

# CS 5000: Lecture 14

Vladimir Kulyukin

Department of Computer Science

Utah State University

# Outline

- CFL closure properties
- Cocke-Younger-Kasami algorithm for CFL membership

# CFL Closure Properties

CFLs are closed under concatenation.

# Closure under Concatenation

If  $L_1$  and  $L_2$  are any context - free languages,  $L_1L_2$ ,  
is context - free.

Proof : Let  $G_1 = (V_1, \Sigma_1, S_1, P_1)$  and  $G_2 = (V_2, \Sigma_2, S_2, P_2)$   
such that  $L(G_1) = L_1$  and  $L(G_2) = L_2$ . Assume that  $V_1$  and  
 $V_2$  are disjoint. This does not lose generality, because the  
symbols can be renamed. Construct a new grammar

$G = (V, \Sigma, S, P)$  where :

$$V = V_1 \cup V_2 \cup \{S\};$$

$$\Sigma = \Sigma_1 \cup \Sigma_2;$$

$$P = P_1 \cup P_2 \cup \{(S \rightarrow S_1S_2)\}.$$

# CFL Closure Properties

CFLs are closed under Kleene closure.

# Closure under Kleene Closure

If  $L$  is a context - free language, then  $L^*$   
is context - free.

Proof : Let  $G = (V, \Sigma, S, P)$  such that  $L(G) = L$ .

$G' = (V', \Sigma', S', P')$  such that

$$V' = V \cup \{S'\};$$

$$P' = P \cup \{(S' \rightarrow SS'), (S' \rightarrow \varepsilon)\}.$$

# Closure under Intersection with Regular Languages

If  $L_1$  is a CFL and  $L_2$  is a regular language, then  $L_1 \cap L_2$  is context - free.

# Non-closure Properties

- The CFLs are not closed under intersection.
- The CFLs are not closed under complement.

# Non-closure under Intersection

Theorem : CFL's are not closed under intersection.

Proof : Consider two languages  $\{a^m b^n c^n\}$  and  $\{a^n b^n c^m\}$ .

Both of these languages are CF. However,

$\{a^m b^n c^n\} \cap \{a^n b^n c^m\} = \{a^n b^n c^n\}$ , which is not CF.

# Non-closure under Complement

Theorem : CFL's are not closed under complement.

Proof : Suppose they are. Let  $L_1$  and  $L_2$  be CFL's.

We know that CFL's are closed under union. But

then  $\overline{\overline{L_1} \cup \overline{L_2}}$  is also CFL. But  $\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$ .

# String Membership in a CFL

- Given a CFG  $G = (V, T, P, S)$  and a string  $x$  in  $T^*$ , is  $x$  in  $L(G)$ ?
- Cocke-Younger-Kasami (CYK) algorithm takes a CFG in Chomsky Normal Form and a string and returns true or false depending on whether  $x$  is in  $L(G)$ .
- The CYK algorithm runs in  $O(n^3)$ , where  $|x|=n$ .

# CYK Algorithm: Two Basic Insights

- The input is a CNF grammar  $G$  and a string  $x$  such that  $|x| \geq 1$ .
- For each  $i$ ,  $1 \leq i \leq n$ , and  $j$  and each variable  $V$  check whether  $V \rightarrow^* x_{ij}$  where  $x_{ij}$  is the substring of  $x$  of length  $j$  beginning at position  $i$ .

# Why Does CYK Work?

- We can induct on  $j$ .
- In the base case  $j=1$ .  $V \rightarrow^* x_{ij}$  if and only if  $V \rightarrow x_{ij}$  is a production, since  $x_{i_1}$  is a string of length 1.

# Why Does CYK Work?

- For  $j > 1$ ,  $V \rightarrow^* x_{ij}$  if and only if there is some production  $V \rightarrow BC$  and some  $k$ ,  $1 \leq k < j$ , such that  $B \rightarrow^* x_{ik}$  and  $C \rightarrow^* x_{(i+k)(j-k)}$ .

b	a	a	b	a
---	---	---	---	---

Example: if  $i = 1$  and  $j = 5$ , then  $1 \leq k < 5$ .

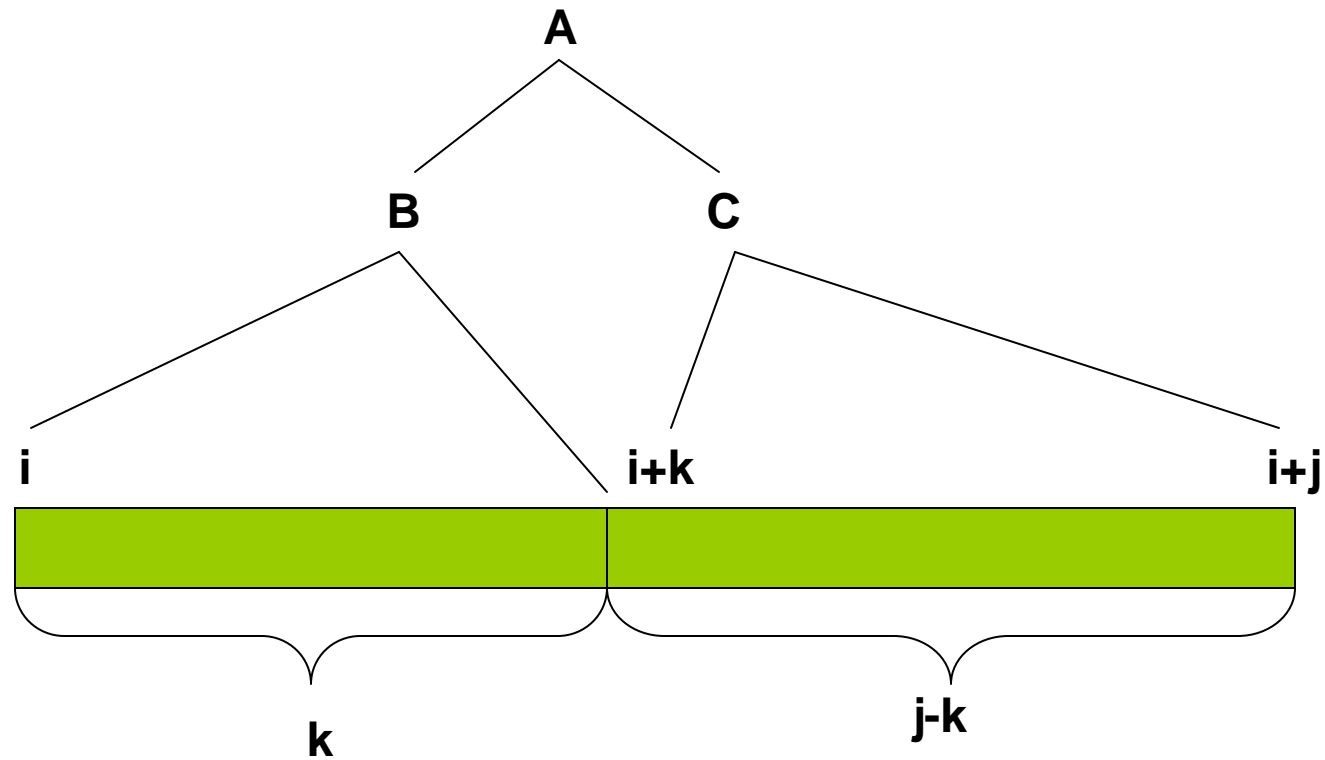
If  $k = 1$ , then  $B \rightarrow^* x_{11}$  and  $C \rightarrow^* x_{24}$ .

If  $k = 2$ , then  $B \rightarrow^* x_{12}$  and  $C \rightarrow^* x_{33}$ .

If  $k = 3$ , then  $B \rightarrow^* x_{13}$  and  $C \rightarrow^* x_{42}$ .

If  $k = 4$ , then  $B \rightarrow^* x_{14}$  and  $C \rightarrow^* x_{51}$ .

# Why Does CYK Work?



# Why Does CYK Work?

- Since  $k$  and  $j-k$  are both less than  $j$ , we already know whether each of the last two derivations exists.
- When we reach  $j=n$ , we can determine if  $S \rightarrow^* x_{1n}$ .

# CYK Algorithm

$V_{ij}$  is the set of variables  $A$  such that  $A \rightarrow^* x_{ij}$

for  $i=1$  to  $n$  {

$V_{i1} = \{V \mid V \rightarrow a \text{ is a production and the } i^{\text{th}} \text{ symbol of } x \text{ is } a\}$   
}

for  $j=2$  to  $n$  {

    for  $i=1$  to  $n-j+1$  {

$V_{ij} = \{\}$ ;

        for  $k=1$  to  $j-1$  {

$V_{ij} = V_{ij} + \{V \mid V \rightarrow BC \text{ is a production, } B \text{ is in } V_{ik} \text{ and } C \text{ is in } V_{(i+k)(j-k)}\}$ ;

        }

    }

}

# Example

$S \rightarrow AB|BC$

$A \rightarrow BA|a$

$B \rightarrow CC|b$

$C \rightarrow AB|a$

# Input: baaba

**i**

	1	2	3	4	5
1					
2					
3					
4					
5					

**j**

Input: baaba

- Computing  $V_{11}$ , so  $i=1, j=1$ .
- The first symbol is b.
- We need to find a production  $V \rightarrow b$ .
- There is only one production:
  - $B \rightarrow b$

Input: baaba

i

	1	2	3	4	5
1	B				
2					
3					
4					
5					

# Input: baaba

- Computing  $V_{21}$ , so  $i=2, j=1$ .
- The second symbol is a.
- We need to find productions of the form  $V \rightarrow a$ .
- There are two productions:
  - $A \rightarrow a$
  - $C \rightarrow a$

Input: baaba

i

	1	2	3	4	5
1	B	A, C			
2					
3					
4					
5					

j

# Input: baaba

- Computing  $V_{31}$ ,  $i=3$ ,  $j=1$ .
- The third symbol is a.
- We need to find productions of the form  $V \rightarrow a$ .
- There are two productions:
  - $A \rightarrow a$
  - $C \rightarrow a$

# Input: baaba

i

	1	2	3	4	5
1	B	A, C	A, C		
2					
3					
4					
5					

j

# Input: baabaa

- Computing  $V_{41}$ ,  $i=4$ ,  $j=1$ .
- The fourth symbol is b.
- We need to find productions of the form  $V \rightarrow b$ .
- There is only one production:
  - $B \rightarrow b$

Input: baaba

i

	1	2	3	4	5
1	B	A, C	A, C	B	
2					
3					
4					
5					

j

Input: baabaa

- $V_{51}$ ,  $i=5$ ,  $j=1$ .
- The fifth symbol is a.
- We need to find productions of the form  $V \rightarrow a$ .
- There is only one production:
  - $A \rightarrow a$
  - $C \rightarrow a$

Input: baabaa

		i				
		1	2	3	4	5
j	1	B	A,C	A,C	B	A,C
	2					
	3					
	4					
	5					

# Input: baaba

- Computing  $V_{12}$ , so  $i=1, j=2$
- In this case  $k$  goes from 1 to 1.
- Need rules  $V \rightarrow BC$  where  $B$  is in  $V_{11}$  and  $C$  is in  $V_{21}$ .
- $V_{11} = \{B\}$ ;  $V_{21} = \{A, C\}$ . The possibilities for  $BC$  are  $BA, BC$ . The actual rules with these right-hand sides in the grammar are
  - $S \rightarrow BA|BC$
  - $A \rightarrow BA$

Input: baabaa

		i				
		1	2	3	4	5
j	1	B	A,C	A,C	B	A,C
	2	S, A				
	3					
	4					
	5					

# Input: baaba

- Computing  $V_{22}$ , so  $i=2, j=2$
- In this case  $k$  goes from 1 to 1.
- Need rules  $V \rightarrow BC$  where  $B$  is in  $V_{21}$  and  $C$  is in  $V_{31}$ .
- $V_{21} = \{A, C\}$ ;  $V_{21} = \{A, C\}$ . The possibilities for  $BC$  are  $AA, AC, CA, CC$ . The actual rules with these right-hand sides in the grammar are:
  - $B \rightarrow CC$

Input: baabaa

		i				
		1	2	3	4	5
j	1	B	A,C	A,C	B	A,C
	2	S, A	B			
	3					
	4					
	5					

# Input: baaba

- Computing  $V_{32}$ ,  $i=3$ ,  $j=2$
- In this case,  $k$  goes from 1 to 1.
- Need rules  $V \rightarrow BC$  where  $B$  is in  $V_{31}$  and  $C$  is in  $V_{41}$ .
- $V_{31} = \{A, C\}$ ;  $V_{41} = \{B\}$ . The possibilities for  $BC$  are  $AB$ ,  $CB$ . The actual rules with these right-hand sides in the grammar are:
  - $S \rightarrow AB$
  - $C \rightarrow AB$

Input: baabaa

		i				
		1	2	3	4	5
j	1	B	A,C	A,C	B	A,C
	2	S, A	B	S,C		
	3					
	4					
	5					

Input: baabaa

		i				
		1	2	3	4	5
j	1	B	A,C	A,C	B	A,C
	2	S, A	B	S,C	S,A	
	3	{	B	B		
	4	{	S,A,C			
	5	S,A,C				