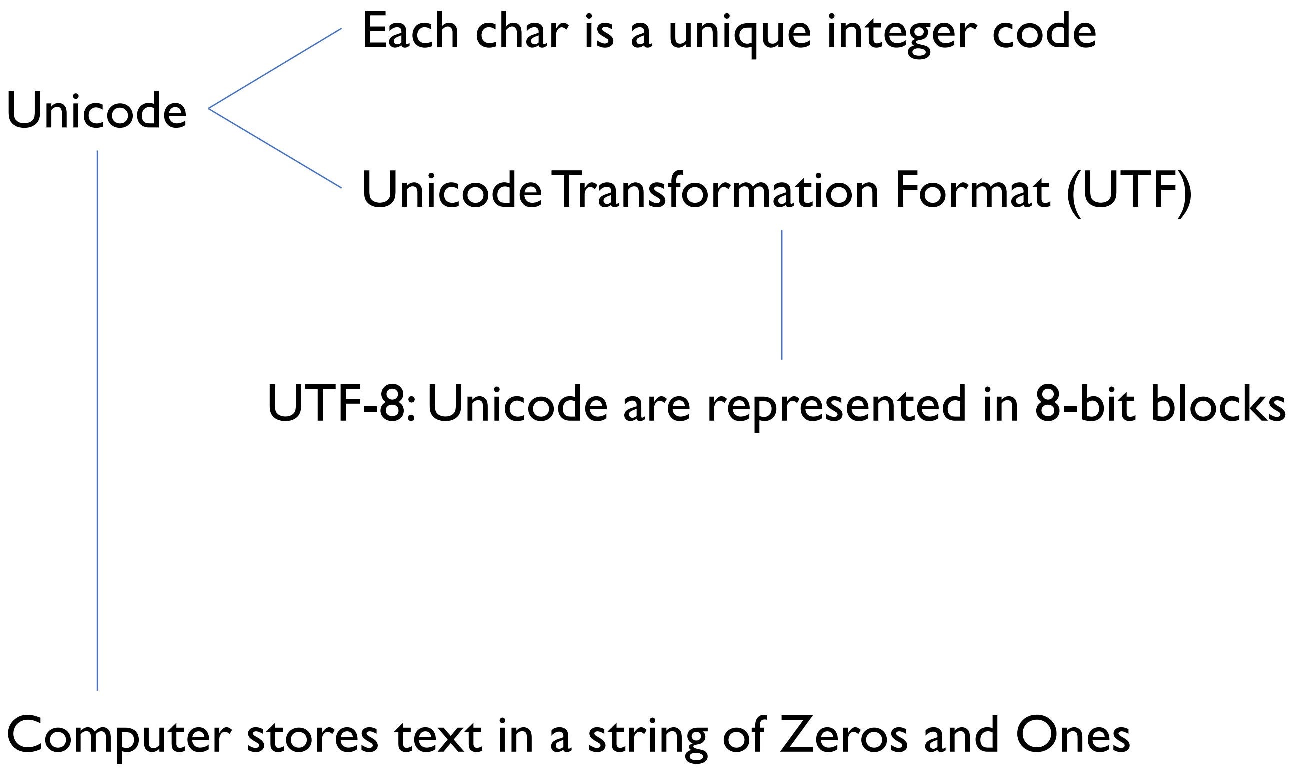
CE4045 CZ4045 SC4002
Natural Language Processing

Review of first half topics

Dr. Sun Aixin

Summary: UTF-8

For your info



Unicode

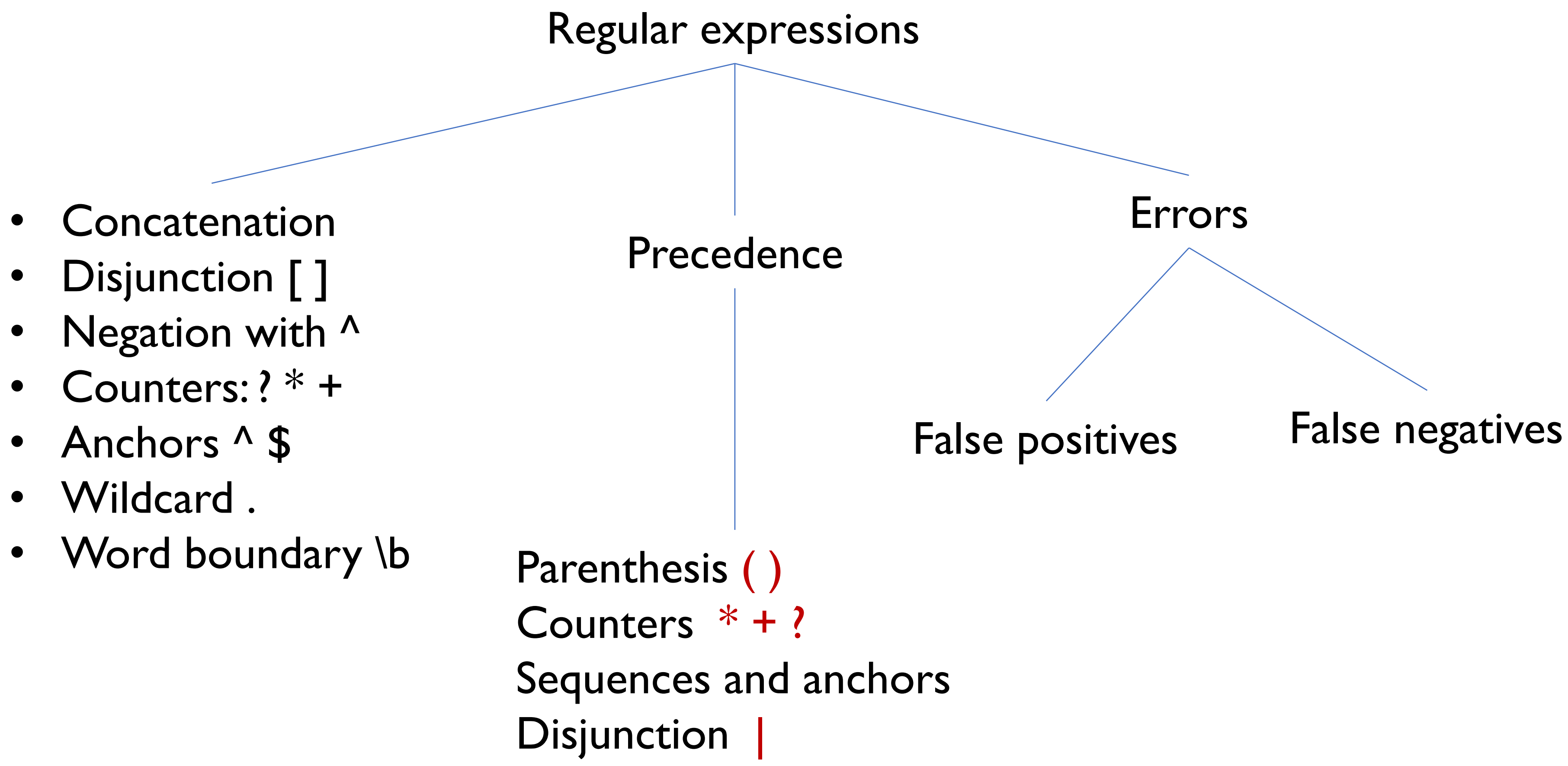
Each char is a unique integer code

Unicode Transformation Format (UTF)

UTF-8: Unicode are represented in 8-bit blocks

Computer stores text in a string of Zeros and Ones

Summary: Regular expressions



Regular expressions

- Concatenation
- Disjunction [ ]
- Negation with ^
- Counters: ? * +
- Anchors ^ $
- Wildcard .
- Word boundary \b

Precedence

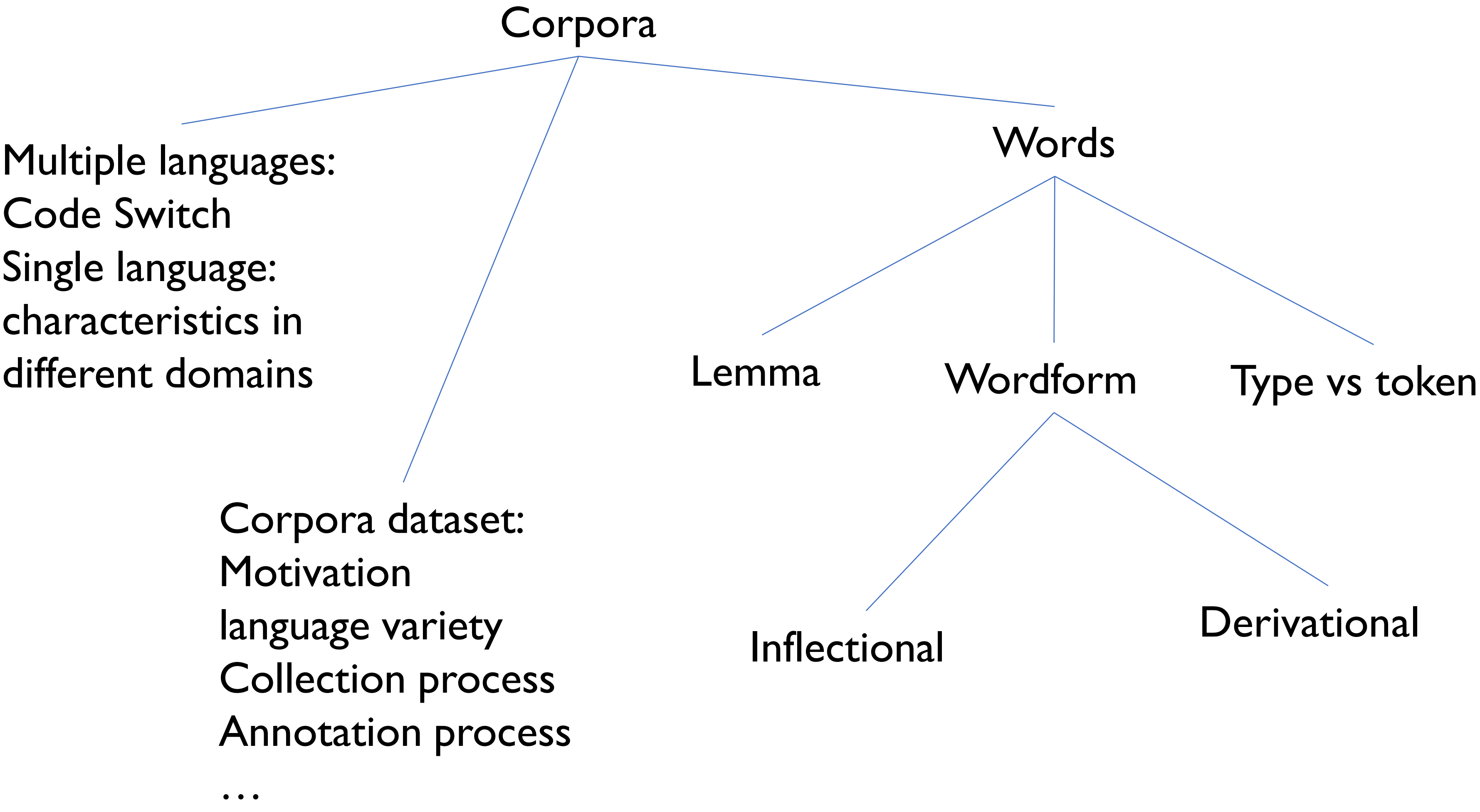
Parenthesis ( )
Counters * + ?
Sequences and anchors
Disjunction |

Errors

False positives

False negatives

Summary: Text Normalization



Corpora

Multiple languages:
Code Switch
Single language:
characteristics in
different domains

Corpora dataset:
Motivation
language variety
Collection process
Annotation process
…

Words

Lemma

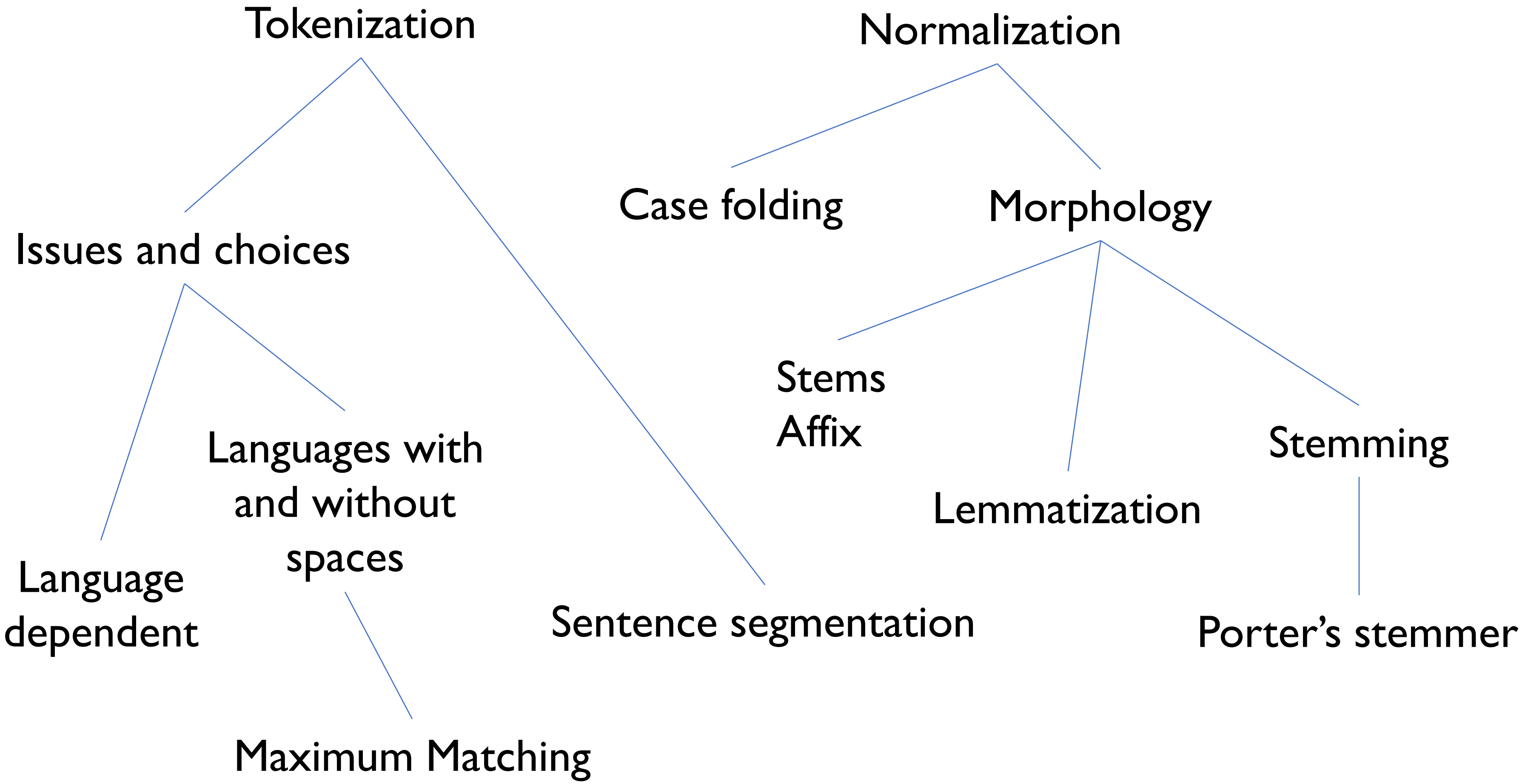
Wordform

Type vs token

Inflectional

Derivational

Summary: Text Normalization



Tokenization

Issues and choices

Language dependent

Languages with and without spaces

Maximum Matching

Sentence segmentation

Normalization

Case folding

Morphology

Stems

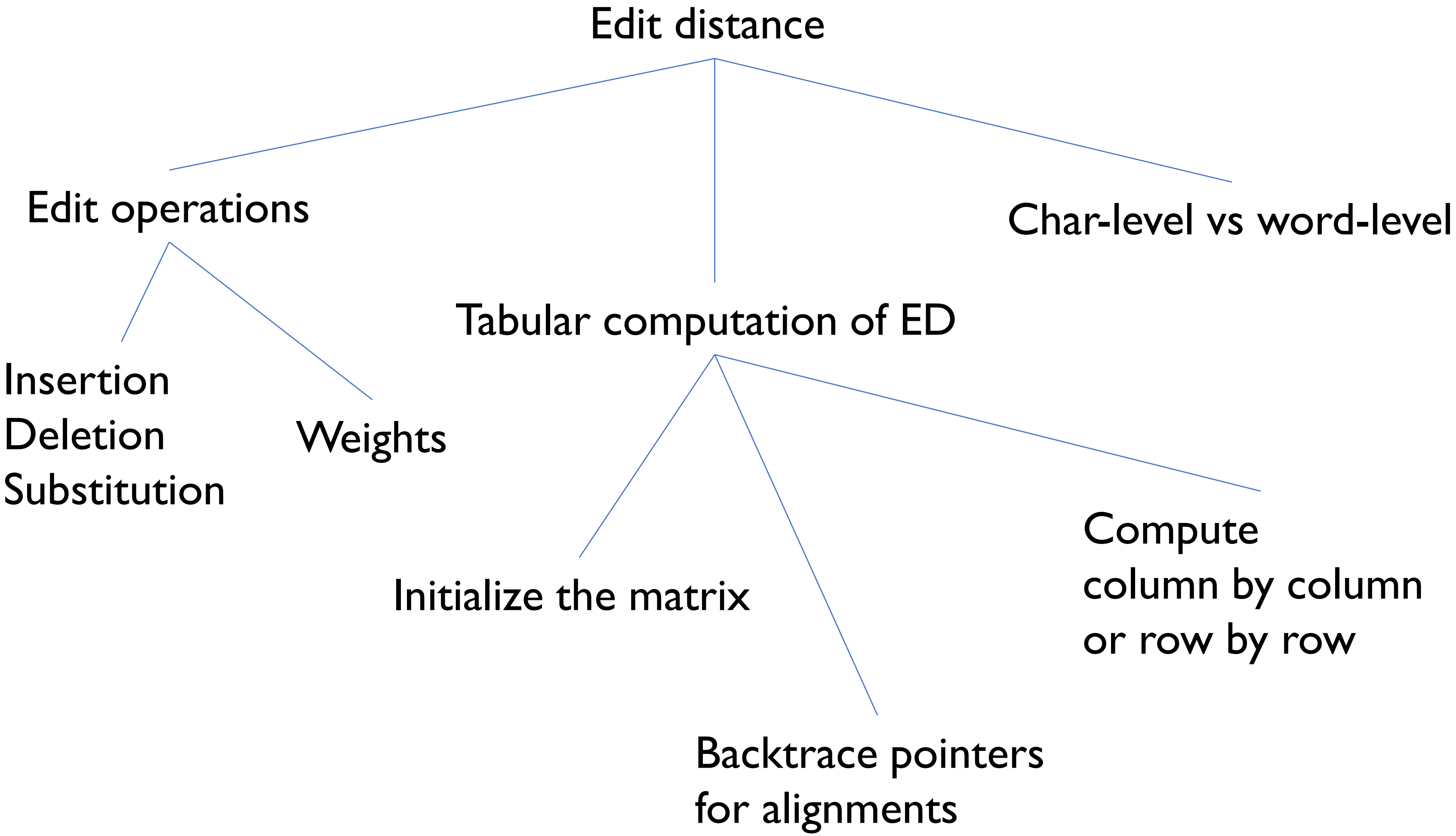
Affix

Lemmatization

Stemming

Porter's stemmer

Summary: Edit distance



Edit distance

Edit operations

Insertion
Deletion
Substitution

Weights

Tabular computation of ED

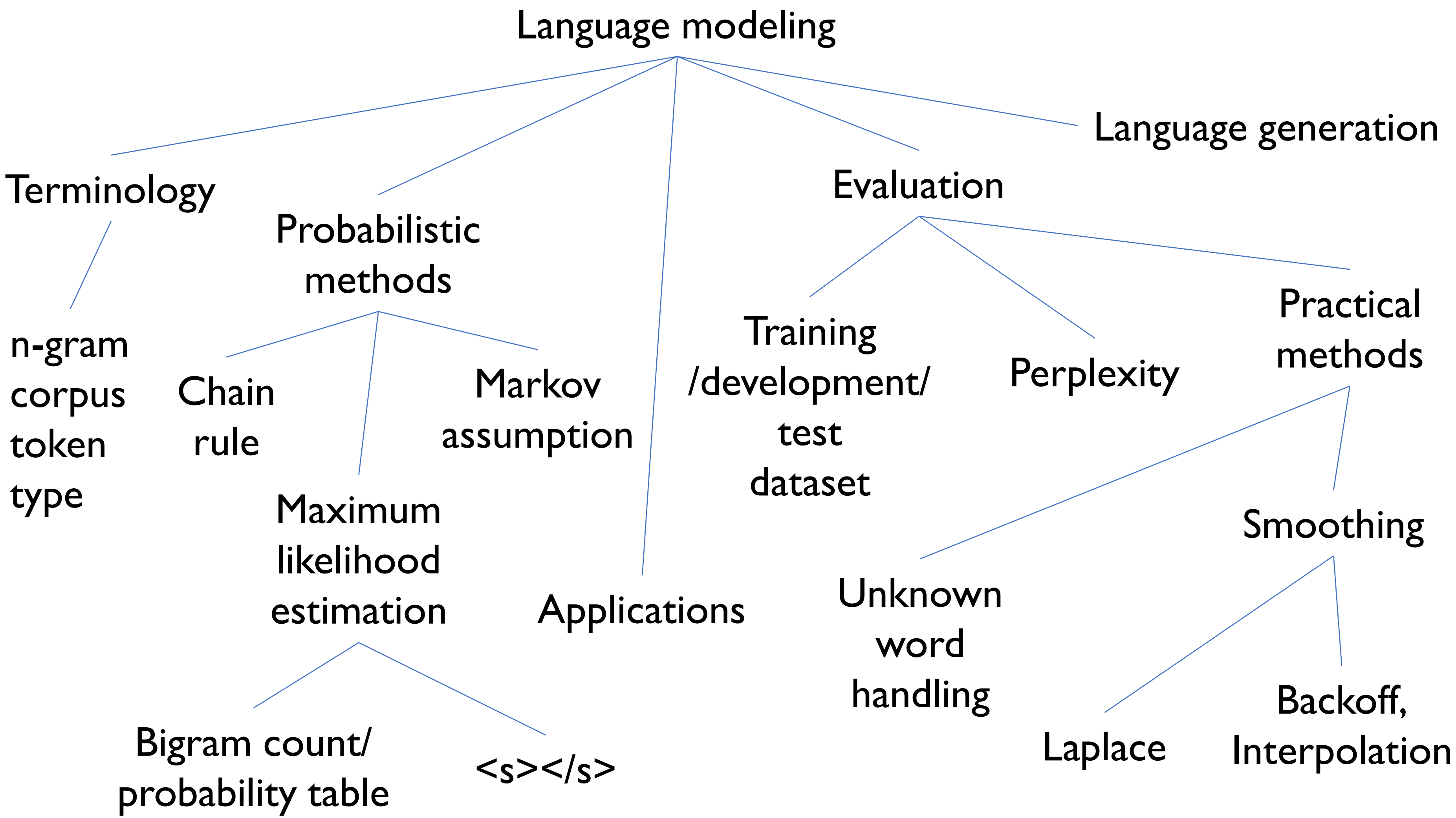
Initialize the matrix

Backtrace pointers
for alignments

Compute
column by column
or row by row

Char-level vs word-level

Summary: N-gram Language Models



Language modeling

Terminology

n-gram
corpus
token
type

Probabilistic methods

Chain rule

Markov assumption

Maximum likelihood estimation

Bigram count/ probability table

<s></s>

Applications

Evaluation

Training /development/ test dataset

Perplexity

Practical methods

Unknown word handling

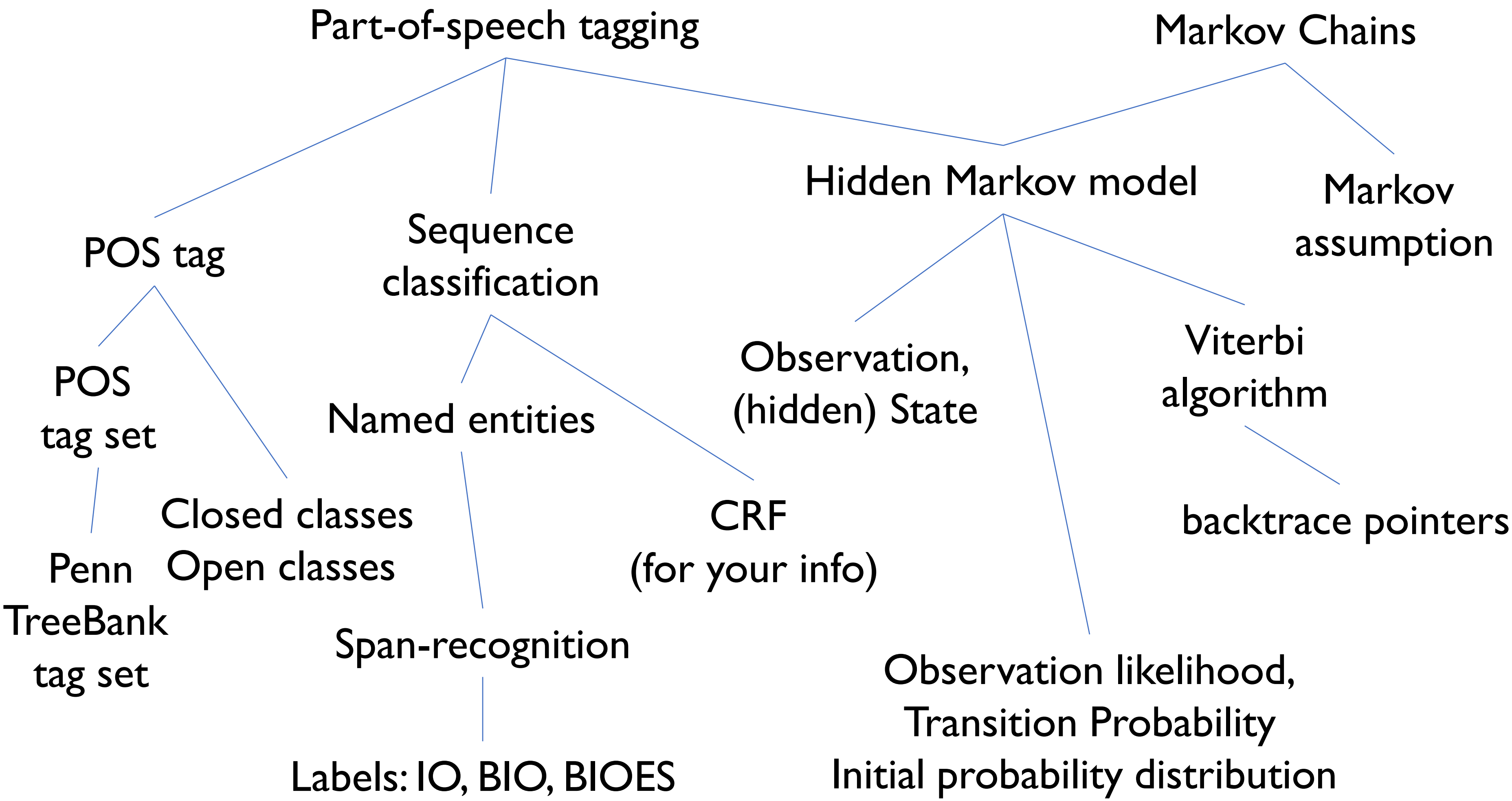
Smoothing

Laplace

Backoff, Interpolation

Language generation

Summary: Part-of-speech and Named entities



Part-of-speech tagging

Markov Chains

Hidden Markov model

Markov assumption

POS tag

Sequence classification

Observation, (hidden) State

Viterbi algorithm

POS tag set

Named entities

backtrace pointers

Closed classes
Open classes

CRF (for your info)

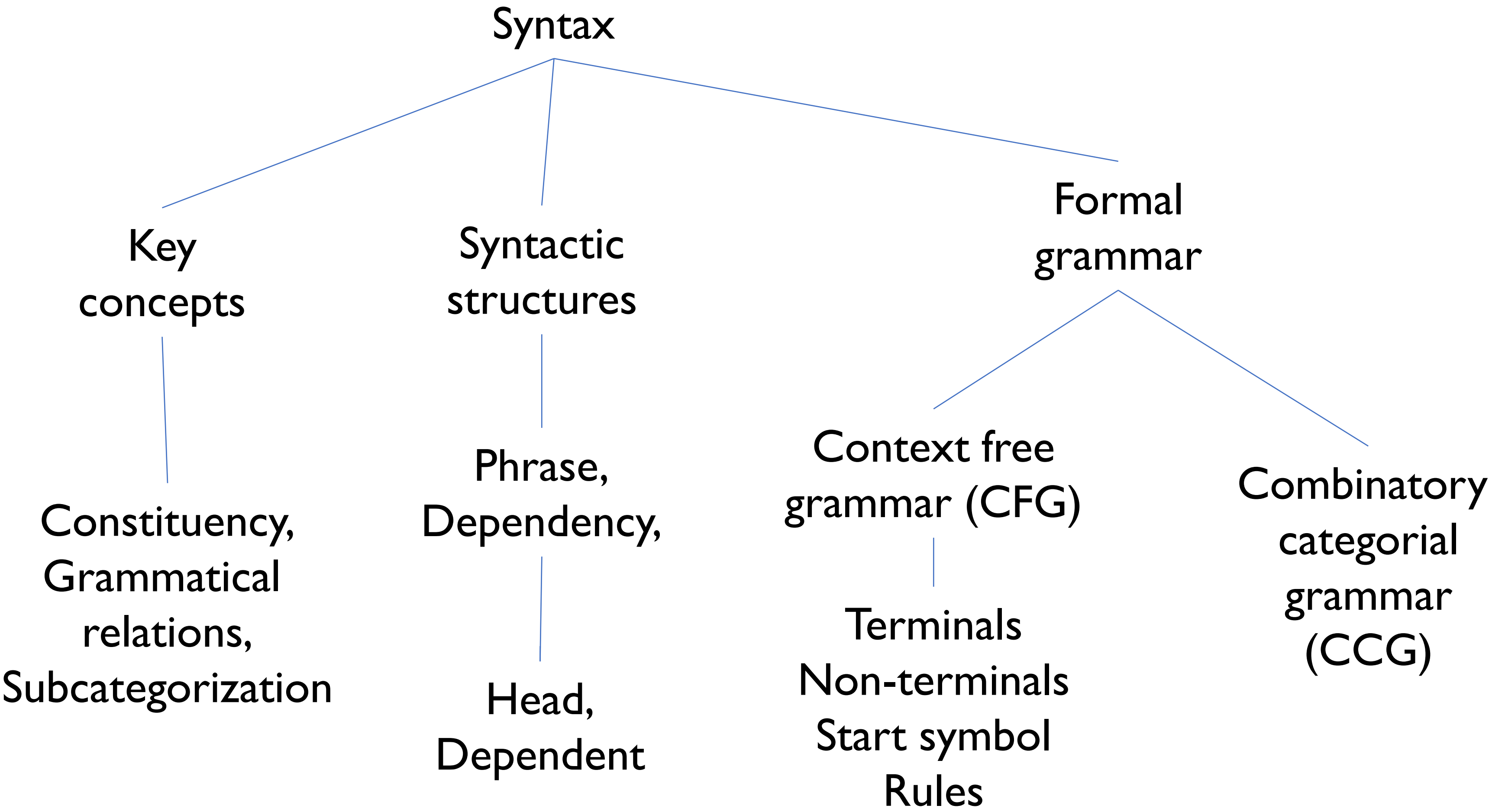
Penn TreeBank tag set

Span-recognition

Observation likelihood,
Transition Probability
Initial probability distribution

Labels: IO, BIO, BIOES

Summary: Constituency Grammars and Parsing



Syntax

Key concepts

Constituency, Grammatical relations, Subcategorization

Syntactic structures

Phrase, Dependency,

Head, Dependent

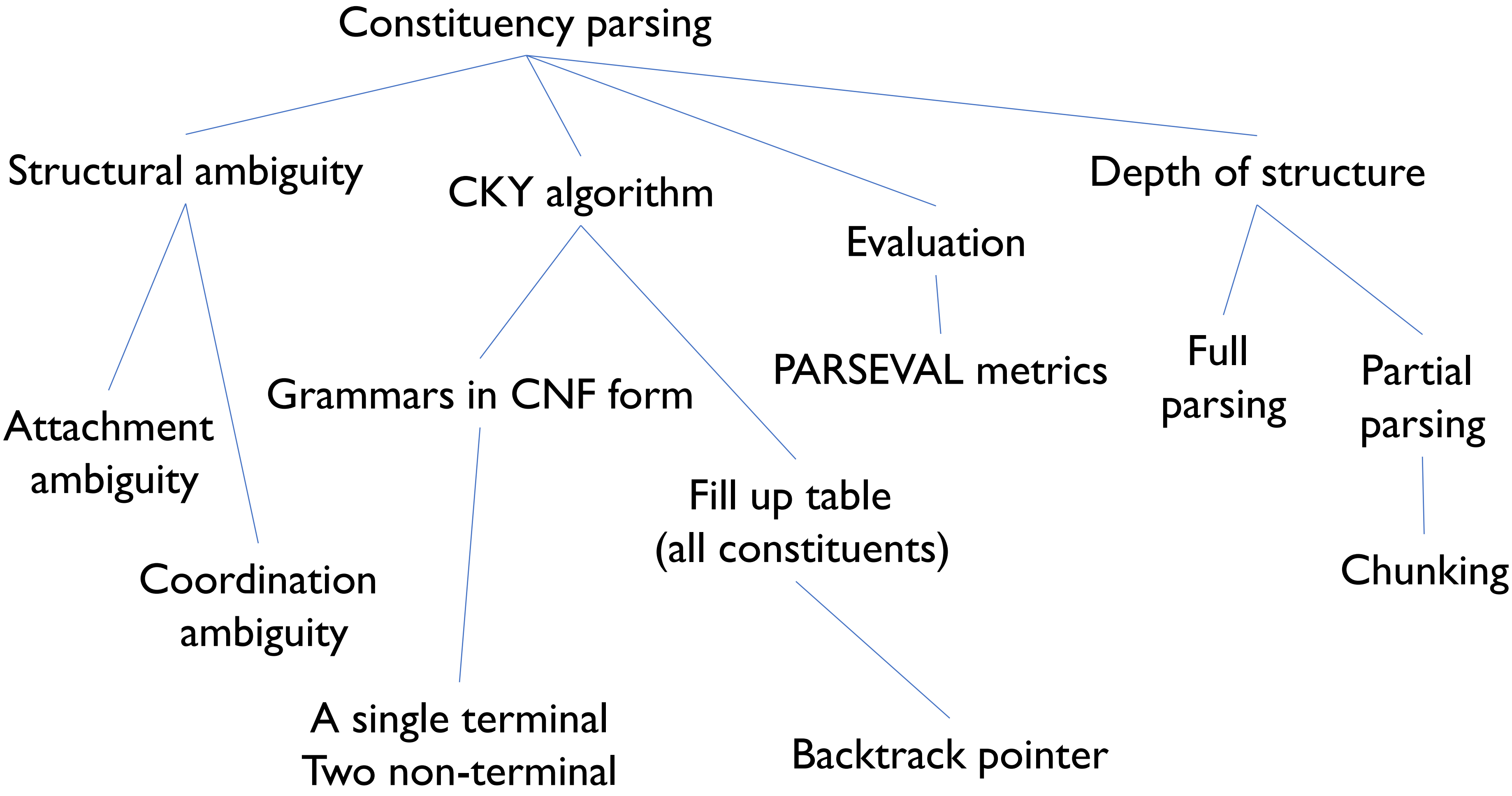
Formal grammar

Context free grammar (CFG)

Terminals
Non-terminals
Start symbol
Rules

Combinatory categorial grammar (CCG)

Summary: Constituency Grammars and Parsing



Constituency parsing

Structural ambiguity

Attachment ambiguity

Coordination ambiguity

CKY algorithm

Grammars in CNF form

A single terminal
Two non-terminal

Fill up table
(all constituents)

Backtrack pointer

Evaluation

PARSEVAL metrics

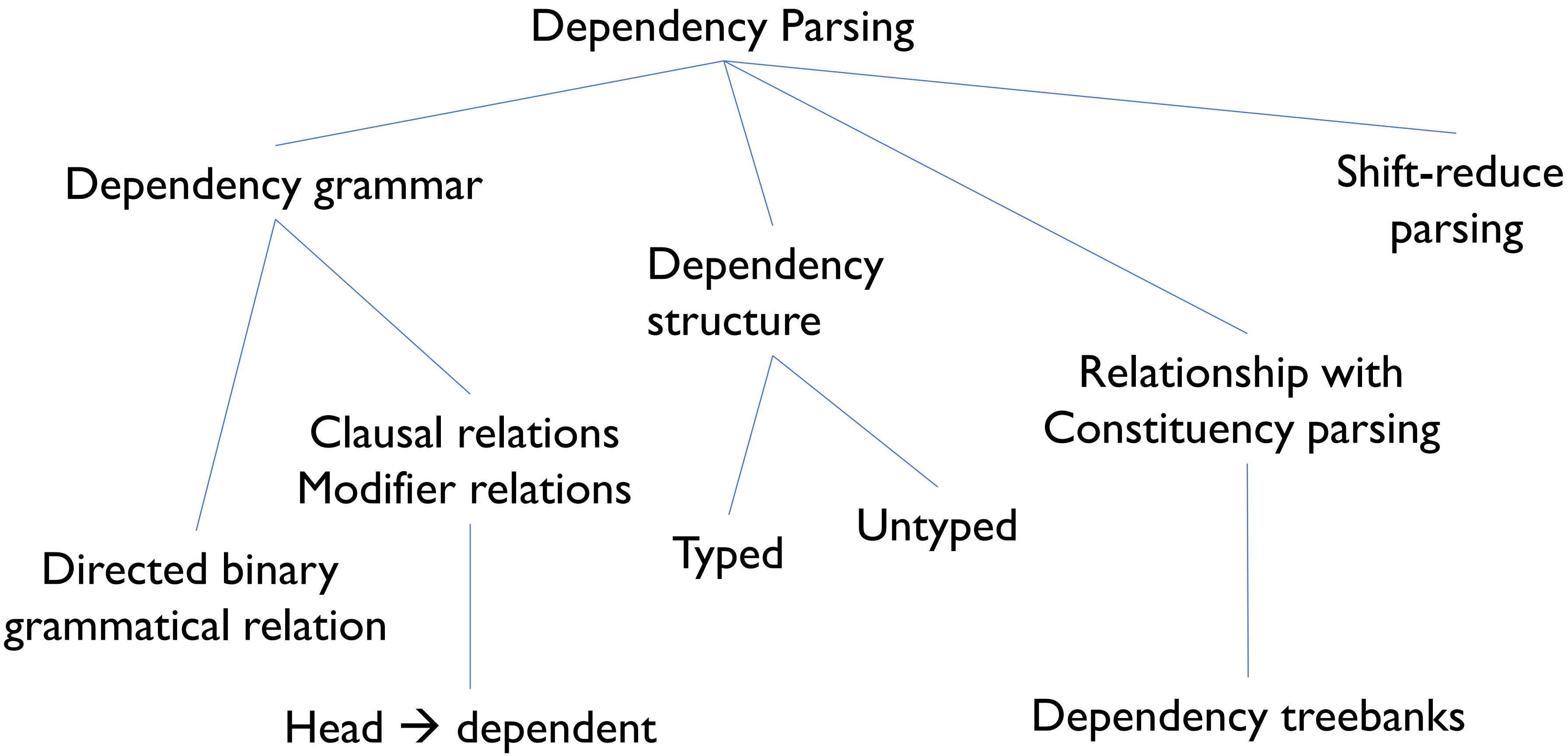
Depth of structure

Full parsing

Partial parsing

Chunking

Summary: Dependency Parsing



Dependency Parsing

Dependency grammar

Directed binary grammatical relation

Clausal relations
Modifier relations

Head → dependent

Dependency structure

Typed

Untyped

Relationship with Constituency parsing

Dependency treebanks

Shift-reduce parsing

Summary

- RegExr: an online tool to learn, build, & test Regular Expressions
  - http://regexr.com/

- Java RegEx API and Tutorial
  - http://docs.oracle.com/javase/8/docs/api/java/util/regex/package-summary.html
  - http://docs.oracle.com/javase/tutorial/essential/regex/

- Reference: https://web.stanford.edu/~jurafsky/slp3/
  - **Chapter 2**, Regular Expressions, Text Normalization, Edit Distance

What can we do?

- Given a document, we are able to search for the matching strings with a query specified in Regular Expression
  - The given document is basically a sequence of characters
  - At this stage, we do not understand words or sentences in the document.

- Next, it would be useful to recognize the words, sentences in the document
  - With the words and sentences, we will be able to understand the structure or meaning of the sentences.

Summary

- Words and Corpora
  - Datasheet specifies properties of a dataset
  - Words: lemma, word forms
- Tokenization and normalization
  - Issues with tokenization
  - Case folding, lemmatization, stemming
  - Sentence segmentation
- Edit distance
  - Applications
  - Algorithm

- Reading: Chapter 2 https://web.stanford.edu/~jurafsky/slp3/

What can we do?

- Given a document we are able to segment its words and sentences.
  - The idea of word segmentation and sentence segmentation is similar, except the unit of processing is different, i.e., word vs sentence.
  - Depends on the characteristics of the document, we may need to select the most appropriate tokenizers.

- Given a word, we are able to perform normalization, to get the lemma or stem.

- Given two words, we are able to measure the similarity or distance between them, by Edit Distance
  - The same idea can be applied to measure two sentences, except the unit of processing is different, i.e., character vs word

N-gram Language Model

- Word prediction
  - Probability of a sequence of words $P(w_1 w_2 \ldots . w_n)$, or probability of a word given some history $P(w|h)$
- N-grams
  - Counting and basic concepts
- N-gram Language Model
  - Modeling unknown words
  - Smoothing to avoid assigning zero probabilities to unseen sequences
  - Evaluation
- Reference: https://web.stanford.edu/~jurafsky/slp3/
  - **Chapter 3**, N-gram Language Models

What can we do?

- Given a collection of documents, we are able to train a language model
- Given a language model, we are able to compute the probability of sentences
- Given a language model, we can also generate sentences

Summary

- POS tag: word types
  - POS tagging with HMM
  - The Viterbi algorithm
  - Conditional Random Fields
- Named entity
  - NER as a sequence labelling task
- Reference:
  - Chapter 8 https://web.stanford.edu/~jurafsky/slp3/

What can we do?

- Given a sentence, we can select POS taggers to tag the words in the sentence with their correct word categories
  - This would immediately enable us to select the words in certain categories
  - We can also combine with RegEx to find word sequences by patterns
  - For example, a noun phrase may have this pattern: an optional determiner, zero, one or more adjectives, then a noun.

- Given a sentence, we can also find the named entities from the sentence with a NER model.
  - This offers many more ways to understand the document, like linking the entities to Wikipedia to understand the background information for each entities

- We may also formulate other related problems to a sequence labelling task, by using the BIO tagging scheme.

Summary

- Structural ambiguity
- Parsing with CKY algorithm
- Evaluating parsers
- Partial or Shallow Parsing
- References
  - Chapter 13 https://web.stanford.edu/~jurafsky/slp3/

What can we do?

- Given a sentence, we can have its parse tree with the help from a parser

- We are able to traverse the parse tree to obtain various subtrees, corresponding to different segments of the sentence

- We can also compare the structural similarity between two sentences based on their parse trees.

Summary

- Dependency: Head-dependent

- Dependency formalism

- Dependency parsing

- Reference
  - Chapter 14 https://web.stanford.edu/~jurafsky/slp3/