Properties of CFL's

- *Simplification* of CFG's. This makes life easier, since we can claim that if a language is CF, then it has a grammar of a special form.

- *Pumping Lemma for CFL's.* Similar to the regular case.

- *Closure properties.* Some, but not all, of the closure properties of regular languages carry over to CFL's.

- *Decision properties.* We can test for membership and emptiness, but for instance, equivalence of CFL's is undecidable.

Chomsky Normal Form

We want to show that every CFL (without ϵ) is generated by a CFG where all productions are of the form

$$A \to BC, \text{ or } A \to a$$

where A, B, and C are variables, and a is a terminal. This is called CNF, and to get there we have to

1. Eliminate *useless symbols*, those that do not appear in any derivation $S \stackrel{*}{\Rightarrow} w$, for start symbol S and terminal w.

2. Eliminate *ϵ-productions*, that is, productions of the form $A \to \epsilon$.

3. Eliminate *unit productions*, that is, productions of the form $A \to B$, where A and B are variables.

Eliminating Useless Symbols

- A symbol X is *useful* for a grammar $G = (V, T, P, S)$, if there is a derivation

$$S \overset{*}{\underset{G}{\Rightarrow}} \alpha X \beta \overset{*}{\underset{G}{\Rightarrow}} w$$

for a teminal string w. Symbols that are not useful are called *useless*.

- A symbol X is *generating* if $X \overset{*}{\underset{G}{\Rightarrow}} w$, for some $w \in T^*$

- A symbol X is *reachable* if $S \overset{*}{\underset{G}{\Rightarrow}} \alpha X \beta$, for some $\{\alpha, \beta\} \subseteq (V \cup T)^*$

It turns out that if we eliminate non-generating symbols first, and then non-reachable ones, we will be left with only useful symbols.

Example: Let G be

$$S \rightarrow AB|a, \; A \rightarrow b$$

S and A are generating, B is not. If we eliminate B we have to eliminate $S \rightarrow AB$, leaving the grammar

$$S \rightarrow a, \; A \rightarrow b$$

Now only S is reachable. Eliminating A and b leaves us with

$$S \rightarrow a$$

with language $\{a\}$.

OTH, if we eliminate non-reachable symbols first, we find that all symbols are reachable. From

$$S \rightarrow AB|a, \; A \rightarrow b$$

we then eliminate B as non-generating, and are left with

$$S \rightarrow a, \; A \rightarrow b$$

that still contains useless symbols

Theorem 7.2: Let $G = (V, T, P, S)$ be a CFG such that $L(G) \neq \emptyset$. Let $G_1 = (V_1, T_1, P_1, S)$ be the grammar obtained by

1. Eliminating all nongenerating symbols and the productions they occur in. Let the new grammar be $G_2 = (V_2, T_2, P_2, S)$.

2. Eliminate from G_2 all nonreachable symbols and the productions they occur in.

Then G_1 has no useless symbols, and $L(G_1) = L(G)$.

Proof: We first prove that G_1 has no useless symbols:

Let X remain in $V_1 \cup T_1$. Thus $X \stackrel{*}{\Rightarrow} w$ in G , for some $w \in T^*$. Moreover, every symbol used in this derivation is also generating. Thus $X \stackrel{*}{\Rightarrow} w$ in G_2 also. But this is not enough!

Since X was not eliminated in step 2, there are α and β, such that $S \stackrel{*}{\Rightarrow} \alpha X \beta$ in G_2. Furthermore, every symbol used in this derivation is also reachable, so $S \stackrel{*}{\Rightarrow} \alpha X \beta$ in G_1.

Now every symbol in $\alpha X \beta$ is reachable and in $V_2 \cup T_2 \supseteq V_1 \cup T_1$, so each of them is generating in G_2.

The terminal derivation $\alpha X \beta \stackrel{*}{\Rightarrow} xwy$ in G_2 involves only symbols that are reachable from S, because they are reached from symbols in $\alpha X \beta$. Thus the terminal derivation is also a derviation in G_1, i.e.,

$$S \stackrel{*}{\Rightarrow} \alpha X \beta \stackrel{*}{\Rightarrow} xwy$$

in G_1.

We then show that $L(G_1) = L(G)$.

Since $P_1 \subseteq P$, we have $L(G_1) \subseteq L(G)$.

Then, let $w \in L(G)$. Thus $S \overset{*}{\underset{G}{\Rightarrow}} w$. Each symbol is this derivation is evidently both reachable and generating, so this is also a derivation of G_1.

Thus $w \in L(G_1)$.

We have to give algorithms to compute the generating and reachable symbols of $G = (V, T, P, S)$.

The generating symbols $g(G)$ are computed by the following closure algorithm:

Basis: $g(G) == T$

Induction: If $\alpha \in g(G)^*$ and $X \to \alpha \in P$, then $g(G) == g(G) \cup \{X\}$.

Example: Let G be $S \to AB|a,\ A \to b$

Then first $g(G) == \{a, b\}$.

Since $S \to a$ we put S in $g(G)$, and because $A \to b$ we add A also, and that's it.

Theorem 7.4: At saturation, $g(G)$ contains all and only the generating symbols of G.

Proof:

We'll show in class by an induction on the stage in which a symbol X is added to $g(G)$ that X is indeed generating.

Then, suppose that X is generating. Thus $X \overset{*}{\underset{G}{\Rightarrow}} w$, for some $w \in T^*$. We prove by induction on this derivation that $X \in g(G)$.

Basis: Zero Steps. Then X is added in the basis of the closure algo.

Induction: The derivation takes $n > 0$ steps. Let the first production used be $X \to \alpha$. Then

$$X \Rightarrow \alpha \overset{*}{\Rightarrow} w$$

and $\alpha \overset{*}{\Rightarrow} w$ in less than n steps and by the IH $\alpha \in g(G)^*$. From the inductive part of the algo it follows that $X \in g(G)$.

The set of reachable symbols $r(G)$ of $G = (V, T, P, S)$ is computed by the following closure algorithm:

Basis: $r(G) == \{S\}$.

Induction: If variable $A \in r(G)$ and $A \rightarrow \alpha \in P$ then add all symbols in α to $r(G)$

Example: Let G be $S \rightarrow AB|a, \ A \rightarrow b$

Then first $r(G) == \{S\}$.

Based on the first production we add $\{A, B, a\}$ to $r(G)$.

Based on the second production we add $\{b\}$ to $r(G)$ and that's it.

Theorem 7.6: At saturation, $r(G)$ contains all and only the reachable symbols of G.

Proof: Homework.

Eliminating ϵ-Productions

We shall prove that if L is CF, then $L \setminus \{\epsilon\}$ has a grammar without ϵ-productions.

Variable A is said to be *nullable* if $A \stackrel{*}{\Rightarrow} \epsilon$.

Let A be nullable. We'll then replace a rule like

$$A \rightarrow BAD$$

with

$$A \rightarrow BAD, \; A \rightarrow BD$$

and delete any rules with body ϵ.

We'll compute $n(G)$, the set of nullable symbols of a grammar $G = (V, T, P, S)$ as follows:

Basis: $n(G) == \{A : A \rightarrow \epsilon \in P\}$

Induction: If $\{C_1 C_2 \cdots C_k\} \subseteq n(G)$ and $A \rightarrow C_1 C_2 \cdots C_k \in P$, then $n(G) == n(G) \cup \{A\}$.

Theorem 7.7: At saturation, $n(G)$ contains all and only the nullable symbols of G.

Proof: Easy induction in both directions.

Once we know the nullable symbols, we can transform G into G_1 as follows:

- For each $A \to X_1 X_2 \cdots X_k \in P$ with $m \leq k$ nullable symbols, replace it by 2^m rules, one with each sublist of the nullable symbols absent.

Exeption: If $m = k$ we don't delete all m nullable symbols.

- Delete all rules of the form $A \to \epsilon$.

Example: Let G be

$$S \rightarrow AB, \ A \rightarrow aAA|\epsilon, \ B \rightarrow bBB|\epsilon$$

Now $n(G) = \{A, B, S\}$. The first rule will become

$$S \rightarrow AB|A|B$$

the second

$$A \rightarrow aAA|aA|aA|a$$

the third

$$B \rightarrow bBB|bB|bB|b$$

We then delete rules with ϵ-bodies, and end up with grammar G_1 :

$$S \rightarrow AB|A|B, \ A \rightarrow aAA|aA|a, \ B \rightarrow bBB|bB|b$$

Theorem 7.9: $L(G_1) = L(G) \setminus \{\epsilon\}$.

Proof: We'll prove the stronger statement:

($\sharp$) $A \stackrel{*}{\Rightarrow} w$ in G_1 if and only if $w \neq \epsilon$ and $A \stackrel{*}{\Rightarrow} w$ in G.

$\subseteq$*-direction:* Suppose $A \stackrel{*}{\Rightarrow} w$ in G_1. Then clearly $w \neq \epsilon$ (Why?). We'll show by an induction on the length of the derivation that $A \stackrel{*}{\Rightarrow} w$ in G also.

Basis: One step. Then there exists $A \rightarrow w$ in G_1. From the construction of G_1 it follows that there exists $A \rightarrow \alpha$ in G, where α is w plus some nullable variables interspersed. Then

$$A \Rightarrow \alpha \stackrel{*}{\Rightarrow} w$$

in G.

Induction: Derivation takes $n > 1$ steps. Then

$$A \Rightarrow X_1 X_2 \cdots X_k \stackrel{*}{\Rightarrow} w \text{ in } G_1$$

and the first derivation is based on a production

$$A \rightarrow Y_1 Y_2 \cdots Y_m \quad \text{in G}$$

where $m \geq k$, some Y_i's are X_j's and the other are nullable symbols of G.

Furthermore, $w = w_1 w_2 \cdots w_k$, and $X_i \stackrel{*}{\Rightarrow} w_i$ in G_1 in less than n steps. By the IH we have $X_i \stackrel{*}{\Rightarrow} w_i$ in G. Now we get

$$A \underset{G}{\Rightarrow} Y_1 Y_2 \cdots Y_m \underset{G}{\stackrel{*}{\Rightarrow}} X_1 X_2 \cdots X_k \underset{G}{\stackrel{*}{\Rightarrow}} w_1 w_2 \cdots w_k = w$$

$\supseteq$*-direction:* Let $A \overset{*}{\underset{G}{\Rightarrow}} w$, and $w \neq \epsilon$. We'll show by induction of the length of the derivation that $A \overset{*}{\Rightarrow} w$ in G_1.

Basis: Length is one. Then $A \rightarrow w$ is in G, and since $w \neq \epsilon$ the rule is in G_1 also.

Induction: Derivation takes $n > 1$ steps. Then it looks like

$$A \underset{G}{\Rightarrow} Y_1 Y_2 \cdots Y_m \overset{*}{\underset{G}{\Rightarrow}} w$$

Now $w = w_1 w_2 \cdots w_m$, and $Y_i \overset{*}{\underset{G}{\Rightarrow}} w_i$ in less than n steps.

Let $X_1 X_2 \cdots X_k$ be those Y_j's in order, such that $w_j \neq \epsilon$. Then $A \rightarrow X_1 X_2 \cdots X_k$ is a rule in G_1.

Now $X_1 X_2 \cdots X_k \overset{*}{\underset{G}{\Rightarrow}} w$ (Why?)

Each $X_j/Y_j \overset{*}{\underset{G}{\Rightarrow}} w_j$ in less than n steps, so by IH we have that if $w_j \neq \epsilon$ then $Y_j \overset{*}{\Rightarrow} w_j$ in G_1. Thus

$$A \Rightarrow X_1 X_2 \cdots X_k \overset{*}{\Rightarrow} w \text{ in } G_1$$

The claim of the theorem now follows from statement ($\sharp$) on slide 238 by choosing $A = S$.

Eliminating Unit Productions

$$A \rightarrow B$$

is a *unit* production, whenever A and B are variables.

Unit productions can be eliminated.

Let's look at grammar

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

It has unit productions $E \rightarrow T$, $T \rightarrow F$, and $F \rightarrow I$

We'll expand rule $E \rightarrow T$ and get rules

$$E \rightarrow F,\ E \rightarrow T * F$$

We then expand $E \rightarrow F$ and get

$$E \rightarrow I|(E)|T * F$$

Finally we expand $E \rightarrow I$ and get

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

The expansion method works as long as there are no cycles in the rules, as e.g. in

$$A \rightarrow B,\ B \rightarrow C,\ C \rightarrow A$$

The following method based on *unit pairs* will work for all grammars.

(A, B) is a *unit pair* if $A \stackrel{*}{\Rightarrow} B$ using unit productions only.

Note: In $A \to BC$, $C \to \epsilon$ we have $A \stackrel{*}{\Rightarrow} B$, but not using unit productions only.

To compute $u(G)$, the set of all unit pairs of $G = (V, T, P, S)$ we use the following closure algorithm

Basis: $u(G) == \{(A, A) : A \in V\}$

Induction: If $(A, B) \in u(G)$ and $B \to C \in P$ then add (A, C) to $u(G)$.

Theorem: At saturation, $u(G)$ contains all and only the unit pair of G.

Proof: Easy.

Given $G = (V, T, P, S)$ we can construct $G_1 = (V, T, P_1, S)$ that doesn't have unit productions, and such that $L(G_1) = L(G)$ by setting

$$P_1 = \{A \to \alpha : \alpha \notin V, B \to \alpha \in P, (A, B) \in u(G)\}$$

Example: For the grammar of slide 242 we get

Pair	Productions
(E, E)	$E \to E + T$
(E, T)	$E \to T * F$
(E, F)	$E \to (E)$
(E, I)	$E \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$
(T, T)	$T \to T * F$
(T, F)	$T \to (E)$
(T, I)	$T \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$
(F, F)	$F \to (E)$
(F, I)	$F \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$
(I, I)	$I \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

The resulting grammar is equivalent to the original one (proof omitted).

Summary

To "clean up" a grammar we can

1. Eliminate ϵ-productions

2. Eliminate unit productions

3. Eliminate useless symbols

in this order.

This cannot be done earlier due to the removal of ε–productions and unit productions.

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 $\{A, B, C\} \subseteq V$, 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 $A \rightarrow a$.

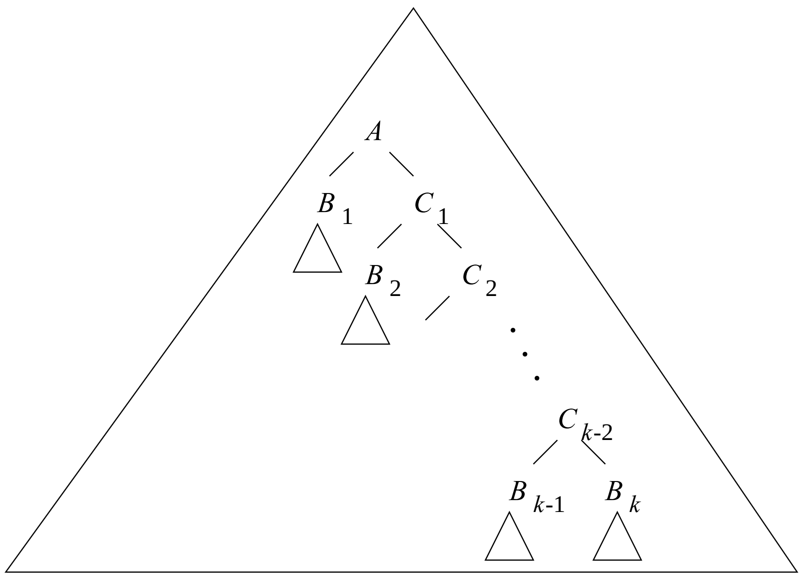
• For step 3, for each rule of the form

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

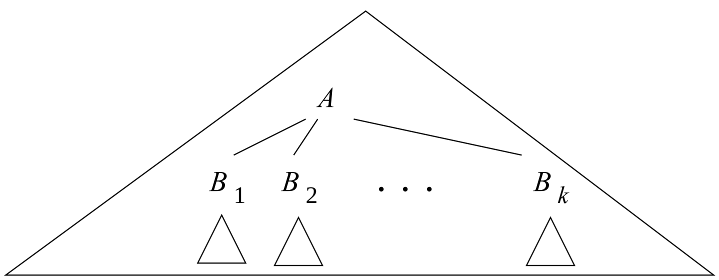
$k \geq 3$, introduce new variables $C_1, C_2, \ldots C_{k-2}$, and replace the rule with

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

Illustration of the effect of step 3



(a)



(b)

Example of CNF conversion

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

$$
\begin{aligned}
&E \rightarrow E + T \mid T * F \mid (E) \mid a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \\
&T \rightarrow T * F \mid (E) \mid a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \\
&F \rightarrow (E) \mid a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \\
&I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1
\end{aligned}
$$

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

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

For step 3, we replace

$$E \rightarrow EPT \text{ by } E \rightarrow EC_1, C_1 \rightarrow PT$$

$$E \rightarrow TMF, T \rightarrow TMF \text{ by}$$
$$E \rightarrow TC_2, T \rightarrow TC_2, C_2 \rightarrow MF$$

$$E \rightarrow LER, T \rightarrow LER, F \rightarrow LER \text{ by}$$
$$E \rightarrow LC_3, T \rightarrow LC_3, F \rightarrow LC_3, C_3 \rightarrow ER$$

The final CNF grammar is

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