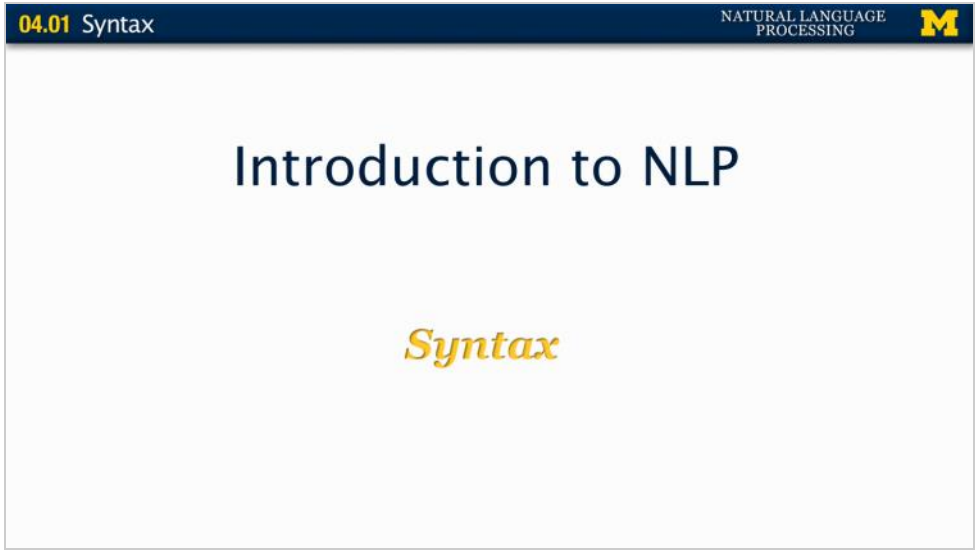
Week 4

2016年7月19日 星期二　　下午11:19

4.1



Syntax

- Is language more than just a "bag of words"?
- Grammatical rules apply to categories and groups of words, not individual words.
- Example – a sentence includes a subject and a predicate. The subject is a noun phrase and the predicate is a verb phrase.
  - Noun phrase: The cat, Samantha, She
  - Verb phrase: arrived, went away, had dinner
- When people learn a new word, they learn its syntactic usage.
  - Examples: wug (n), cluvious (adj) – use them in sentences
  - Hard to come up with made up words: forkle, vleer, etc. all taken.

Defining Parts of Speech

- What do nouns typically have in common?
  - E.g., *can* be preceded by "the".
- Verbs can be preceded by "can't".
- Adjectives can come between "the" and a noun.
- How is this different from grade school definitions?
- Determiners: a, the, many, no, five
- Prepositions: for, to, in, without, before

Noun
property:
can be preceded by "the"

verb.

Adj.

From a syntatic point of view we are interested in how the words can form sentences.

The Lexicon

- How do we think of words like cat, run, five?
  - pronunciation, part of speech, meaning
- Five: /faɪv/, numeral, "5"
- Ambiguity

Constituents

- Constituents are continuous
- Constituents are non-crossing
  - if two constituents share one word, then one of them must completely contain the other.
- Each word is a constituent

Constituent.

noun:

she. Samantha.

Constituent Tests

- "coordination" test
- "pronoun" test
  - A small dog is barking in the park.
  - It is barking in the park
- "question by repetition" test:
  - I have seen blue elephants
  - Blue elephants? ✓
  - * Seen blue? ✗
  - Seen blue elephants? ✓
- "topicalization" test:
  - Blue elephants, I have seen.
- "question" test:
  - *What* have I seen?
- "deletion" test
  - Last year I saw a blue elephant in the zoo.
- "semantic" test
- "intuitition" test

valid constituent

two constituent with a conjunction

⇒ the two constituents are of the same type.

Many test to determine if several words form a constituent.

How To Generate Sentences

- One way: tree structure
  - Generate the tree structure first
  - Then fill the leaf nodes with terminals

A Simple Syntactic Rule

- The simplest rule for a sentence, e.g. "Birds fly"

S → N V

noun + verb.

Birds fly.

Simplest Grammar

S → N V

N → Samantha | Min | Jorge

V → left | sang | walked

Sample sentences:

Samantha sang

Jorge left

Syntax

- The verbs so far were intransitive (no direct object)
- What rules are needed next?
  - Transitive verbs and direct objects ("Jorge saw Samantha")
  - Determiners ("the cats")
- Combinatorial explosion (even for the simplest form of sentences)
- Need for noun phrases
- Ditto for verb phrases

Latest Grammar

```
S → NP VP
NP → DT N
VP → V NP
DT → the | a
N → child | cat | dog
V → took | saw | liked | scared | chased
```

Determinant.

Sample sentences:

a dog chased the cat

the child saw a dog

Alternatives

- Different expansions of a category are delineated with "|"
  - NP → PN | DT CN
- One rule for proper nouns and another for common nouns

PN: proper noun Samantha

CN: common noun

Latest Grammar

```
S → NP VP
NP → DT CN
NP → PN
VP → V NP
DT → the | a
CN → child | cat | dog
PN → Samantha | Jorge | Min
V → took | saw | liked | scared | chased
```

Sample sentences:

a child scared Jorge

Min took the child

Optional Categories

- Wherever N is allowed in a sentence,
  - DT N
  - JJ N
  - DT JJ N

  are also allowed
- We can use the notation for alternatives
  - NP → N | DT N | JJ N | DT JJ N
- Optional categories can be also marked using parentheses:
  - NP → (DT) (JJ) N

DT JJ N
determinant | adjective | noun

JJ: Adjective

cats | the cats | small cats | the small cats

Verb Phrases

- Samantha ran.
- Samantha ran to the park.
- Samantha ran away.
- Samantha bought a cookie.
- Samantha bought a cookie for John.
- Overall structure: VP → V (NP) (P) (NP)

V NP P NP
prep.

Latest Grammar

```
S → NP VP
NP → DT CN
NP → PN
VP → V (NP) (P) (NP)
DT → the | a
CN → child | cat | dog
PN → Samantha | Jorge | Min
P → to | for | from | in
V → took | saw | liked | scared | chased | gave
```

Sample sentences:

Samantha saw the cat
Jorge gave the cat to Min

Prepositional Phrases

- Examples:
  - Mary bought a book for John **in a bookstore**.
  - The bookstore sells magazines.
  - The bookstore **on Main St.** sells magazines.
  - Mary ran away.
  - Mary ran **down the hill**.
- Changes are needed to both NP and VP to accommodate prepositional phrases
  - Wherever a preposition is allowed, it can be followed by a noun phrase.
  - Run up
  - NP can contain any number of PPs but only up to two NPs.
- How do we revise the grammar accordingly?

PP : prepositional phrase.

The Rules So Far

- **S → NP VP**
- **NP → (DT) (JJ) N (PP)**
- **VP → V (NP) (PP)**
- **PP → P (NP)**

Can apply Rule 2-4 to a long sequence
=> producing arbitrarily long sentences
=> But this is just theoretical possibility
We don't see such long sentences in reality

PP Ambiguity

- The boy saw the woman with the telescope. Ambiguity

```
PP → PREP NP
VP → V NP PP
VP → V NP
NP → DT N
NP → DT N PP
```

Repetition (*)

- (JJ*) = a sequence of zero or more JJ — Adjectives
- Are all sequences of adjectives allowed?
  - a big red house
  - * a red big house
- Adjective ordering in English depends on semantics!

Adjective ordering depends on semantics.

Exercise

- The Little Red Riding Hood
- Three Little Pigs
- The Three Musketeers
- The Steadfast Tin Soldier
- The French Connection
- Old Macdonald
- Five Golden Rings
- The Ancient Mariner

little. red.
size before color

nationality.

Adjective Ordering

- **Det**
- Number
- Strength
- Size
- Age
- Shape
- Color
- Origin
- Material
- Purpose
- **Noun**

- det < number < size < color < purpose < noun
- strength < material < noun
- origin < noun

Nested Sentences

- Examples:
  - I don't recall whether I took the dog out.
  - Do you know if the mall is still open?
- **VP → V (NP) (NP) (C S) (PP*)**
- Can (C S) appear inside an NP?
  - Whether he will win the elections remains to be seen.

Recursion

- S can generate VP, VP can generate S
- NP can generate PP, PP can generate NP
- What does recursion allow?
- Is there a longest sentence in English?
- Conjunction of NPs:
  NP → NP and NP
- Conjunction of PPs:
  PP → PP and PP
- Conjunction of VPs:
  VP → VP and VP

Meta-patterns

- S → NP VP
  - NP → (DT) (JJ) N (PP)
  - VP → V (NP) (PP)
  - PP → P (NP)
- Is there a meta-pattern here?
  - XP → (specifier) X'
  - X' → X (complement)
- Example: NP → DT N'
- X-bar Theory
  - http://www.unlweb.net/wiki/X-bar_theory

Meta-rules for Conjunctions

- Conjunction
  - X → X and X
- This kind of rule even covers entire sentences
  - S → S and S

Conjunctions.

Auxiliaries

- Is "Aux V" a constituent?
  - I have seen blue elephants and will remember them forever.
- Recursion:
  - VP -> Aux VP
  - Raj may have been sleeping.
- Is such recursion unlimited?

Aux VP
have seen blue elephant

Auxiliary.

Exercise

- Grammar:
  - S → NP VP | CP VP
  - NP → (DT) (JJ*) N (CP) (PP*)
  - VP → V (NP) (NP) (PP*) | V (NP) (CP) (PP*)
  - PP → P NP
  - CP → C S
- What rules are needed to generate these three sentences:
  - 1. The small dog of the neighbors brought me an old tennis ball.
  - 2. That wugs have three eyes is unproven by scientists.
  - 3. I saw the gift that the old man gave me at the meeting.

Ex:

① S → NP VP

NP → DT JJ N PP

VP → V NP NP.

4.2

Parsing ☆ Important part of Natural Language Processing

Introduction to NLP

Introduction to Parsing

Parsing Programming Languages

```
#include <stdio.h>

int main()
{
  int n, reverse = 0;

  printf("Enter a number to reverse\n");
  scanf("%d",&n);

  while (n != 0)
  {
    reverse = reverse * 10;
    reverse = reverse + n%10;
    n = n/10;
  }
  printf("Reverse of entered number is = %d\n", reverse);

  return 0;
}
```

Parsing Programming Language is easy

Parsing Human Languages

- Rather different than computer languages
  - Can you think in which ways?

Parsing Human Languages

- Rather different than computer languages
  - No types for words
  - No brackets around phrases
  - Ambiguity
    - Words
    - Parses
  - Implied information

The Parsing Problem

- Parsing means associating tree structures to a sentence, given a grammar (often a CFG)
  - There may be exactly one such tree structure
  - There may be many such structures
  - There may be none
- Grammars (e.g., CFG) are declarative
  - They don't specify how the parse tree will be constructed

What is

Context free grammar

Can be multiple or no structures

Syntactic Ambiguities

- PP attachment
  - I saw the man with the telescope
- Gaps
  - Mary likes Physics but hates Chemistry
- Coordination scope
  - Small boys and girls are playing
- Particles vs. prepositions
  - She ran up a large bill
- Gerund vs. adjective
  - Frightening kids can cause trouble

two interpretations

not explict in the structure that Mary is the subject

Two possible interpretations

She ran up a large hill.

prep.

adj or gerund

Types of ambiguity

- PP attachment
- gaps
- Coordination scope
- Particles vs preposition
- Gerund vs adj.

Application of Parsing

Applications Of Parsing

- Grammar checking
  - I want to return this shoes.
- Question answering
  - How many people in sales make $40K or more per year?
- Machine translation
  - E.g., word order - SVO vs. SOV
- Information extraction
  - Breaking Bad takes place in New Mexico.
- Speech generation
- Speech understanding
- Interpretation

Different language have diff constituent order.

name of TV show, name of state

NLP

Introduction to NLP

Context-free grammars

Context-free Grammars

- A context-free grammar is a 4-tuple (N,Σ,R,S)
  - N: non-terminal symbols
  - Σ: terminal symbols (disjoint from N)
  - R: rules (A → β), where β is a string from (Σ ∪ N)*
  - S: start symbol from N

★ Context-free grammar.
4-tuple. (N, Σ, R, S)

Example

```
["the", "child", "ate", "the", "cake", "with", "the", "fork"]

      S -> NP VP
      NP -> DT N | NP PP
      PP -> PRP NP
      VP -> V NP | VP PP
      DT -> 'a' | 'the'
      N -> 'child' | 'cake' | 'fork'
      PRP -> 'with' | 'to'
      V -> 'saw' | 'ate'
```

options.

Example

```
["the", "child", "ate", "the", "cake", "with", "the", "fork"]

      S -> NP VP
      NP -> DT N | NP PP
      PP -> PRP NP
      VP -> V NP | VP PP
      DT -> 'a' | 'the'
      N -> 'child' | 'cake' | 'fork'
      PRP -> 'with' | 'to'
      V -> 'saw' | 'ate'
```

Heads marked in bold face

Bolded: headed constituent: one of their component is more important than others
For instance: N for NP, PRP for PP, V for VP. We'll come back to this later

Phrase-structure Grammars (1/2)

- Sentences are not just bags of words
  - Alice bought Bob flowers
  - Bob bought Alice flowers
- Context-free view of language
  - A prepositional phrase looks the same whether it is part of the subject NP or part of the VP
- Constituent order
  - SVO (subject verb object)
  - SOV (subject object verb)

parsing tell us who is the receiver.

Phrase-structure Grammars (2/2)

- Auxiliary verbs
  - The dog may have eaten my homework
- Imperative sentences
  - Leave the book on the table
- Interrogative sentences
  - Did the customer have a complaint?
- Negative sentences
  - The customer didn't have a complaint

aux aux

祈使句结构并不完整.

Yes/No question.

缩写.

A Longer Example

have the children arrive home

```
S -> NP VP | Aux NP VP | VP
NP -> PRON | Det Nom
Nom -> N | Nom N | Nom PP
PP -> PRP NP
VP -> V | V NP | VP PP
Det -> 'the' | 'a' | 'this'
PRON -> 'he' | 'she'
N -> 'book' | 'boys' | 'girl'
PRP -> 'with' | 'in'
V -> 'takes' | 'take'
```

use noun to modify noun [H: banana tree?]

only V. intransitive word

What changes were made to the grammar?

A Longer Example

```
S -> NP VP | Aux NP VP | VP
NP -> PRON | Det Nom
Nom -> N | Nom N | Nom PP
PP -> PRP NP
VP -> V | V NP | VP PP
Det -> 'the' | 'a' | 'this'
PRON -> 'he' | 'she'
N -> 'book' | 'boys' | 'girl'
PRP -> 'with' | 'in'
V -> 'takes' | 'take'
```


A Longer Example

```
S -> NP VP | Aux NP VP | VP
NP -> PRON | Det Nom
Nom -> N | Nom N | Nom PP
PP -> PRP NP
VP -> V | V NP | VP PP
Det -> 'the' | 'a' | 'this'
PRON -> 'he' | 'she'
N -> 'book' | 'boys' | 'girl'
PRP -> 'with' | 'in'
V -> 'takes' | 'take'
```


Penn Treebank Example

Two sentences from WSJ.

Real sentence

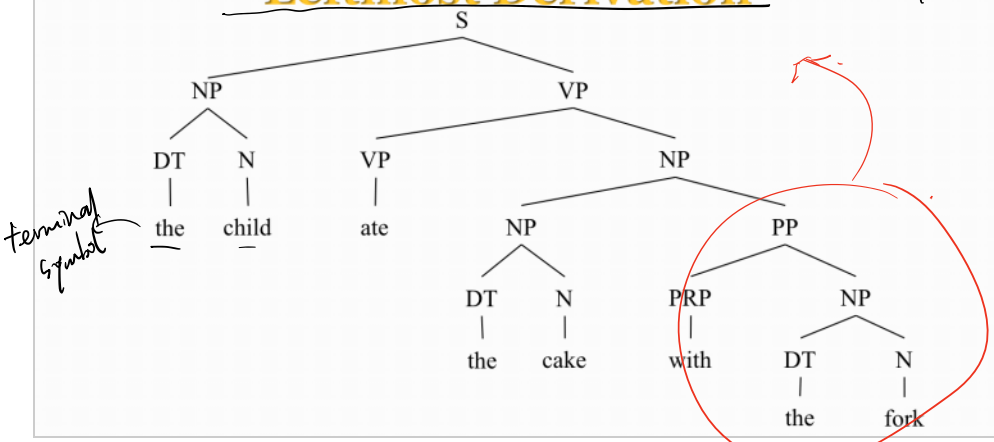
```
( (S
    (NP-SBJ
      (NP (NNP Pierre) (NNP Vinken) )
      (, ,)
      (ADJP
        (NP (CD 61) (NNS years) )
        (JJ old) )
      (, ,) )
    (VP (MD will)
      (VP (VB join)
        (NP (DT the) (NN board) )
        (PP-CLR (IN as)
          (NP (DT a) (JJ nonexecutive) (NN director) ))
        (NP-TMP (NNP Nov.) (CD 29) )))
    (. .) ))
( (S
    (NP-SBJ (NNP Mr.) (NNP Vinken) )
    (VP (VBZ is)
      (NP-PRD
        (NP (NN chairman) )
        (PP (IN of)
          (NP
            (NP (NNP Elsevier) (NNP N.V.) )
            (, ,)
            (NP (DT the) (NNP Dutch) (VBG publishing) (NN group) )))))
    (. .) ))
```

Leftmost Derivation

- A leftmost derivation is a sequence of strings s_1, s_2, ..., s_n
  - s_1 = S, the start symbol
  - s_n includes only terminal symbols
- Example:
  - [S]
  - [S] [NP VP]
  - [S] [NP VP] [DT N VP]
  - ...
  - [S] [NP VP] [DT N VP] ... [the child ate the cake with the fork]

Given a grammar.
Always expanding the left-most
← left most.
→ everything is an terminal symbol.

Leftmost Derivation



terminal symbol

This is not necessarily the correct Semantic — Just come up following the leftmost derivation

NLP

4.3

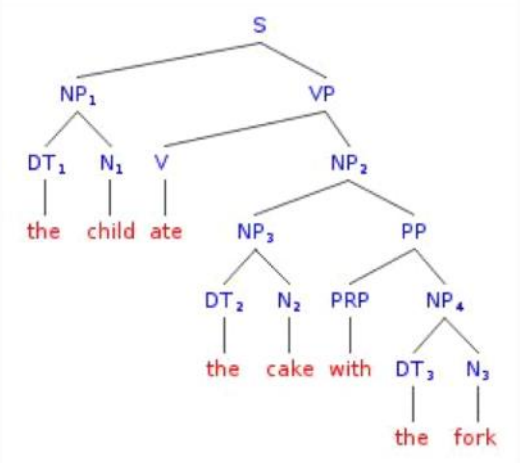
NLP

04.03 Classic Parsing Methods NATURAL LANGUAGE PROCESSING

Introduction to NLP

Classic parsing methods

04.03 Classic Parsing Methods NATURAL LANGUAGE PROCESSING




```
S -> NP VP
NP -> DT N | NP PP
PP -> PRP NP
VP -> V NP | VP PP
DT -> 'a' | 'the'
N -> 'child' | 'cake' | 'fork'
PRP -> 'with' | 'to'
V -> 'saw' | 'ate'
```

Parsing as Search

- There are two types of constraints on the parses
  - From the input sentence
  - From the grammar
- Therefore, two general approaches to parsing
  - Top-down
  - Bottom-up

Top Down Parsing

S

```
S -> NP VP
NP -> DT N | NP PP
PP -> PRP NP
VP -> V NP | VP PP
DT -> 'a' | 'the'
N -> 'child' | 'cake' | 'fork'
PRP -> 'with' | 'to'
V -> 'saw' | 'ate'
```


Top Down Parsing


```
S -> NP VP
NP -> DT N | NP PP
PP -> PRP NP
VP -> V NP | VP PP
DT -> 'a' | 'the'
N -> 'child' | 'cake' | 'fork'
PRP -> 'with' | 'to'
V -> 'saw' | 'ate'
```

First thing to try
S → NP VP.
Try NP → NP PP @ WRONG.
NP → DT N ✓

Top Down Parsing


```
S -> NP VP
NP -> DT N | NP PP
PP -> PRP NP
VP -> V NP | VP PP
DT -> 'a' | 'the'
N -> 'child' | 'cake' | 'fork'
PRP -> 'with' | 'to'
V -> 'saw' | 'ate'
```

04.03 Classic Parsing Methods
NATURAL LANGUAGE PROCESSING

Top Down Parsing


```
S -> NP VP
NP -> DT N | NP PP
PP -> PRP NP
VP -> V NP | VP PP
DT -> 'a' | 'the'
N -> 'child' | 'cake' | 'fork'
PRP -> 'with' | 'to'
V -> 'saw' | 'ate'
```

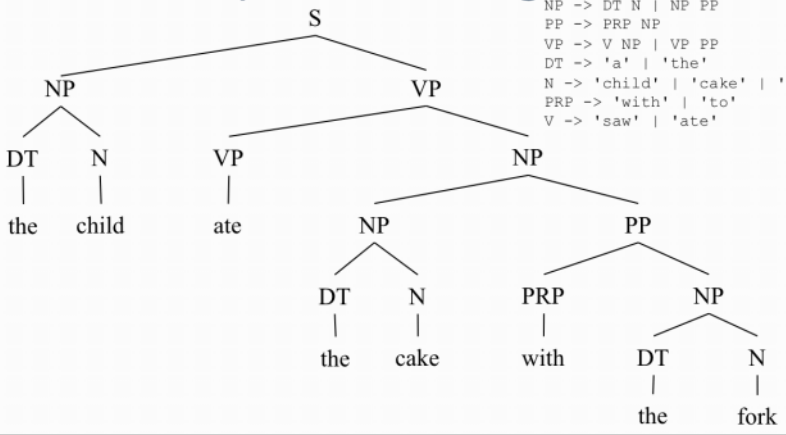
04.03 Classic Parsing Methods
NATURAL LANGUAGE PROCESSING

Top Down Parsing


```
S -> NP VP
NP -> DT N | NP PP
PP -> PRP NP
VP -> V NP | VP PP
DT -> 'a' | 'the'
N -> 'child' | 'cake' | 'fork'
PRP -> 'with' | 'to'
V -> 'saw' | 'ate'
```

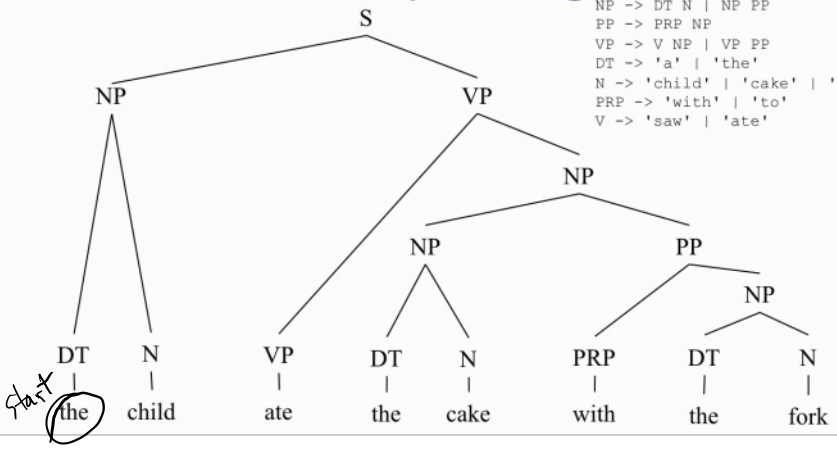
Top Down Parsing

```
S -> NP VP
NP -> DT N | NP PP
PP -> PRP NP
VP -> V NP | VP PP
DT -> 'a' | 'the'
N -> 'child' | 'cake' | 'fork'
PRP -> 'with' | 'to'
V -> 'saw' | 'ate'
```



Bottom Up Parsing

```
S -> NP VP
NP -> DT N | NP PP
PP -> PRP NP
VP -> V NP | VP PP
DT -> 'a' | 'the'
N -> 'child' | 'cake' | 'fork'
PRP -> 'with' | 'to'
V -> 'saw' | 'ate'
```



get parts of speech for all the words.

binarized form

right hand side always have two non-terminal.

Bottom Up Vs. Top Down Methods

- Bottom up
  - explores options that won't lead to a full parse.
- Top down
  - explores options that don't match the full sentence.
- Dynamic programming
  - caches of intermediate results (memoization)
- Cocke-Kasami-Younger (CKY) parser
  - based on dynamic programming

deadend starting from the top.

→ The technique to achieve the above two.

→ store the info of the part of text already parsed.

Introduction to NLP

Shift-reduce parsing

Shift-reduce Parsing

- A bottom-up parser
  - Tries to match the RHS of a production until it can build an S
- *Shift* operation
  - Each word in the input sentence is pushed onto a stack
- *Reduce* operation
  - If the top n words on the top of the stack match the RHS of a production, then they are popped and replaced by the LHS of the production
- Stopping condition
  - The process stops when the input sentence has been processed and S has been popped from the stack.

example. run-down below.

Shift-reduce Parsing Example

[* the child ate the cake]

Shift-reduce Parsing Example

```
  [ * the child ate the cake]
S [ 'the' * child ate the cake]
```

shift

Shift-reduce Parsing Example

```
  [ * the child ate the cake]
S [ 'the' * child ate the cake]
R [ DT * child ate the cake]
```


Shift-reduce Parsing Example

```
  [ * the child ate the cake]
S [ 'the' * child ate the cake]
R [ DT * child ate the cake]
S [ DT 'child' * ate the cake]
R [ DT N * ate the cake]
```

Shift-reduce Parsing Example

```
  [ * the child ate the cake]
S [ 'the' * child ate the cake]
R [ DT * child ate the cake]
S [ DT 'child' * ate the cake]
R [ DT N * ate the cake]
R [ NP * ate the cake]
S [ NP 'ate' * the cake]
```

04.03 Classic Parsing Methods
NATURAL LANGUAGE PROCESSING

Shift-reduce Parsing Example

```
  [ * the child ate the cake]
S [ 'the' * child ate the cake]
R [ DT * child ate the cake]
S [ DT 'child' * ate the cake]
R [ DT N * ate the cake]
R [ NP * ate the cake]
S [ NP 'ate' * the cake]
R [ NP V * the cake]
S [ NP V 'the' * cake]
R [ NP V DT * cake]
S [ NP V DT 'cake' * ]
```

04.03 Classic Parsing Methods
NATURAL LANGUAGE PROCESSING

Shift-reduce Parsing Example

```
  [ * the child ate the cake]
S [ 'the' * child ate the cake]
R [ DT * child ate the cake]
S [ DT 'child' * ate the cake]
R [ DT N * ate the cake]
R [ NP * ate the cake]
S [ NP 'ate' * the cake]
R [ NP V * the cake]
S [ NP V 'the' * cake]
R [ NP V DT * cake]
S [ NP V DT 'cake' * ]
R [ NP V DT N * ]
R [ NP V NP * ]
R [ NP VP * ]
R [ S * ]
```

Shift-reduce Parsing Example

```
  [ * the child ate the cake]
S [ 'the' * child ate the cake]
R [ DT * child ate the cake]
S [ DT 'child' * ate the cake]
R [ DT N * ate the cake]
R [ NP * ate the cake]
S [ NP 'ate' * the cake]
R [ NP V * the cake]
S [ NP V 'the' * cake]
R [ NP V DT * cake]
S [ NP V DT 'cake' * ]
R [ NP V DT N * ]
R [ NP V NP * ]
R [ NP VP * ]
R [ S * ]
```

← end, successful parsing

```
(S (NP (DT the) (N child)) (VP (V ate) (NP (DT the) (N cake))))
```

04.03 Classic Parsing Methods NATURAL LANGUAGE PROCESSING

NLP

04.03 Classic Parsing Methods NATURAL LANGUAGE PROCESSING

Introduction to NLP

Cocke-Kasami-Younger (CKY) Parsing

CKY
Based on dynamic programming.

Dynamic Programming

- Motivation
  - A lot of the work is repeated
  - Caching intermediate results improves the complexity
- Dynamic programming
  - Building a parse for a substring [i,j] based on all parses [i,k] and [k, j] that are included in it.
- Complexity
  - $O(n^3)$ for recognizing an input string of length n

Cubic

to revisit

Dynamic Programming

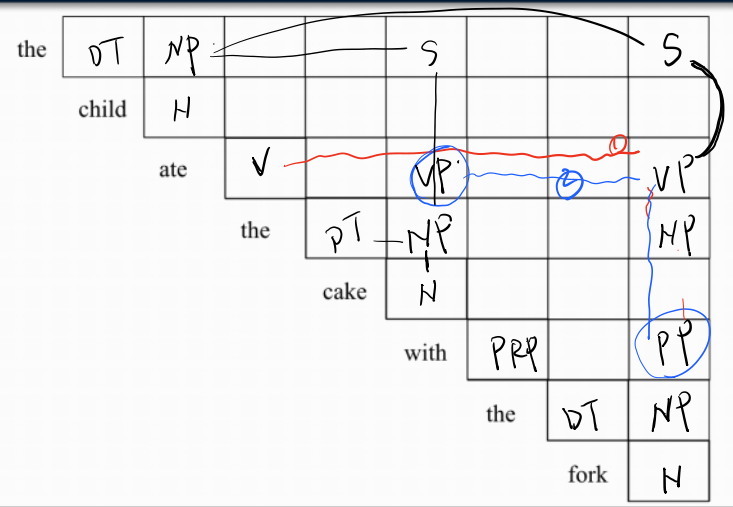
require normalized grammar.

- CKY (Cocke-Kasami-Younger)
  - bottom-up
  - requires a normalized (binarized) grammar
- Earley parser
  - top-down
  - more complicated

Example

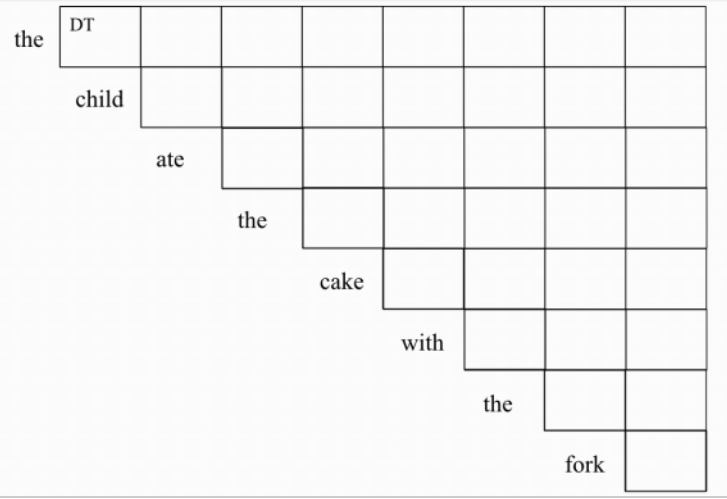
```
["the", "child", "ate", "the", "cake", "with", "the", "fork"]
```

```
S -> NP VP
NP -> DT N | NP PP
PP -> PRP NP
VP -> V NP | VP PP
DT -> 'a' | 'the'
N -> 'child' | 'cake' | 'fork'
PRP -> 'with' | 'to'
V -> 'saw' | 'ate'
```

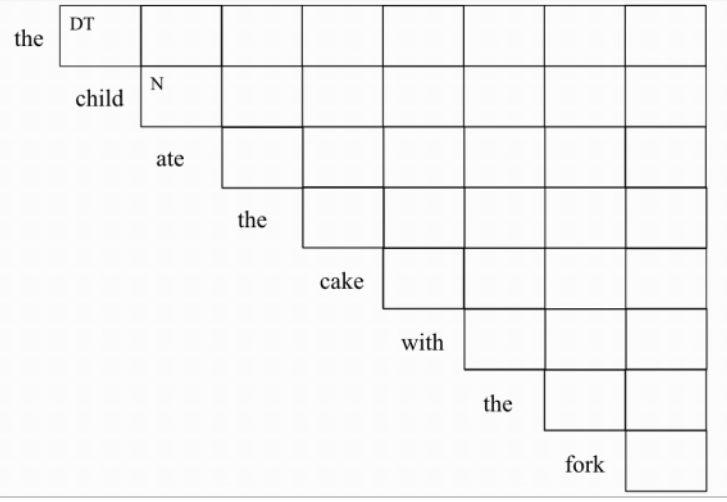


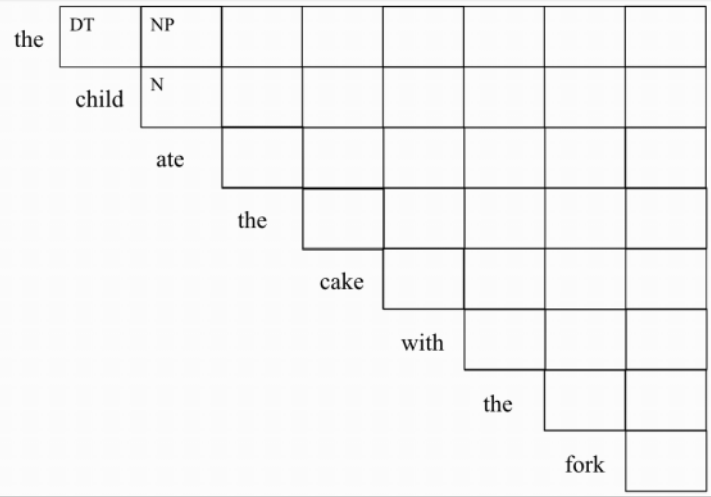
ate the cake with the fork
VP PP
VP
ate the cake with the fork
V NP
VP

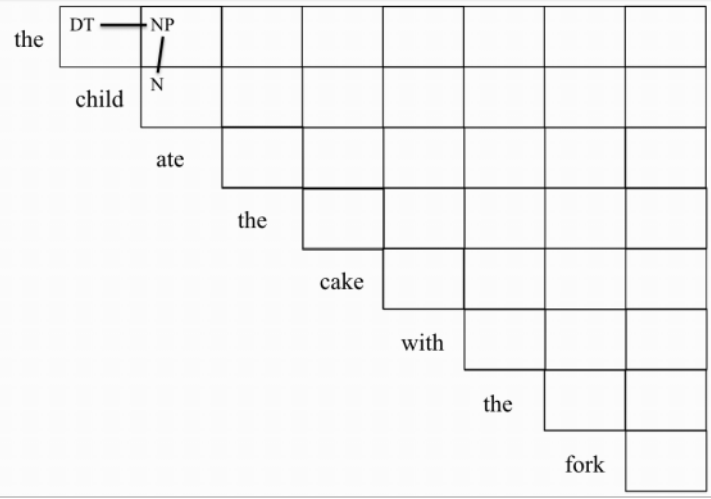
04.03 Classic Parsing Methods
NATURAL LANGUAGE PROCESSING

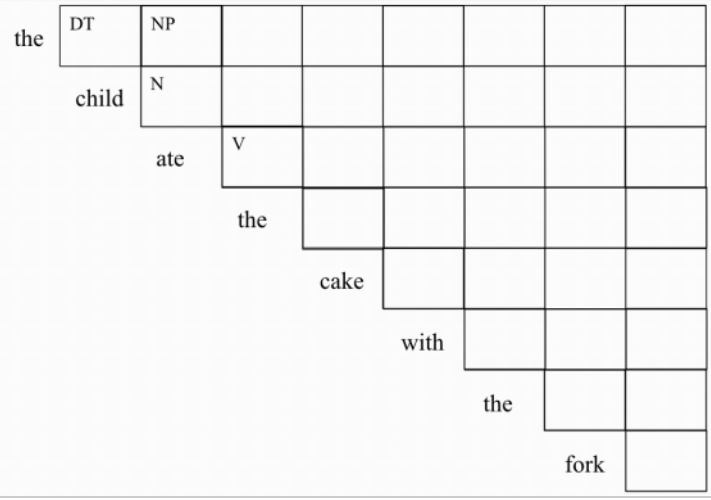


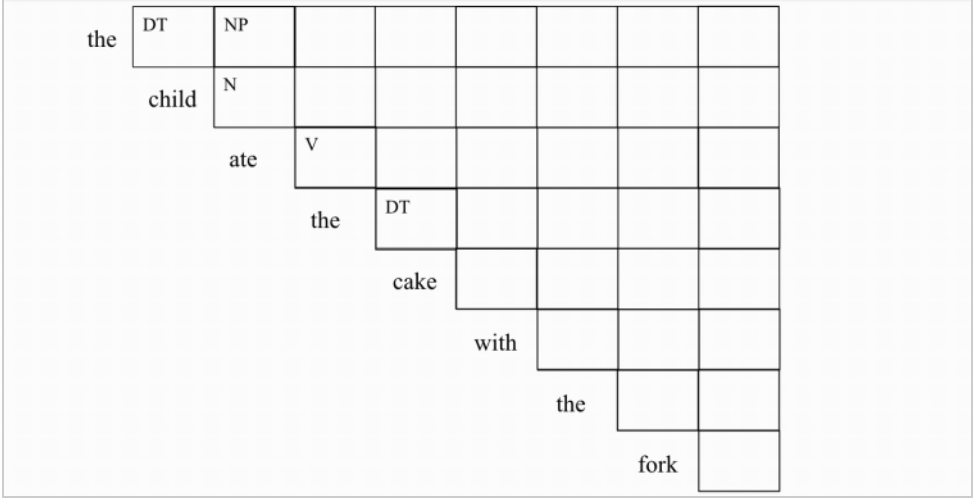
04.03 Classic Parsing Methods
NATURAL LANGUAGE PROCESSING

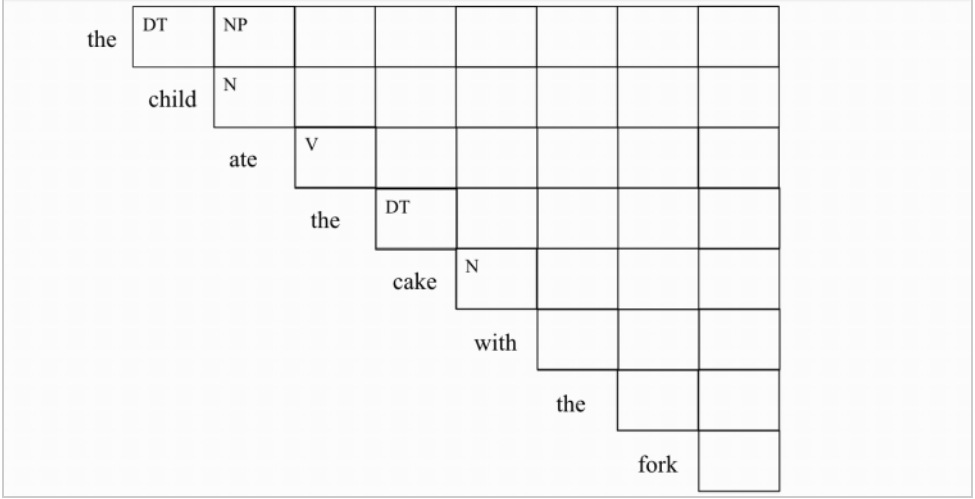


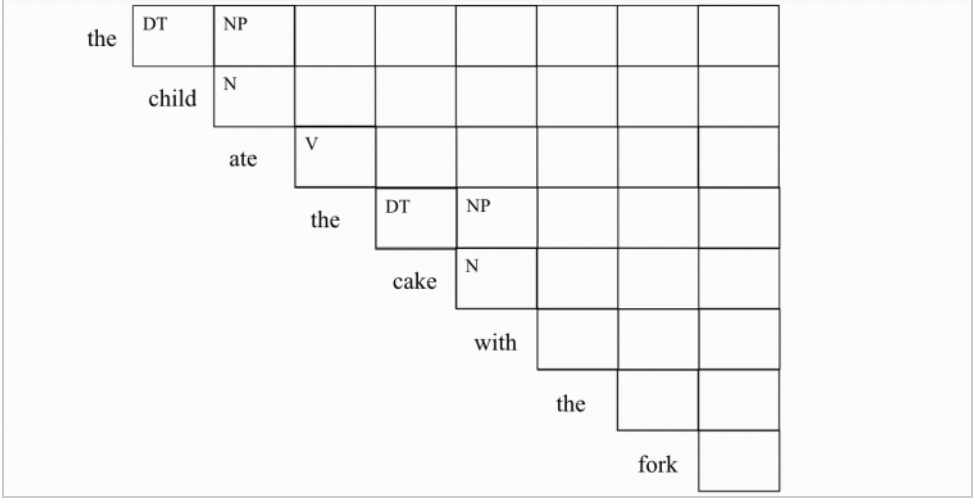

the
child
ate
the
cake
with
the
fork
DT
NP
N

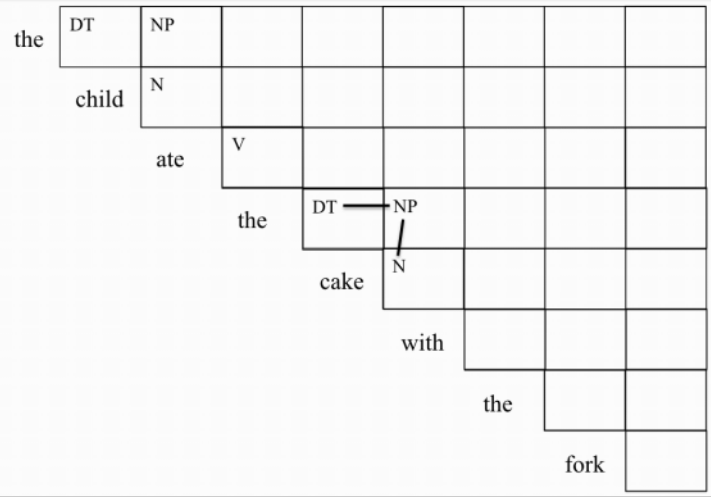

the
child
ate
the
cake
with
the
fork
DT
NP
N

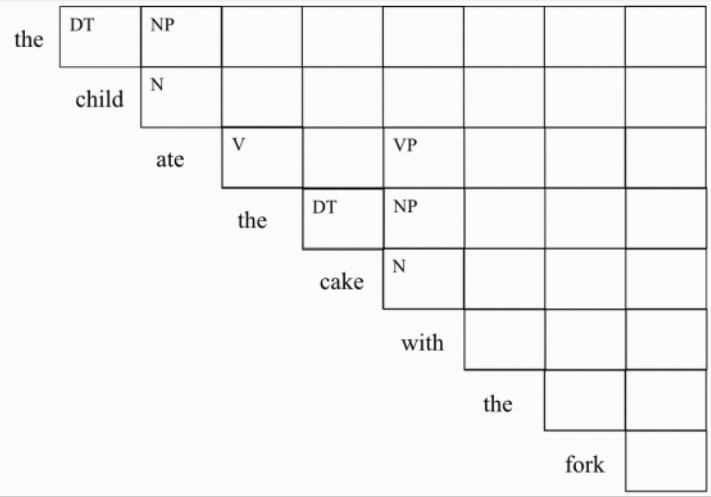

the
child
ate
the
cake
with
the
fork
DT
NP
N
V

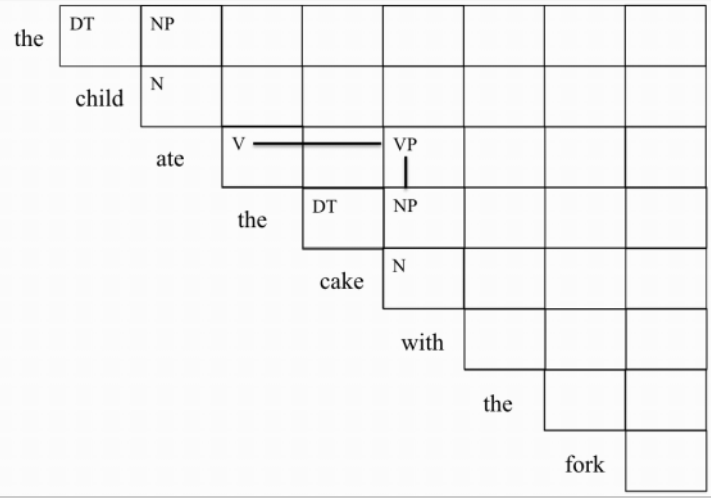

the
DT
NP
child
N
ate
V
the
DT
cake
with
the
fork

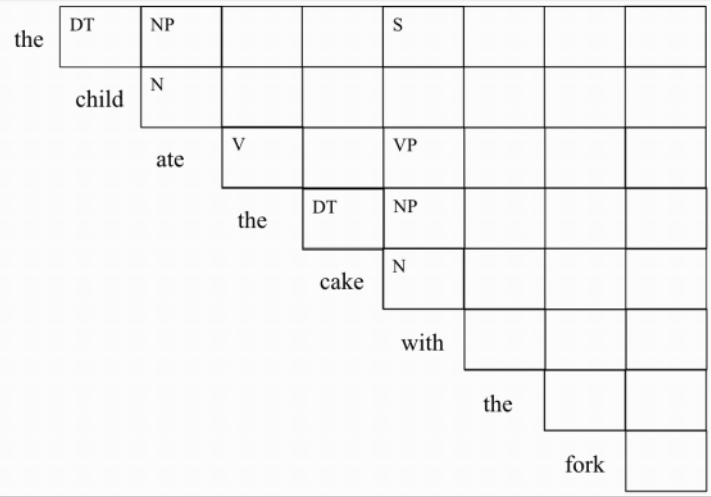

the
DT
NP
child
N
ate
V
the
DT
cake
N
with
the
fork

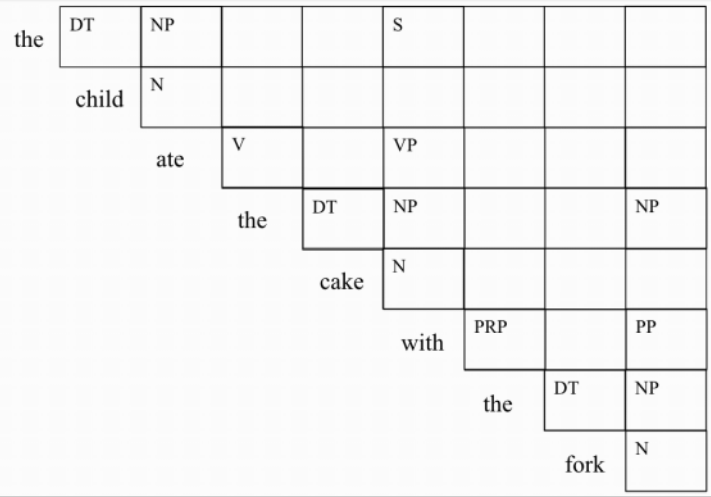

the
DT
NP
child
N
ate
V
the
DT
NP
cake
N
with
the
fork


the
child
ate
the
cake
with
the
fork
DT
NP
N
V
DT
NP
N

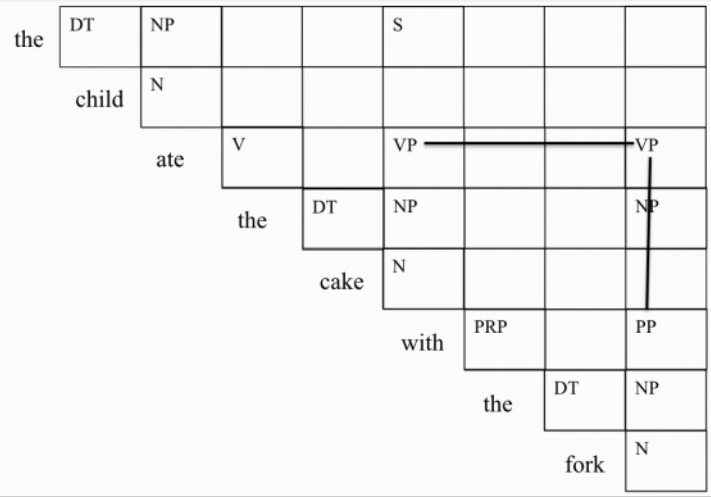

the
child
ate
the
cake
with
the
fork
DT
NP
N
V
VP
DT
NP
N


the
child
ate
the
cake
with
the
fork
DT
NP
N
V
VP
DT
NP
N

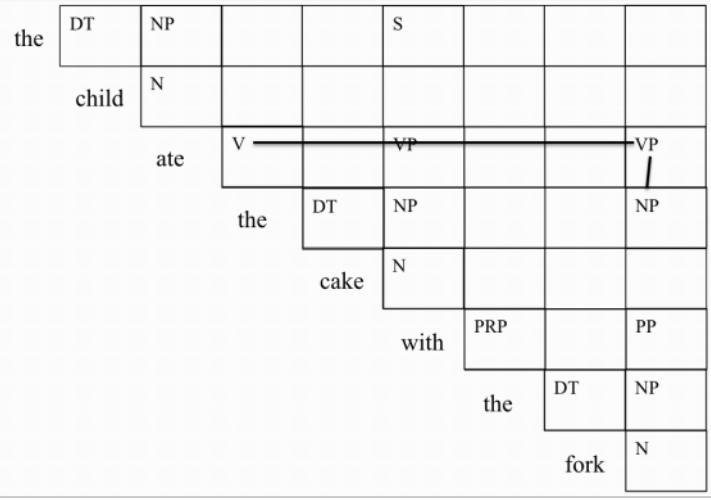

the
child
ate
the
cake
with
the
fork
DT
NP
S
N
V
VP
DT
NP
N
PRP



04.03 Classic Parsing Methods
NATURAL LANGUAGE PROCESSING

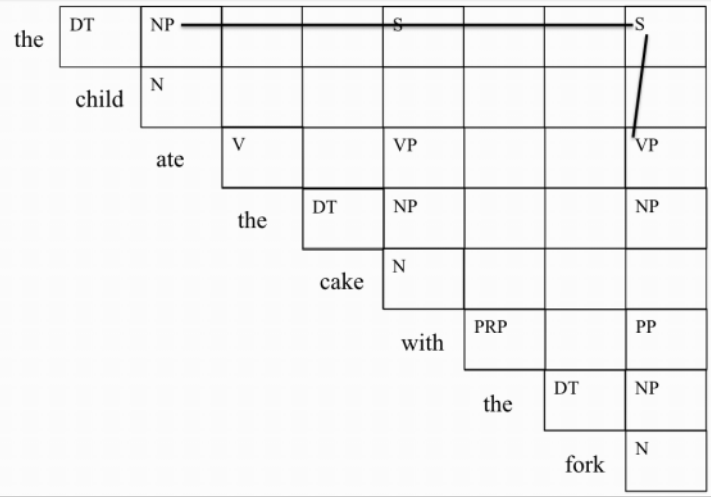


04.03 Classic Parsing Methods
NATURAL LANGUAGE PROCESSING



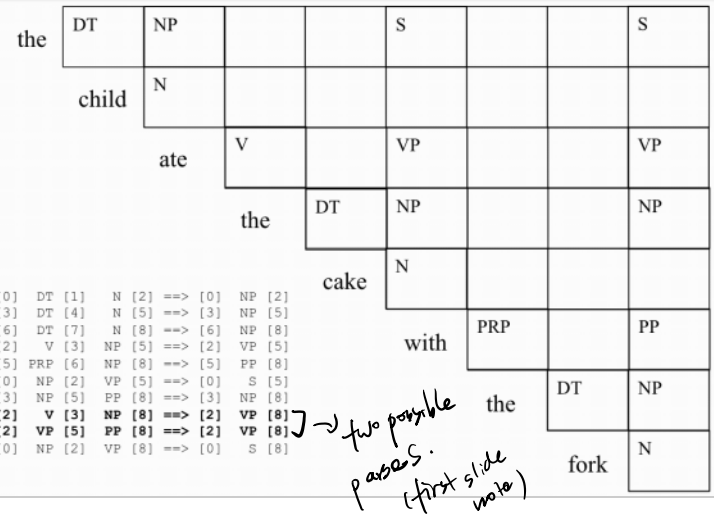
04.03 Classic Parsing Methods

NATURAL LANGUAGE PROCESSING



04.03 Classic Parsing Methods

NATURAL LANGUAGE PROCESSING

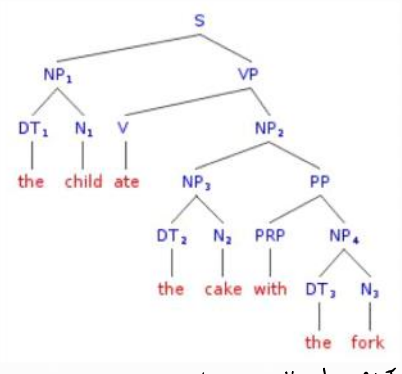


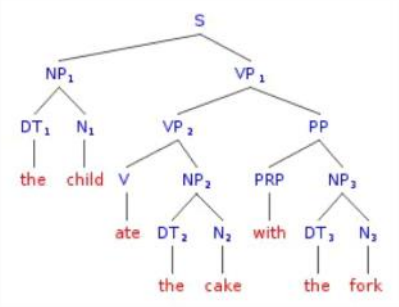

```
[0]  DT [1]   N [2] ==> [0]  NP [2]
[3]  DT [4]   N [5] ==> [3]  NP [5]
[6]  DT [7]   N [8] ==> [6]  NP [8]
[2]   V [3]  NP [5] ==> [2]  VP [5]
[5] PRP [6]  NP [8] ==> [5]  PP [8]
[0]  NP [2]  VP [5] ==> [0]   S [5]
[3]  NP [5]  PP [8] ==> [3]  NP [8]
[2]   V [3]  NP [8] ==> [2]  VP [8]
[2]  VP [5]  PP [8] ==> [2]  VP [8]
[0]  NP [2]  VP [8] ==> [0]   S [8]
```

two possible passes. (first slide note)

04.03 Classic Parsing Methods

NATURAL LANGUAGE PROCESSING





noun/lower attachment

higher attachment

What is the *meaning* of each of these sentences?

```
(S
  (NP (DT the) (N child))
  (VP
    (VP (V ate) (NP (DT the) (N cake)))
    (PP (PRP with) (NP (DT the) (N fork)))))
```



```
(S
  (NP (DT the) (N child))
  (VP
    (VP (V ate) (NP (DT the) (N cake)))
    (PP (PRP with) (NP (DT the) (N fork)))))
(S
  (NP (DT the) (N child))
  (VP
    (V ate)
    (NP
      (NP (DT the) (N cake))
      (PP (PRP with) (NP (DT the) (N fork))))))
```


Online Demo

- http://www.diotavelli.net/people/void/demos/cky.html

Complexity of CKY

- There are $O(n^2)$ cells in the table
- Single parse
  - Each cell requires a linear lookup.
  - Total time complexity is $O(n^3)$
- All parses
  - Total time complexity is exponential

? diff.
Single parse vs all parse.
都是 n^3 的

A Longer Example

```
["take", "this", "book"]

    S -> NP VP | Aux NP VP | VP
    NP -> PRON | Det Nom
    Nom -> N | Nom N | Nom PP
    PP -> PRP NP
    VP -> V | V NP | VP PP
    Det -> 'the' | 'a' | 'this'
    PRON -> 'he' | 'she'
    N -> 'book' | 'boys' | 'girl'
    PRP -> 'with' | 'in'
    V -> 'takes' | 'take'
```


Non-binary Productions

```
["take", "this", "book"]

    S -> NP VP | Aux NP VP | VP
    NP -> PRON | Det Nom
    Nom -> N | Nom N | Nom PP
    PP -> PRP NP
    VP -> V | V NP | VP PP
    Det -> 'the' | 'a' | 'this'
    PRON -> 'he' | 'she'
    N -> 'book' | 'boys' | 'girl'
    PRP -> 'with' | 'in'
    V -> 'takes' | 'take'
```

Not binarized
Can't apply CKY directly

Chomsky Normal Form

- All rules have to be in binary form:
  - $X \rightarrow Y\ Z$ or $X \rightarrow w$
- This introduces new non-terminals for
  - hybrid rules
  - n-ary rules
  - unary rules

ATIS Grammar

Original version

S → NP VP
S → Aux NP VP

S → VP

NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → VP PP
PP → Prep NP

From Jurafsky and Martin

ATIS Grammar In CNF

Original version

S → NP VP
S → Aux NP VP

S → VP

NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → VP PP
PP → Prep NP

CNF version

S → NP VP
S → X1 VP
X1 → Aux NP
S → book | include | prefer
S → Verb NP
S → VP PP
NP → I | he | she | me
NP → Houston | NWA
NP → Det Nominal
Nominal → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → book | include | prefer
VP → Verb NP
VP → VP PP
PP → Prep NP

ATIS Grammar In CNF

Original version	CNF version
S → NP VP	S → NP VP
S → Aux NP VP	S → X1 VP
	X1 → Aux NP
S → VP	**S → book | include | prefer**
	S → Verb NP
	S → VP PP
NP → Pronoun	**NP → I | he | she | me**
NP → Proper-Noun	**NP → Houston | NWA**
NP → Det Nominal	NP → Det Nominal
Nominal → Noun	**Nominal → book | flight | meal | money**
Nominal → Nominal Noun	Nominal → Nominal Noun
Nominal → Nominal PP	Nominal → Nominal PP
VP → Verb	**VP → book | include | prefer**
VP → Verb NP	VP → Verb NP
VP → VP PP	VP → VP PP
PP → Prep NP	PP → Prep NP

Chomsky Normal Form

Chomsky Normal Form.

- All rules have to be in binary form:
  - X → Y Z or X → w
- New non-terminals for hybrid rules, n-ary and unary rules:
  - INF-VP → to VP *becomes*
    - INF-VP → TO
    - TO → to
  - S → Aux NP VP *becomes*
    - S → R1 VP
    - R1 → Aux NP
  - S → VP VP → Verb VP → Verb NP VP → Verb PP *becomes*
    - S → book
    - S → buy
    - S → R2 PP
    - S → Verb PP
  - etc.

Issues with CKY

- Weak equivalence only
  - Same language, different structure
  - If the grammar had to be converted to CNF, then the final parse tree doesn't match the original grammar
  - However, it can be converted back using a specific procedure
- Syntactic ambiguity
  - (Deterministic) CKY has no way to perform syntactic disambiguation

original processing need to be done

other techniques introduced next segment


04.03 Classic Parsing Methods
NATURAL LANGUAGE PROCESSING
NLP

4.4


04.04 Earley Parser
NATURAL LANGUAGE PROCESSING
NLP

Introduction to NLP

Earley Parser

Earley's Parser

- Background
  - Developed by Jay Earley in 1970
  - No need to convert the grammar to CNF
  - Left to right
- Complexity
  - Faster than $O(n^3)$ in many cases

chomsky normal form

faster.

Earley's Parser

- Looks for both full and partial constituents
- Example:
  - S → Aux . NP VP
- When reading word *k*, it has already identified all hypotheses that are consistent with words 1 to *k*–1
- Example:
  - If the parser matches NP in the example above
  - S → Aux NP . VP

Earley's Parser

- It uses a dynamic programming table, just like CKY
- Example entry in column 1
  - [0:1] VP -> VP . PP
  - Created when processing word 1
  - Corresponds to words 0 to 1 (these words correspond to the VP part of the RHS of the rule)
  - The dot separates the completed (known) part from the incomplete (and possibly unattainable) part

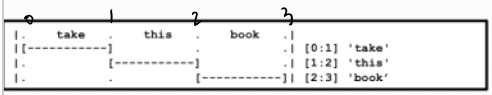
Earley's Parser

- Three types of entries
  - 'scan' – for words
  - 'predict' – for non-terminals
  - 'complete' – otherwise


```
S -> NP VP
S -> Aux NP VP
S -> VP
NP -> PRON
NP -> Det Nom
Nom -> N
Nom -> Nom N
Nom -> Nom PP
PP -> PRP NP
VP -> V
VP -> V NP
VP -> VP PP
Det -> 'the'
Det -> 'a'
Det -> 'this'
PRON -> 'he'
PRON -> 'she'
N -> 'book'
N -> 'boys'
N -> 'girl'
PRP -> 'with'
PRP -> 'in'
V -> 'takes'
V -> 'take'
```

04.04 Earley Parser

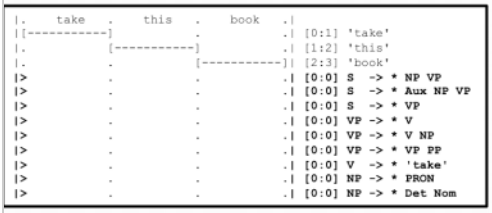
NATURAL LANGUAGE PROCESSING



Example created using NLTK

04.04 Earley Parser

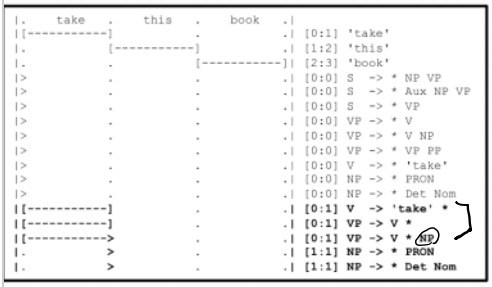
NATURAL LANGUAGE PROCESSING

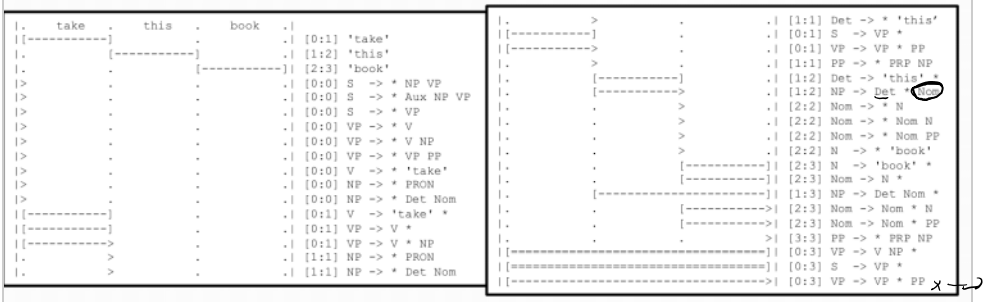


haven't seen anything
hypothesize any possible rule in the sentence

04.04 Earley Parser

NATURAL LANGUAGE PROCESSING





not going to be completed

It has some disadvantage as CKY does

NLP

Introduction to NLP

Issues with Context-free grammars

Agreement

- Number
  - Chen is/people are
- Person
  - I am/Chen is
- Tense
  - Chen was reading/Chen is reading/Chen will be reading
- Case
  - not in English but in many other languages such as German, Russian, Greek
- Gender
  - not in English but in many other languages such as German, French, Spanish

Combinatorial Explosion

- Many combinations of rules are needed to express agreement
  - S → NP VP
  - S → 1sgNP 1sgVP
  - S → 2sgNP 2sgVP
  - S → 3sgNP 3sgVP
  - …
  - 1sgNP → 1sgN
  - …

Subcategorization Frames

- Direct object
  - The dog ate a sausage
- Prepositional phrase
  - Mary left the car in the garage
- Predicative adjective
  - The receptionist looked worried
- Bare infinitive
  - She helped me buy this place
- To-infinitive
  - The girl wanted to be alone
- Participial phrase
  - He stayed crying after the movie ended
- That-clause
  - Ravi doesn't believe that it will rain tomorrow
- Question-form clauses
  - She wondered where to go

CFG independence Assumptions

- Non-independence
  - All NPs
    - 11% NP PP, 9% DT NN, 6% PRP
  - NPs under S
    - 9% NP PP, 9% DT NN, 21% PRP
  - NPs under VP
    - 23% NP PP, 7% DT NN, 4% PRP
  - (example from Dan Klein)
- Lexicalized grammars
  - later

the idea of context free grammar is flawed.

The assumption that, say, NP is the same under different structures is wrong.

Need to model this dependency

Conclusions

- Syntax helps understand the meaning of a sentence.
  - Bob gave Alice a flower
  - Who gave a flower to Alice?
  - What did Bob give to Alice?
- Context-free grammars are an appopriate representation for syntactic information with some limit
- Dynamic programming is needed for efficient parsing
  - Cubic time to find one parse
  - Still exponential time to find all parses
  - Why?

exponential time? (?)

Answer

- Why does it still take an exponential time to find all parses?
  - Very simple – because the number of parses can be exponential

cannot do faster than exponential

Solution: lexical parser

4.5

Penntree bank: important resource used to build parser

Introduction to NLP

The Penn Treebank

Description

- Background
  - From the early 90's
  - Developed at the University of Pennsylvania
  - (Marcus, Santorini, and Marcinkiewicz 1993)
- Size
  - 40,000 training sentences
  - 2400 test sentences
- Genre
  - Mostly Wall Street Journal news stories and some spoken conversations
- Importance
  - Helped launch modern automatic parsing methods

External Links

- Treebank-3
  - http://catalog.ldc.upenn.edu/LDC99T42
- Original version
  - http://catalog.ldc.upenn.edu/LDC95T7
- Tokenization guidelines
  - http://www.cis.upenn.edu/~treebank/tokenization.html
- The American National Corpus
  - http://www.americannationalcorpus.org/OANC/penn.html

Penn Treebank tagset (1/2)

Tag	Description	Example
CC	coordinating conjunction	and
CD	cardinal number	1, third
DT	determiner	the
EX	existential there	*there* is
FW	foreign word	d'oeuvre
IN	preposition/subordinating conjunction	in, of, like
JJ	adjective	green
JJR	adjective, comparative	greener
JJS	adjective, superlative	greenest
LS	list marker	1)
MD	modal	could, will
NN	noun, singular or mass	table
NNS	noun plural	tables
NNP	proper noun, singular	John
NNPS	proper noun, plural	Vikings
PDT	predeterminer	*both* the boys
POS	possessive ending	friend*'s*

Penn Treebank tagset (2/2)

Tag	Description	Example
PRP	personal pronoun	I, he, it
PRP$	possessive pronoun	my, his
RB	adverb	however, usually, naturally, here, good
RBR	adverb, comparative	better
RBS	adverb, superlative	best
RP	particle	give *up*
TO	to	*to* go, *to* him
UH	interjection	uhhuhhuhh
VB	verb, base form	take
VBD	verb, past tense	took
VBG	verb, gerund/present participle	taking
VBN	verb, past participle	taken
VBP	verb, sing. present, non-3d	take
VBZ	verb, 3rd person sing. present	takes
WDT	wh-determiner	which
WP	wh-pronoun	who, what
WP$	possessive wh-pronoun	whose
WRB	wh-abverb	where, when

Example Sentence

- **`WSJ/12/WSJ_1273.MRG, sentence 11`**
- Because the CD had an effective yield of 13.4 % when it was issued in 1984 , and interest rates in general had declined sharply since then , part of the price Dr. Blumenfeld paid was a premium -- an additional amount on top of the CD 's base value plus accrued interest that represented the CD 's increased market value .

Parsed sentence

```
(S
  (SBAR-PRP
    (IN Because)
    (S
      (S
        (NP-SBJ (DT the) (NNP CD))
        (VP
          (VBD had)
          (NP
            (NP (DT an) (JJ effective) (NN yield))
            (PP (IN of) (NP (CD 13.4) (NN %))))
          (SBAR-TMP
            (WHADVP-4 (WRB when))
            (S
              (NP-SBJ-1 (PRP it))
              (VP
                (VBD was)
                (VP
                  (VBN issued)
                  (NP (-NONE- *-1))
                  (PP-TMP (IN in) (NP (CD 1984)))
                  (ADVP-TMP (-NONE- *T*-4)))))))))
                     ...
```

04.05 The Penn Treebank
NATURAL LANGUAGE PROCESSING

```
(S
  (SBAR-PRP
    (IN Because)
    (S
      (S
        (NP-SBJ (DT the) (NNP CD))
        (VP
          (VBD had)
          (NP
            (NP (DT an) (JJ effective) (NN yield))
            (PP (IN of) (NP (CD 13.4) (NN %))))
          (SBAR-TMP
            (WHADVP-4 (WRB when))
            (S
              (NP-SBJ-1 (PRP it))
              (VP
                (VBD was)
                (VP
                  (VBN issued)
                  (NP (-NONE- *-1))
                  (PP-TMP (IN in) (NP (CD 1984)))
                  (ADVP-TMP (-NONE- *T*-4)))))))))
      (, ,)
      (CC and)
      (S
        (NP-SBJ
          (NP (NN interest) (NNS rates))
          (PP (IN in) (ADJP (JJ general))))
        (VP
          (VBD had)
          (VP
            (VBN declined)
            (ADVP-MNR (RB sharply))
            (PP-TMP (IN since) (NP (RB
then)))))))))
(, ,)
(NP-SBJ
  (NP (NN part))
  (PP
    (IN of)
    (NP
      (NP (DT the) (NN price))
      (SBAR
        (WHNP-3 (-NONE- 0))
        (S
          (NP-SBJ (NNP Dr.) (NNP Blumenfeld))
          (VP (VBD paid) (NP (-NONE- *T*-3)))))))))
(VP
  (VBD was)
  (NP-PRD
    (NP (DT a) (NN premium))
    (: --)
    (NP
      (NP
        (NP (DT an) (JJ additional) (NN amount))
        (PP-LOC
          (IN on)
          (NP
            (NP (NN top))
            (PP
              (IN of)
              (NP
                (NP (DT the) (NNP CD) (POS 's))
                (NN base)
                (NN value))))))
      (CC plus)
      (NP (VBN accrued) (NN interest)))))
  (SBAR
    (WHNP-2 (WDT that))
    (S
      (NP-SBJ (-NONE- *T*-2))
      (VP
        (VBD represented)
        (NP
          (NP (DT the) (NNP CD) (POS 's))
          (VBN increased)
          (NN market)
          (NN value)))))))
(. .))
```

04.05 The Penn Treebank
NATURAL LANGUAGE PROCESSING

```
(S
  (SBAR-PRP
    (IN Because)
    (S
      (S
        (NP-SBJ (DT the) (NNP CD))
        (VP
          (VBD had)
          (NP
            (NP (DT an) (JJ effective) (NN yield))
            (PP (IN of) (NP (CD 13.4) (NN %))))
          (SBAR-TMP
            (WHADVP-4 (WRB when))
            (S
              (NP-SBJ-1 (PRP it))
              (VP
                (VBD was)
                (VP
                  (VBN issued)
                  (NP (-NONE- *-1))
                  (PP-TMP (IN in) (NP (CD 1984)))
                  (ADVP-TMP (-NONE- *T*-4)))))))))
      (, ,)
      (CC and)
      (S
        (NP-SBJ
          (NP (NN interest) (NNS rates))
          (PP (IN in) (ADJP (JJ general))))
        (VP
          (VBD had)
          (VP
            (VBN declined)
            (ADVP-MNR (RB sharply))
            (PP-TMP (IN since) (NP (RB
then)))))))))
(, ,)
(NP-SBJ
  (NP (NN part))
  (PP
    (IN of)
    (NP
      (NP (DT the) (NN price))
      (SBAR
        (WHNP-3 (-NONE- 0))
        (S
          (NP-SBJ (NNP Dr.) (NNP Blumenfeld))
          (VP (VBD paid) (NP (-NONE- *T*-3)))))))))
(VP
  (VBD was)
  (NP-PRD
    (NP (DT a) (NN premium))
    (: --)
    (NP
      (NP
        (NP (DT an) (JJ additional) (NN amount))
        (PP-LOC
          (IN on)
          (NP
            (NP (NN top))
            (PP
              (IN of)
              (NP
                (NP (DT the) (NNP CD) (POS 's))
                (NN base)
                (NN value))))))
      (CC plus)
      (NP (VBN accrued) (NN interest)))))
  (SBAR
    (WHNP-2 (WDT that))
    (S
      (NP-SBJ (-NONE- *T*-2))
      (VP
        (VBD represented)
        (NP
          (NP (DT the) (NNP CD) (POS 's))
          (VBN increased)
          (NN market)
          (NN value)))))))
(. .))
```

Allows conjunctions

Peculiarities

- Complementizers
  - e.g., "that"
- Gaps
  - *NONE*
- SBAR
  - SBAR → COMP S
  - E.g., ("that *NONE* represented the CD' market value")

tgrep

`A < B`	A immediately dominates B
`A << B`	A dominates B
`A <- B`	B is the last child of A
`A <<, B`	B is a leftmost descendant of A
`A <<` B`	B is a rightmost descendant of A
`A . B`	A immediately precedes B
`A .. B`	A precedes B
`A $ B`	A and B are sisters
`A $. B`	A and B are sisters and A immediately precedes B
`A $.. B`	A and B are sisters and A precedes B

The Use Of Treebanks

- Disadvantages
  - A lot more work to annotate 40K+ sentences than to write a grammar.
- Advantages
  - Statistics about different constituents and phenomena
  - Training systems
  - Evaluating systems
  - Multilingual extensions

* There's Chinese Penn-treebank European language. Korean ...

Introduction to NLP

Parsing evaluation

Evaluation Methodology (1/2)

NLP classification

- Classification tasks
  - Document retrieval
  - Part of speech tagging — the way to label a word in POS etc.
  - Parsing
- Data split ★
  - Training
  - Dev-test → seperate a part as an official test set
  - Test → Don't look at it more than once!

Evaluation Methodology (2/2)

- Baselines
  - Dumb baseline → e.g. label every word as noun
  - Intelligent baseline
  - Human performance (ceiling) → if human disagree ... say at most they agree on 98% of the labeling. then the programs performance should not > 98%
- New method
- Evaluation methods
  - Accuracy
  - Precision and Recall
- Multiple references
  - Interjudge agreement → chance that human coder pick the same label.

Kappa

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)}$$

- Agreement vs. expected agreement
  - P(A) is the level of agreement of the judges
  - P(E) is the expected probability of agreement by chance
- When $\kappa > .7$ – agreement is considered high
- Question
  - Judge agreement on a binary classification task is 60%, is this high?

Kappa — Normalized performance of your sys. → If say 0.3, 0.4 the task is not well-defined should not consider it at all. if random benchmark 50% ✓

04.05 The Penn Treebank — NATURAL LANGUAGE PROCESSING

Answer

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)}$$

- Data
  - P(A) = .6
  - P(E) = .5
- Kappa
  - k = .1/.5 = .2
  - not high

In this case we say the task is not well-defined. $\frac{0.6-0.5}{1-0.5} = 0.2$

04.05 The Penn Treebank — NATURAL LANGUAGE PROCESSING

Parsing Evaluation

- Precision and recall
  - get the proper constituents
- Labeled precision and recall
  - also get the correct non-terminal labels
- F1
  - harmonic mean of precision and recall
- Crossing brackets
  - (A (B C)) vs ((A B) C)
- PTB corpus
  - training 02-21, development 22, test 23

label precision: the label of each nonterminal is right to get credit. BC should go together while your sys does not have them go together → Crossing bracket error. evaluating parser. ? 没看出来怎算的

04.05 The Penn Treebank — NATURAL LANGUAGE PROCESSING

Evaluation Example

```
GOLD = (S (NP (DT The) (JJ Japanese) (JJ industrial) (NNS companies))
        (VP (MD should) (VP (VB know) (ADVP (JJR better)))) (. .)
```

Evaluation Example

```
GOLD = (S (NP (DT The) (JJ Japanese) (JJ industrial) (NNS companies))
          (VP (MD should) (VP (VB know) (ADVP (JJR better)))) (. .)

CHAR = (S (NP (DT The) (JJ Japanese) (JJ industrial) (NNS companies))
           (VP (MD should) (VP (VB know)) ((ADVP (RBR better)))) (. .))
```

Bracketing Recall	= 80.00
Bracketing Precision	= 66.67
Bracketing FMeasure	= 72.73
Complete match	= 0.00
No crossing	= 100.00
Tagging accuracy	= 87.50

8/10 are picked.
6/9
not completely parsed correctly

04.05 The Penn Treebank NATURAL LANGUAGE PROCESSING

NLP