Dynamic Programming Parsing

Dynamic Programming Parsing

- To avoid extensive repeated work, must cache intermediate results, i.e.,completed phrases.
- Dynamic programming algorithms based on both top-down and bottom-up search can achieve $O(n^3)$ recognition time where n is the length of the input string.

Dynamic Programming Parsing Methods

1. CKY (Cocke-Kasami-Younger) algorithm: bottom-up, requires normalizing the grammar
2. Chart Parsers - retain completed phrases in a chart and can combine top-down and bottom-up searches.
3. Earley Parser - top-down, does not require normalizing grammar, more complex

The CYK Algorithm

CYK Algorithm

- The **Cocke–Younger–Kasami algorithm** (alternatively called **CYK**, or **CKY**) is a parsing algorithm for context-free grammars, named after its inventors, John Cocke, Daniel Younger and Tadao Kasami.
- It employs bottom-up parsing and dynamic programming
- It determines if a sentence is in the language generated by grammar.

The CYK Algorithm

- Problem:
  - Given a context-free grammar **G** and a string **w**
    - **G** = (V, ∑,P , S) where
      - ➢ V finite set of variables
      - ➢ ∑(the alphabet) finite set of terminal symbols
      - ➢ P finite set of rules
      - ➢ S start symbol (distinguished element of V)
      - ➢ V and ∑ are assumed to be disjoint
    - **G** is used to generate the string of a language
- Question:
  - Is **w** in **L**(**G**)?

CYK Algorithm Basics

- The Structure of the rules in a Chomsky Normal Form grammar

- Uses a "dynamic programming" or "table-filling algorithm"

Chomsky Normal Form

- *Normal Form* is described by a set of conditions that each rule in the grammar must satisfy
- Context-free grammar is in CNF if each rule has one of the following forms:

$A \rightarrow BC$	at most 2 symbols on right
$A \rightarrow a$, or	side terminal symbol
$S \rightarrow \epsilon$	null string

where $B, C \in V - \{S\}$

Converting to CNF

Original Grammar

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
Pronoun → I | he | she | me
Noun → book | flight | meal | money
Verb → book | include | prefer
Proper-Noun → Houston | NWA
Aux → is, am
PP → on, in, over

Converting to CNF

Original Grammar

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
Pronoun → I | he | she | me
Noun → book | flight | meal | money
Verb → book | include | prefer
Proper-Noun → Houston | NWA
Aux → is, am
PP → on, in, over

Chomsky Normal Form

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
Noun → book | flight | meal | money
Verb → book | include | prefer
Aux → is, am
PP → on, in, over

CYK Algorithm

Construct a Triangular Table

- Each row corresponds to some length of substrings

Construct a Triangular Table

0 w_1 1 w_2 2 w_3 3 w_4 4 w_5 5

$x_{0,1}$	$x_{0,2}$	$x_{0,3}$	$x_{0,4}$	$x_{0,5}$
	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$	$x_{1,5}$
		$x_{2,3}$	$x_{2,4}$	$x_{2,5}$
			$x_{3,4}$	$x_{3,5}$
				$x_{4,5}$

Table for string ‘**w**’ that has length 5

Example CYK Algorithm

- Show that the sentence s ϵ L(G) [where s= **“a pilot likes flying planes”**, using CYK Algorithm with the following G:

Grammar G

S → NP VP	VBZ → likes
VP → VBG NNS	VBG → flying
VP→ VBZ VP	DT → a
VP→ VBZ NP	NN → pilot
NP →DT NN	JJ → flying
NP → JJ NNS	NNS → planes

Construct a Triangular Table

0 a 1 pilot 2 likes 3 flying 4 planes 5

a	pilot	likes	flying	planes
$x_{0,1}$	$x_{0,2}$	$x_{0,3}$	$x_{0,4}$	$x_{0,5}$
	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$	$x_{1,5}$
		$x_{2,3}$	$x_{2,4}$	$x_{2,5}$
			$x_{3,4}$	$x_{3,5}$
				$x_{4,5}$

Table for sentence that has length 5

Construct a Triangular Table

0 a 1 pilot 2 likes 3 flying 4 planes 5

a	pilot	likes	flying	planes
DT $x_{0,1}$	$x_{0,2}$	$x_{0,3}$	$x_{0,4}$	$x_{0,5}$
	NN $x_{1,2}$	$x_{1,3}$	$x_{1,4}$	$x_{1,5}$
		VBZ $x_{2,3}$	$x_{2,4}$	$x_{2,5}$
			VBG , JJ $x_{3,4}$	$x_{3,5}$
				NNS $x_{4,5}$

$x_{0,1}$ = DT -> a

$x_{1,2}$ = NN -> pilot

$x_{2,3}$ = VBZ -> likes

$x_{3,4}$ = VBG -> flying, JJ ->flying

$x_{4,5}$ = NNS -> planes

Construct a Triangular Table

0 a 1	1 pilot 2	2 likes 3	3 flying 4	4 planes 5
DT $x_{0,1}$	NP $x_{0,2}$	$x_{0,3}$	$x_{0,4}$	$x_{0,5}$
	NN $x_{1,2}$	-- $x_{1,3}$	$x_{1,4}$	$x_{1,5}$
		VBZ $x_{2,3}$	-- $x_{2,4}$	$x_{2,5}$
			VBG , JJ $x_{3,4}$	VP, NP $x_{3,5}$
				NNS $x_{4,5}$

$x_{0,2} = x_{0,1} \quad x_{1,2}$

$= \text{DT NN} = \text{NP}$

$x_{1,3} = x_{1,2} \quad x_{2,3}$

$= \text{NN VBZ},$

$\text{NN JJ} = \varphi$

$_0$ a $_1$	pilot $_2$	likes $_3$	flying $_4$	planes $_5$
DT $x_{0,1}$	NP $x_{0,2}$	-- $x_{0,3}$	$x_{0,4}$	$x_{0,5}$
	NN $x_{1,2}$	-- $x_{1,3}$	-- $x_{1,4}$	$x_{1,5}$
		VBZ $x_{2,3}$	-- $x_{2,4}$	VP1=VBZ VP VP2=VBZ NP $x_{2,5}$
			VBG , JJ $x_{3,4}$	VP, NP $x_{3,5}$
				NNS $x_{4,5}$

$$x_{0,3} = x_{0,1}\, x_{1,3},\ x_{0,2}\, x_{2,3}$$
$$= \text{DT --},\quad \text{NP VBZ}$$
$$= \varphi$$

$$x_{1,4} = x_{1,2}\, x_{2,4}\,,\ x_{1,3}\, x_{3,4}$$
$$= \text{NN --, --VBG, --JJ}$$
$$= \varphi$$

$$x_{2,5} = x_{2,3}\, x_{3,5}\,,\ x_{2,4}\, x_{4,5}$$
$$= \text{VBZ VP, VBZ NP, -- NNS}$$
$$= \text{VP1, VP2}$$

0 a 1	pilot 2	likes 3	flying 4	planes 5
DT $x_{0,1}$	NP $x_{0,2}$	-- $x_{0,3}$	-- $x_{0,4}$	$x_{0,5}$
	NN $x_{1,2}$	-- $x_{1,3}$	-- $x_{1,4}$	-- $x_{1,5}$
		VBZ $x_{2,3}$	-- $x_{2,4}$	VP1=VBZ VP VP2=VBZ NP $x_{2,5}$
			VBG , JJ $x_{3,4}$	VP, NP $x_{3,5}$
				NNS $x_{4,5}$

$x_{0,4} = x_{0,1}x_{1,4}$, $x_{0,2}x_{2,4}$, $x_{0,3}$ $x_{3,4}$ = DT--, NP--, -- VBG ,

-- JJ = = φ

$x_{1,5} = x_{1,2}x_{2,5}$, $x_{1,3}$ $x_{3,5}$, $x_{1,4}$ $x_{4,5}$ = NN VP1,NN VP2,-- VP, --NP, --NNS= φ

0 a 1	pilot 2	likes 3	flying 4	planes 5
DT $x_{0,1}$	NP $x_{0,2}$	-- $x_{0,3}$	-- $x_{0,4}$	S=NP VP1 S=NP VP2 $x_{0,5}$
	NN $x_{1,2}$	-- $x_{1,3}$	-- $x_{1,4}$	-- $x_{1,5}$
		VBZ $x_{2,3}$	-- $x_{2,4}$	VP1=VBZ VP VP2=VBZ NP $x_{2,5}$
			VBG , JJ $x_{3,4}$	VP, NP $x_{3,5}$
				NNS $x_{4,5}$

$x_{0,5} = x_{0,1}\, x_{1,5},\ x_{0,2}\, x_{2,5},\ x_{0,3}\, x_{3,5},\ x_{0,4}\, x_{4,5}$

=DT --,NP VP1,NP VP2, -- VP/NP, -- NNS

$= S_1, S_2$

Theorem

- The CYK Algorithm correctly computes $X_{i\,j}$ for all i and j; thus sentence S is in L(G) if and only if S is in X_{0n}.
- The running time of the algorithm is $O(n^3)$.

Question

Analyze the following sentence using CYK algorithm. Does the sentence belongs to the L(G) ?

Astronomers saw stars with telescope

Grammar(G):

S → NP VP

VP → VP PP

VP → V NP

NP → NP PP

PP → P NP

NP → stars

NP → telescope

NP → saw

V → saw

P → with

NP -> astronomers

Question

- Show the CYK Algorithm with the following example:
  - CNF grammar **G**
    - S → AB | BC
    - A → BA | a
    - B → CC | b
    - C → AB | a
  - **w** is ababa
  - Question Is **ababa** in L(G)?
- Basics of CYK Algorithm
  - The Structure of the rules in a Chomsky Normal Form grammar
  - Uses a "dynamic programming" or "table-filling algorithm"
- Complexity O(n^3)