NLP
Introduction to NLP

Cocke-Kasami-Younger (CKY) Parsing
Notes on Left Recursion

• Problematic for many parsing methods
  – Infinite loops when expanding
• But appropriate linguistically
  – NP → DT N
  – NP → PN
  – DT → NP ‘s
  – Mary’s mother’s sister’s friend
Chart Parsing

• Top-down parsers have problems with expanding the same non-terminal
  – In particular, pre-terminals such as POS
  – Bad idea to use top-down (recursive descent) parsing as is

• Bottom-up parsers have problems with generating locally feasible subtrees that are not viable globally

• Chart parsing will address these issues
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 $n$
Dynamic Programming

- CKY (Cocke–Kasami–Younger)
  - bottom-up
  - requires a normalized (binarized) grammar
- Earley parser
  - top-down
  - more complicated
  - (separate lecture)
function cky (sentence W, grammar G) returns table

    for i in 1..length(W) do
        table[i-1,i] = {A|A->Wi in G}
    end for

    for j in 2..length(W) do
        for i in j-2 down to 0 do
            for k in (i+1) to (j-1) do
                table[i,j] = table[i,j] union {A|A->BC in G, B in table [I,k], C in table [k,j]}
            end for
        end for
    end for

If the start symbol S is in table [0,n] then W is in L(G)
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'
the child ate the cake with the fork
the child ate the cake with the fork
the child ate the cake with the fork.
the child ate the cake with the fork
the child ate the cake with the fork
the child ate the cake with the fork.
the child ate the cake with the fork.
the child ate the cake with the fork.
the child ate the cake with the fork
the child ate the cake with the fork
the child ate the cake with the fork
The child ate the cake with the fork.
the child ate the cake with the fork.
the child ate the cake with the fork
the child ate the cake with the fork
the child ate the cake with the fork
the child ate the cake with the fork
The child ate the cake with the fork.
The child ate the cake with the fork.
the child ate the cake with the fork.
What is the meaning of each of these sentences?
the child ate the cake with the 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))))))
Complexity of CKY

• **Space complexity**
  – 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
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'
Chomsky Normal Form (CNF)

- 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
  - epsilon rules (e.g., $NP \rightarrow \varepsilon$)
- Any CFG can be converted to CNF (See Aho & Ullman p. 152)
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

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
Chomsky Normal Form

• All rules have to be in binary form:
  – $X \rightarrow Y Z$ or $X \rightarrow w$

• New non–terminals for hybrid rules, n–ary and unary rules:
  – $INF-VP \rightarrow to VP$ becomes
    • $INF-VP \rightarrow TO$
    • $TO \rightarrow to$
  – $S \rightarrow Aux NP VP$ becomes
    • $S \rightarrow R1 VP$
    • $R1 \rightarrow Aux NP$
  – $S \rightarrow VP$ $VP \rightarrow Verb$ $VP \rightarrow Verb NP$ $VP \rightarrow Verb PP$ becomes
    • $S \rightarrow book$
    • $S \rightarrow buy$
    • $S \rightarrow R2 PP$
    • $S \rightarrow Verb PP$
  – 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
Notes

• Demo:

• Recognizing vs. parsing
  – Recognizing just means determining if the string is part of the language defined by the CFG
  – Parsing is more complicated – it involves producing a parse tree
NLP