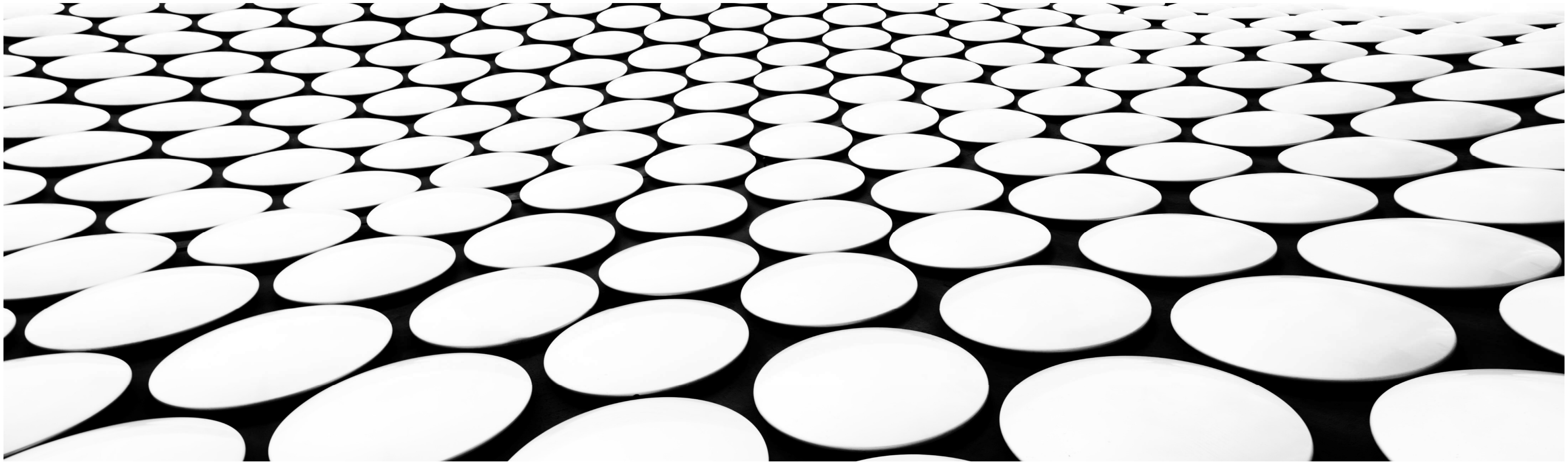
CHOMSKY NORMAL FORM

IF 2124 TEORI BAHASA FORMAL OTOMATA

Judhi S.



Chomsky Normal Form, CNF

We shall show that every nonempty CFL without ϵ has a grammar G without useless symbols, and such that every production is of the form

- $A \to BC$, where $\{A, B, C\} \subseteq T$, or

- $A \to \alpha$, where $A \in V$, and $\alpha \in T$.

To achieve this, start with any grammar for the CFL, and

1. "Clean up" the grammar.

2. Arrange that all bodies of length 2 or more consists of only variables.

3. Break bodies of length 3 or more into a cascade of two-variable-bodied productions.

• For step 2, for every terminal a that appears in a body of length ≥ 2, create a new variable, say A, and replace a by A in all bodies.

Then add a new rule $A \to a$.

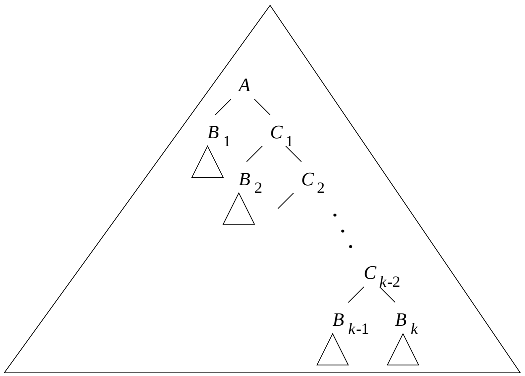
• For step 3, for each rule of the form

$$A \to B_1 B_2 \cdots B_k,$$

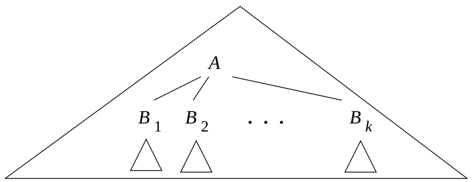
$k \geq 3$, introduce new variables $C_1, C_2, \ldots C_{k-2}$, and replace the rule with

$$\begin{array}{rcl} A & \to & B_1 C_1 \\ C_1 & \to & B_2 C_2 \\ & \ldots & \\ C_{k-3} & \to & B_{k-2} C_{k-2} \\ C_{k-2} & \to & B_{k-1} B_k \end{array}$$

Illustration of the effect of step 3



(a)



(b)

Example of CNF conversion

Let's start with the grammar (step 1 already done)

$E \rightarrow E + T \mid T * F \mid (E) \mid a \mid b \mid Ia \mid Ib \mid I0 \mid I1$
$T \rightarrow T * F \mid (E)a \mid b \mid Ia \mid Ib \mid I0 \mid I1$
$F \rightarrow (E) \;\; a \mid b \mid Ia \mid Ib \mid I0 \mid I1$
$I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

For step 2, we need the rules
$A \rightarrow a, B \rightarrow b, Z \rightarrow 0, O \rightarrow 1$
$P \rightarrow +, M \rightarrow *, L \rightarrow (, R \rightarrow)$
and by replacing we get the grammar

$E \rightarrow EPT \mid TMF \mid LER \mid a \mid b \mid IA \mid IB \mid IZ \mid IO$
$T \rightarrow TMF \mid LER \mid a \mid b \mid IA \mid IB \mid IZ \mid IO$
$F \rightarrow LER \mid a \mid b \mid IA \mid IB \mid IZ \mid IO$
$I \rightarrow a \mid b \mid IA \mid IB \mid IZ \mid IO$
$A \rightarrow a, B \rightarrow b, Z \rightarrow 0, O \rightarrow 1$
$P \rightarrow +, M \rightarrow *, L \rightarrow (, R \rightarrow)$

For step 3, we replace

$E \to EPT$ by $E \to EC_1, C_1 \to PT$

$E \to TMF, T \to TMF$ by
$E \to TC_2, T \to TC_2, C_2 \to MF$

$E \to LER, T \to LER, F \to LER$ by
$E \to LC_3, T \to LC_3, F \to LC_3, C_3 \to ER$

The final CNF grammar is

$E \to EC_1 \mid TC_2 \mid LC_3 \mid a \mid b \mid IA \mid IB \mid IZ \mid IO$
$T \to TC_2 \mid LC_3 \mid a \mid b \mid IA \mid IB \mid IZ \mid IO$
$F \to LC_3 \mid a \mid b \mid IA \mid IB \mid IZ \mid IO$
$I \to a \mid b \mid IA \mid IB \mid IZ \mid IO$
$C_1 \to PT, C_2 \to MF, C_3 \to ER$
$A \to a, B \to b, Z \to 0, O \to 1$
$P \to +, M \to *, L \to (, R \to)$

homomorphism $h\ \Sigma \longrightarrow \Gamma$

Closure Properties of CFL's

Consider a mapping

$$s : \Sigma \to 2^{\Delta^*}$$

where Σ and Δ are finite alphabets. Let $w \in \Sigma^*$, where $w = a_1 a_2 \cdots a_n$, and define

$$s(a_1 a_2 \cdots a_n) = s(a_1).s(a_2).\cdots.s(a_n)$$

and, for $L \subseteq \Sigma^*$,

$$s(L) = \bigcup_{w \in L} s(w)$$

Such a mapping s is called a *substitution*.

Example: $\Sigma = \{0, 1\}, \Delta = \{a, b\}$,
$s(0) = \{a^n b^n : n \geq 1\}, s(1) = \{aa, bb\}$.

Let $w = 01$. Then $s(w) = s(0).s(1) =$ $\{a^n b^n\}\{aa, bb\}$
$\{\underbrace{a^n b^n aa} : n \geq 1\} \cup \{\underbrace{a^n b^{n+2}} : n \geq 1\}$

Let $L = \{0\}^*$. Then $s(L) = (s(0))^* =$
$\{a^{n_1} b^{n_1} a^{n_2} b^{n_2} \cdots a^{n_k} b^{n_k} : k \geq 0, n_i \geq 1\}$ $\{a^{n_1} b^{n_1}, a^{n_1} b^{n_1} a^{n_1} b^{n_1}, \quad k \geq 0 \}$.

Theorem 7.23: Let L be a CFL over Σ, and s a substitution, such that $s(a)$ is a CFL, $\forall a \in \Sigma$. Then $s(L)$ is a CFL.

We start with grammars

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

for L, and

$$G_a = (V_a, T_a, P_a, S_a)$$

for each $s(a)$. We then construct

$$G' = (V', T', P', S')$$

where

$V' = (\bigcup_{a \in \Sigma} V_a) \cup V$

$T' = \bigcup_{a \in \Sigma} T_a$

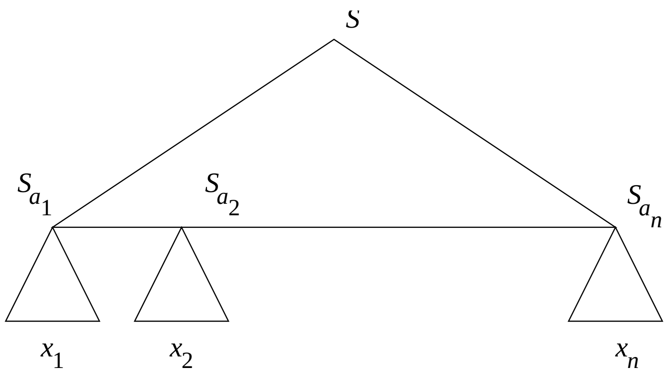
$P' = \bigcup_{a \in \Sigma} P_a$ plus the productions of P with each a in a body replaced with symbol S_a.

Now we have to show that

- $L(G') = s(L)$.

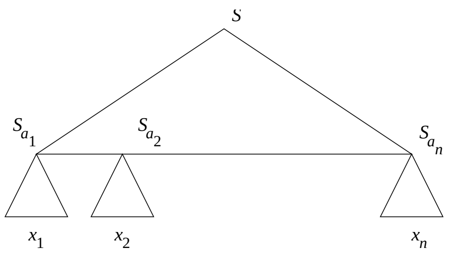
Let $w \in s(L)$. Then $\exists x = a_1 a_2 \cdots a_n$ in L, and $\exists x_i \in s(a_i)$, such that $w = x_1 x_2 \cdots x_n$.

A derivation tree in G' will look like



Thus we can generate $S_{a_1} S_{a_2} \cdots S_{a_n}$ in G' and form there we generate $x_1 x_2 \cdots x_n = w$. Thus $w \in L(G')$.

Then let $w \in L(G')$. Then the parse tree for w must again look like



Now delete the dangling subtrees. Then you have yield

$$S_{a_1} S_{a_2} \cdots S_{a_n}$$

where $a_1 a_2 \cdots a_n \in L(G)$. Now w is also equal to $s(a_1 a_2 \cdots a_n)$, which is in $S(L)$.

Applications of the Substitution Theorem

Theorem 7.24: The CFL's are closed under (i) : union, (ii) : concatenation, (iii) : Kleene closure and positive closure $+$, and (iv) : homomorphism.

Proof: (i): Let L_1 and L_2 be CFL's, let $L = \{1, 2\}$, and $s(1) = L_1, s(2) = L_2$.
Then $L_1 \cup L_2 = s(L)$.

(ii) : Here we choose $L = \{12\}$ and s as before.
Then $L_1.L_2 = s(L)$

(iii) : Suppose L_1 is CF. Let $L = \{1\}^*, s(1) = L_1$. Now $L_1^* = s(L)$. Similar proof for $+$.

(iv) : Let L_1 be a CFL over Σ, and h a homomorphism on Σ. Then define s by

$$a \mapsto \{h(a)\}$$

Then $h(L) = s(L)$.

Theorem: If L is CF, then so in L^R.

Proof: Suppose L is generated b $G = (V, T, P, S)$. Construct $G^R = (V, T, P^R, S)$, where

$$P^R = \{A \to \alpha^R : A \to \alpha \in P\}$$

Show at home by inductions on the lengths of the derivations in G (for one direction) and in G^R (for the other direction) that $(L(G))^R = L(G^R)$.

Let $L_1 = \{0^n 1^n 2^i : n \geq 1, i \geq 1\}$. The L_1 is CF with grammar

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow 0A1|01 \\ B &\rightarrow 2B|2 \end{aligned}$$

Also, $L_2 = \{0^i 1^n 2^n : n \geq 1, i \geq 1\}$ is CF with grammar

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow 0A|0 \\ B &\rightarrow 1B2|12 \end{aligned}$$

However, $L_1 \cap L_2 = \{0^n 1^n 2^n : n \geq 1\}$ which is not CF (see the handout on course-page).

Theorem 7.27: If L is CR, and R regular, then $L \cap R$ is CF.

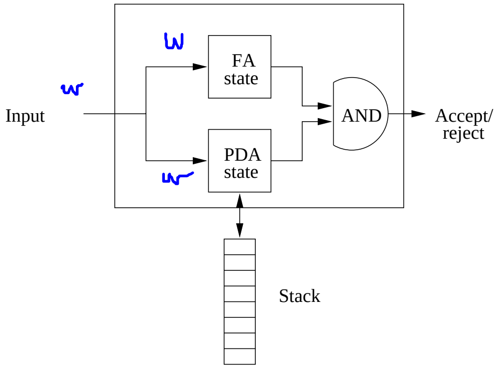
Proof: Let L be accepted by PDA

$$P = (Q_P, \Sigma, \Gamma, \delta_P, q_P, Z_0, F_P)$$

by final state, and let R be accepted by DFA

$$A = (Q_A, \Sigma, \delta_A, q_A, F_A)$$

We'll construct a PDA for $L \cap R$ according to the picture



Formally, define

$$P' = (Q_P \times Q_A, , \Sigma, \Gamma, \delta, (q_P, q_A), Z_0, F_P \times F_A)$$

where

$$\delta((q,p),a,X) = \{((r, \hat{\delta}_A(p,a)), \gamma) : (r,\gamma) \in \delta_P(q,a,X)\}$$

Prove at home by an induction $\overset{*}{\vdash}$, both for P and for P' that

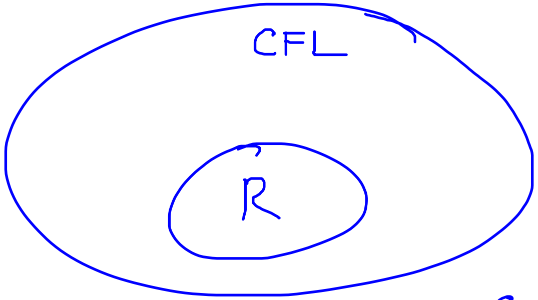
$$(q_P, w, Z_0) \overset{*}{\vdash} (q, \epsilon, \gamma) \text{ in } P$$

if and only if

$$((q_P, q_A), w, Z_0) \overset{*}{\vdash} \Big((q, \hat{\delta}(p_A, w)), \epsilon, \gamma\Big) \text{ in } P'$$

The claim the follows (Why?)

Theorem 7.29: Let L, L_1, L_2 be CFL's and R regular. Then



1. $L \setminus R$ is CF
2. $\bar{L}$ is not necessarily CF
3. $L_1 \setminus L_2$ is not necessarily CF

$S \to 01 \mid A$
$A \to 000A111 \mid \varepsilon$

Proof:

1. $\bar{R}$ is regular, $L \cap \bar{R}$ is ~~regular~~ CF, and $L \cap \bar{R} = L \setminus R$.

   $L = \{0^n 1^n\}$ $\quad R = \{(00)^*(11)\}$

01	11
~~0011~~	0011
000111	000011
⋮	⋮

2. If $\bar{L}$ always was CF, it would follow that

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

   always would be CF.

   tidak selalu CF

3. Note that Σ^* is CF, so if $L_1 \setminus L_2$ was always CF, then so would $\Sigma^* \setminus L = \bar{L}$.

   Σ^* selalu CF
   L selalu CF
   $\Sigma^* \setminus L = \bar{L}$

Inverse homomorphism

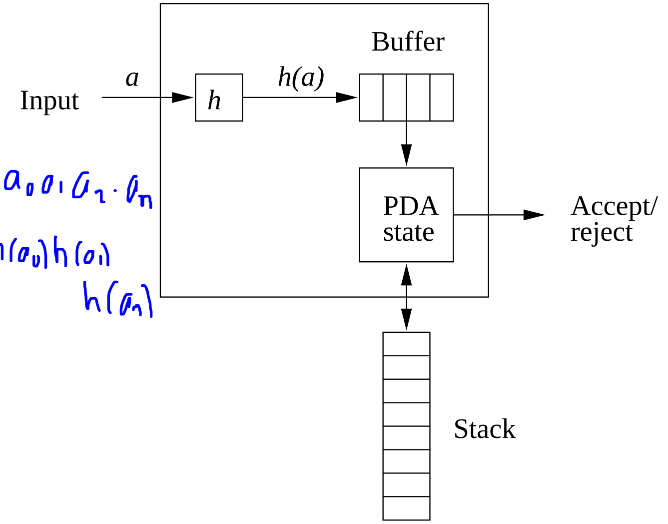
Let $h : \Sigma \rightarrow \Theta^*$ be a homom. Let $L \subseteq \Theta^*$, and define

$$h^{-1}(L) = \{w \in \Sigma^* : h(w) \in L\}$$

Now we have

Theorem 7.30: Let L be a CFL, and h a homomorphism. Then $h^{-1}(L)$ is a CFL.

Proof: The plan of the proof is



$w = a_0 a_1 a_2 \cdot a_n$

$= h(a_0) h(a_1)$

$h(a_n)$

CFL:

Tertutup: - Union
- Concate
- Kleen / Positive
- homomorphism.

Yang tidak:
- intersection / irisan
- complement
- difference.

Let L be accepted by PDA

$$P = (Q, \Theta, \Gamma, \delta, q_0, Z_0, F)$$

$a \longrightarrow X_1 X_1 X_3.$

We construct a new PDA

$$P' = (Q', \Sigma, \Gamma, \delta', (q_0, \epsilon), Z_0, F \times \{\epsilon\})$$

where

$$Q' = \{(q, x) : q \in Q, x \in \textit{suffix}(h(a)), a \in \Sigma\}$$

$$\delta'((q, \epsilon), a, X) = \{((q, h(a)), X) : \epsilon \neq a \in \Sigma, q \in Q, X \in \Gamma\}$$

$$\delta'\overbrace{((q, \boxed{b}x), \epsilon, X) = \{((p, \boxed{x}), \gamma)}^{\text{homom}} : \overbrace{(p, \gamma) \in \delta(q, b, X)}^{\text{awalnya.}}, b \in T \cup \{\epsilon\}, q \in Q, X \in \Gamma\}$$

Show at home by suitable inductions that

- $(q_0, h(w), Z_0) \overset{*}{\vdash} (p, \epsilon, \gamma)$ in P if and only if $((q_0, \epsilon), w, Z_0) \overset{*}{\vdash} ((p, \epsilon), \epsilon, \gamma)$ in P'.

Decision Properties of CFL's

We'll look at the following:

- Complexity of converting among CFA's and PDAQ's

- Converting a CFG to CNF

- Testing $L(G) \neq \emptyset$, for a given G

- Testing $w \in L(G)$, for a given w and fixed G.

- Preview of undecidable CFL problems

Converting between CFA's and PDA's

- Input size is n.
- n is the *total* size of the input CFG or PDA.

The following work in time $O(n)$

1. Converting a CFG to a PDA (slide 203)
2. Converting a "final state" PDA to a "null stack" PDA (slide 199)
3. Converting a "null stack" PDA to a "final state" PDA (slide 195)

Avoidable exponential blow-up

For converting a PDA to a CFG we have

(slide 210)

At most n^3 variables of the form $[pXq]$

If $(\mathbf{r}, Y_1 Y_2 \cdots Y_k) \in \delta(\mathbf{q}, a, X)\}$, we'll have $O(n^n)$ rules of the form

$$[\mathbf{q} X r_k] \to a[\mathbf{r} Y_1 r_1] \cdots [r_{k-1} Y_k r_k]$$

- By introducing $k-2$ new states we can modify the PDA to push at most *one* symbol per transition. Illustration on blackboard in class.

- Now, k will be ≤ 2 for all rules.

- Total length of all transitions is still $O(n)$.

- Now, each transition generates at most n^2 productions

- Total size (and time to calculate) the grammar is therefore $O(n^3)$.

Converting into CNF

Good news:

1. Computing $r(G)$ and $g(G)$ and eliminating useless symbols takes time $O(n)$. This will be shown shortly

   (slides 229,232,234)

2. Size of $u(G)$ and the resulting grammar with productions P_1 is $O(n^2)$

   (slides 244,245)

3. Arranging that bodies consist of only variables is $O(n)$

   (slide 248)

4. Breaking of bodies is $O(n)$ (slide 248)

Bad news:

- Eliminating the nullable symbols can make the new grammar have size $O(2^n)$

(slide 236)

The bad news are avoidable:

Break bodies first before eliminating nullable symbols

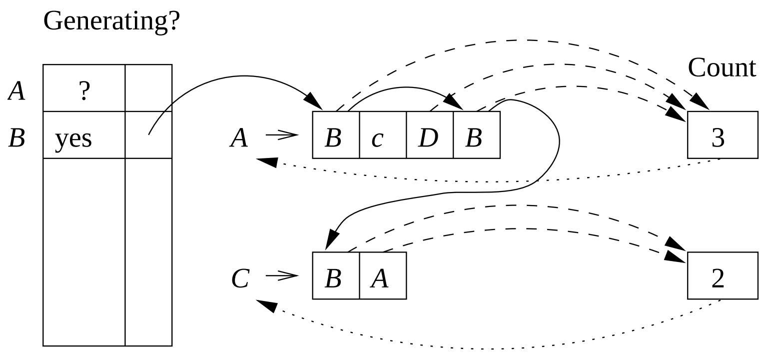
- Conversion into CNF is $O(n^2)$

Testing emptiness of CFL's

$L(G)$ is non-empty if the start symbol S is generating.

A naive implementation on $g(G)$ takes time $O(n^2)$.

$g(G)$ can be computed in time $O(n)$ as follows:



Creation and initialzation of the array is $O(n)$

Creation and initialzation of the links and counts is $O(n)$

When a count goes to zero, we have to

1. Finding the head variable A, checkin if it already is "yes" in the array, and if not, queueing it is $O(1)$ per production. Total $O(n)$

2. Following links for A, and decreasing the counters. Takes time $O(n)$.

Total time is $O(n)$.

$\boxed{w \in L(G)\textbf{?}}$

Inefficient way:

Suppose G is CNF, test string is w, with $|w| = n$. Since the parse tree is binary, there are $2n - 1$ internal nodes.

Generate *all* binary parse trees of G with $2n-1$ internal nodes.

Check if any parse tree generates w

CYK-algo for membership testing

The grammar G is fixed

Input is $w = a_1 a_2 \cdots a_n$

We construct a triangular table, where X_{ij} contains all variables A, such that

$$A \overset{*}{\underset{G}{\Rightarrow}} a_i a_{i+1} \cdots a_j$$

$$\begin{array}{|lllll}
X_{15} & & & & \\
X_{14} & X_{25} & & & \\
X_{13} & X_{24} & X_{35} & & \\
X_{12} & X_{23} & X_{34} & X_{45} & \\
X_{11} & X_{22} & X_{33} & X_{44} & X_{55} \\
\hline
a_1 & a_2 & a_3 & a_4 & a_5
\end{array}$$

To fill the table we work row-by-row, upwards

The first row is computed in the basis, the subsequent ones in the induction.

Basis: $X_{ii} == \{A : A \rightarrow a_i$ is in $G\}$

Induction:

We wish to compute X_{ij}, which is in row $j - i + 1$.

$A \in X_{ij}$, if
$A \stackrel{*}{\Rightarrow} a_i a_i + 1 \cdots a_j$, if
for some $k < j$, and $A \rightarrow BC$, we have
$B \stackrel{*}{\Rightarrow} a_i a_{i+1} \cdots a_k$, and $C \stackrel{*}{\Rightarrow} a_{k+1} a_{k+2} \cdots a_j$, if
$B \in X_{ik}$, and $C \in X_{kj}$

Example:

G has productions

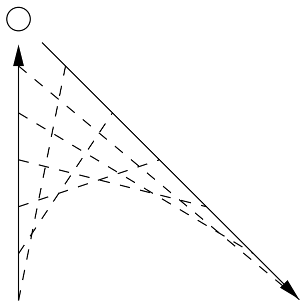
$$
\begin{aligned}
S &\rightarrow AB|BC \\
A &\rightarrow BA|a \\
B &\rightarrow CC|b \\
C &\rightarrow AB|a
\end{aligned}
$$

$\{S,A,C\}$				
-	$\{S,A,C\}$			
-	$\{B\}$	$\{B\}$		
$\{S,A\}$	$\{B\}$	$\{S,C\}$	$\{S,A\}$	
$\{B\}$	$\{A,C\}$	$\{A,C\}$	$\{B\}$	$\{A,C\}$
b	a	a	b	a

To compute X_{ij} we need to compare at most n pairs of previously computed sets:

$$(X_{ii}, X_{i=1,j}), (X_{i,i+1}, X_{i+2,j}), \ldots, (X_{i,j-1}, X_{jj})$$

as suggested below



For $w = a_1 \cdots a_n$, there are $O(n^2)$ entries X_{ij} to compute.

For each X_{ij} we need to compare at most n pairs $(X_{ik}, X_{k+1,j})$.

Total work is $O(n^3)$.