Introduction to the Theory of Computation

XU Ming

School of Software Engineering, East China Normal University

May 30, 2024

OUTLINE

- Normal Forms for Context-Free Grammars
  - Eliminating Useless Symbols
  - Eliminating ϵ-Productions
  - Eliminating Unit Productions
  - Chomsky Normal Form

Simplification of CFG's

Simplifying CFG's makes it easier to claim that if a language is context-free, then it has a grammar of special 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 Chomsky Normal Form (CNF).

To get CNF of a CFG 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 $A \rightarrow \epsilon$.
3. eliminate unit productions, that is, productions of the form $A \rightarrow B$, where A and B are variables.

Eliminating Useless Symbols

Eliminating Useless Symbols

- A symbol X is useful for a grammar $G = (V, T, P, S)$, if there is a derivation
$$S \overset{*}{\underset{G}{\Longrightarrow}} \alpha X \beta \overset{*}{\underset{G}{\Longrightarrow}} w$$
for a terminal string w. X may be in either V or T. Symbols that are not useful are called useless.
- A symbol X is generating if $X \overset{*}{\underset{G}{\Longrightarrow}} w$ holds for some $w \in T^*$.
- A symbol X is reachable if $S \overset{*}{\underset{G}{\Longrightarrow}} \alpha X \beta$ holds for some $\alpha, \beta \in (V \cup T)^*$.

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

Example

Let G be

$$S \to AB \mid a, \quad A \to b.$$

S and A as well as a, b are generating, B is not.
If we eliminate B we have to eliminate $S \to AB$, leaving grammar $S \to a, A \to b$. Now only S and a is reachable. Eliminating A and b leaves us with $S \to a$ with language $\{a\}$.

On the other hand, if we first eliminate non-reachable symbols, we find that all symbols are reachable.

From

$$S \to AB \mid a, \quad A \to b$$

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

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

that still contains useless symbols.

Theorem 7.1

If $G = (V, T, P, S)$ be a CFG s.t. $L(G) \neq \emptyset$, let $G_1 = (V_1, T_1, P_1, S)$ be the grammar obtained by

1. *eliminating all nongenerating symbols and the productions they appear in, which results in the middle grammar $G_2 = (V_2, T_2, P_2, S)$;*
2. *eliminating from G_2 all nonreachable symbols and the productions they appear in.*

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

Computing the Generating and Reachable Symbols

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

The set of generating symbols $g(G)$ is computed by the following closure:

Basis step: $g(G) \leftarrow T$.

Inductive step: If there is an $X \to \alpha \in P$ and every symbol of α is in $g(G)$, then $g(G) \leftarrow g(G) \cup \{X\}$. (Note that α can be ϵ.)

Example

Let G be

$$S \to AB \mid a, \quad A \to b.$$

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

Theorem 7.2

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

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

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

Inductive step: If $A \in r(G)$ and $A \to \alpha \in P$, add all symbols in α to $r(G)$.

Example

Let G be $S \to AB \,|\, a$, $A \to b$. First, $r(G) \leftarrow \{S\}$. Based on the first and second groups of productions, we add $\{A, B, a\}$ and $\{b\}$ to $r(G)$ in turn, and that's it.

Theorem 7.3

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

Eliminating ϵ-Productions

Eliminating ϵ-Productions

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

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

Let A be nullable. We'll replace a rule like $A \to BAD$ with two productions $A \to BAD$, $A \to BD$ 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 step: $n(G) \leftarrow \{A \mid A \rightarrow \epsilon \in P\}$.

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

Theorem 7.4

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 X_i's, replace it by 2^m rules, one with each sublist of the nullable symbols absent.
  Exception: If $m = k$ we don't delete all m nullable symbols.
- Delete all rules of the form $A \to \epsilon$.

Example

Eliminating ϵ-productions in G:

$$S \to AB, \quad A \to aAA \mid \epsilon, \quad B \to bBB \mid \epsilon$$

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

$$S \to AB \mid A \mid B$$

the second and third, respectively

$$A \to aAA \mid aA \mid aA \mid a, \quad B \to bBB \mid bB \mid bB \mid b$$

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

$$S \to AB \mid A \mid B, \quad A \to aAA \mid aA \mid a, \quad B \to bBB \mid bB \mid b$$

Theorem 7.5

If the grammar G_1 is constructed from G by the above construction for eliminating ϵ-productions, then $L(G_1) = L(G) \setminus \{\epsilon\}$.

Proof It suffices to prove the stronger statement:

$$A \overset{*}{\Rightarrow} w \text{ in } G_1 \quad \text{iff} \quad w \neq \epsilon \text{ and } A \overset{*}{\Rightarrow} w \text{ in } G$$

The theorem follows by choosing $A = S$. The remaining is skipping.

Eliminating Unit Productions

Eliminating Unit Production

- A unit production is a production of the form $A \to B$, where both A and B are variables.

In CFG's, unit production can be eliminated.

Let's look at the grammar

1. $I \to a \,|\, b \,|\, Ia \,|\, Ib \,|\, I0 \,|\, I1,$
2. $F \to I \,|\, (E),$
3. $T \to F \,|\, T \times F,$
4. $E \to T \,|\, E + T$

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

We expand rule $E \to T$ and get rules $E \to F \mid T \times F \mid E + T$.

We then expand $E \to F$ and get $E \to I \mid (E) \mid T \times F \mid E + T$.

Finally we expand $E \to I$ and get

$$E \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \mid (E) \mid T \times F \mid E + T$$

The expansion method works as long as there are no cycles in the rules, as e.g. in $A \to B$, $B \to C$ and $C \to A$. The following method based on unit pairs will work for all grammars.

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

In $A \to BC$, $C \to \epsilon$ we have $A \overset{*}{\Rightarrow} B$, 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:

Basis step: $u(G) \leftarrow \{(A, A) | A \in V\}$.

Inductive step: If $(A, B) \in u(G)$ and $B \to C \in P$, where C is a variable, then add (A, C) to $u(G)$.

Theorem 7.6

At saturation, u(G) contains all and only the unit pairs of G.

To eliminate unit productions, we proceed as follows. Given a CFG $G = (V, T, P, S)$, construct CFG $G_1 = (V, T, P_1, S)$:

1. Find all the unit pairs of G.
2. For each unit pair (A, B), add to P_1 all the productions $A \to \alpha$, where $B \to \alpha$ is a nonunit production in P. (Note that $A = B$ is possible.)

Example

Eliminating unit productions from the following grammar

$$
\begin{aligned}
&1.\ I \to a \,|\, b \,|\, Ia \,|\, Ib \,|\, I0 \,|\, I1,\\
&2.\ F \to I \,|\, (E),\\
&3.\ T \to F \,|\, T \times F,\\
&4.\ E \to T \,|\, E + T
\end{aligned}
$$

By the construction as above, we get

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

Theorem 7.7

If the grammar G_1 is constructed from G by the algorithm described above for eliminating unit productions, then $L(G_1) = L(G)$.

We now summarize the various simplifications described so far.

To “clean up” a context-free grammar, we can

1. Eliminating ϵ-productions.
2. Eliminating unit productions.
3. Eliminating useless symbols.

This is a safe order.

We have presented various simplifications for a CFG, and achieved at

Theorem 7.8

If G is a CFG generating a language that contains at least one string other than ϵ, then there is another CFG G' such that $L(G') = L(G) \setminus \{\epsilon\}$, and G' has no ϵ-productions, unit productions, or useless symbols.

To convert G into G', some care must be taken in the order of application of the constructions. A safe order is eliminating ϵ-productions, then eliminating unit productions, and finally eliminating useless symbols.

An Example of Simplifying CFG

Let PDA $P = (\{p, q\}, \{0, 1\}, \{X, Z_0\}, \delta, q, Z_0)$, where δ is given by

1. $\delta(q, 1, Z_0) = \{(q, XZ_0)\}$
2. $\delta(q, 1, X) = \{(q, XX)\}$
3. $\delta(q, 0, X) = \{(p, X)\}$
4. $\delta(q, \epsilon, Z_0) = \{(q, \epsilon)\}$
5. $\delta(p, 1, X) = \{(p, \epsilon)\}$
6. $\delta(p, 0, Z_0) = \{(q, Z_0)\}$

We have constructed the equivalent CFG G_P such that $L(G_P) = N(P)$. Now we want to "clean up" G_P.

We get $G_P = (V, \{0, 1\}, R, S)$ where

$$V = \{S, [pXp], [pXq], [pZ_0p], [pZ_0q], [qXp], [qXq], [qZ_0p], [qZ_0q]\}$$

and the productions in R are

$$
\begin{aligned}
&S \to [qZ_0q] \mid [qZ_0p]\\
&[qZ_0q] \to \epsilon\\
&[pXp] \to 1\\
&[qZ_0q] \to 1[qXq][qZ0q] \mid 1[qXp][pZ_0q]\\
&[qZ_0p] \to 1[qXq][qZ_0p] \mid 1[qXp][pZ_0p]\\
&[qXq] \to 1[qXq][qXq] \mid 1[qXp][pXq]\\
&[qXp] \to 1[qXq][qXp] \mid 1[qXp][pXp]\\
&[qXq] \to 0[pXq]\\
&[qXp] \to 0[pXp]\\
&[pZ_0q] \to 0[qZ0q]\\
&[pZ_0p] \to 0[qZ_0p]
\end{aligned}
$$

We may, for convenience, replace the triple $[pXp]$, $[pXq]$, $[pZ_0p]$, $[pZ_0q]$, $[qXp]$, $[qXq]$, $[qZ_0p]$, $[qZ_0q]$ by some simple symbols, say A, B, C, D, E, F, G, H. If we do, then the complete grammar consists of the productions:

$$
\begin{aligned}
S &\to H \,|\, G \\
H &\to 1FH \,|\, 1ED \\
G &\to 1FG \,|\, 1EC \\
F &\to 1FF \,|\, 1EB \,|\, 0B \\
E &\to 1FE \,|\, 1EA \,|\, 0A \\
H &\to \epsilon \\
A &\to 1 \\
D &\to 0H \\
C &\to 0G
\end{aligned}
$$

1. Eliminating ϵ-productions
   We compute $n(G_P) = \{H\}$, and eliminate nullable symbol H:

$$
\begin{aligned}
S &\to H \mid G \\
H &\to 1F \mid 1FH \mid 1ED \\
G &\to 1FG \mid 1EC \\
F &\to 1FF \mid 1EB \mid 0B \\
E &\to 1FE \mid 1EA \mid 0A \\
A &\to 1 \\
D &\to 0 \mid 0H \\
C &\to 0G
\end{aligned}
$$

2 Eliminating unit productions
We compute $u(G_P) = \{(A, A), (B, B), (C, C), (D, D), (E, E), (F, F), (G, G), (H, H), (S, S), (S, G), (S, H)\}$, and eliminate unit productions $S \rightarrow H$ and $S \rightarrow G$:

$$
\begin{aligned}
S &\rightarrow 1F \,|\, 1FH \,|\, 1ED \,|\, 1FG \,|\, 1EC \\
H &\rightarrow 1F \,|\, 1FH \,|\, 1ED \\
G &\rightarrow 1FG \,|\, 1EC, \\
F &\rightarrow 1FF \,|\, 1EB \,|\, 0B \\
E &\rightarrow 1FE \,|\, 1EA \,|\, 0A \\
A &\rightarrow 1 \\
D &\rightarrow 0 \,|\, 0H \\
C &\rightarrow 0G
\end{aligned}
$$

3.1 Eliminating useless symbols—nongenerating symbols
We compute $g(G_P) = \{0, 1, A, D, E, H, S\}$, and eliminate nongenerating symbols G, F, C, B:

$$\begin{aligned}
S &\rightarrow 1ED \\
H &\rightarrow 1ED \\
E &\rightarrow 1EA \mid 0A \\
A &\rightarrow 1 \\
D &\rightarrow 0 \mid 0H
\end{aligned}$$

3.2 Eliminating useless symbols—nonreachable symbols
We compute $r(G_P) = \{S, 1, E, D, A, 0, H\}$, all symbols are reachable. So:

$$S \rightarrow 1ED, \quad H \rightarrow 1ED, \quad E \rightarrow 1EA \,|\, 0A, \quad A \rightarrow 1, \quad D \rightarrow 0 \,|\, 0H$$

In fact, if we notice that there is only a single production for variables A and H, respectively, we may write the complete grammar G'_P as

$$S \rightarrow 1ED, \quad E \rightarrow 1E1 \,|\, 01, \quad D \rightarrow 0 \,|\, 01ED$$

Notice that $N(P) = L(G'_P) \cup \{\epsilon\}$.

Chomsky Normal Form

Chomsky Normal Form

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
- $A \to a$ where $A \in V$ and $a \in T$.

Such a grammar is said to be in Chomsky Normal Form or CNF. One of the uses of CNF is to turn parse trees into binary trees.

To achieve CNF, start with any grammar for CFL, and

1. "clean up" the grammar.
2. arrange that all bodies of length 2 or more consist only of 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 some bodies of length ≥ 2, create a new variable, say A, and replace a by A in those bodies. Then add a new rule $A \to a$.

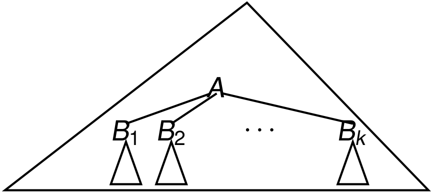
For step 3, for each rule of the form

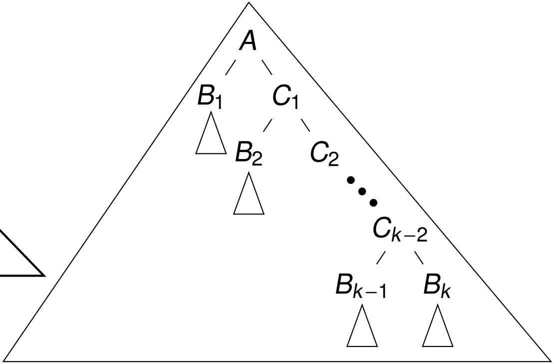
$$A \rightarrow B_1 B_2 \cdots B_k,$$

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

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

Illustration of the effect of step 3





Example of CNF Conversion

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

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

For step 2, we need the rules

$$A \to a,\ B \to b,\ Z \to 0,\ O \to 1,\ P \to +,\ M \to \times,\ L \to (,\ R \to)$$

and by replacing we get the grammar

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

For step 3, we introduce C_1 for EPT with $C_1 \to PT$, C_2 for TMT with $C_2 \to MF$, and C_3 for LER with $C_3 \to ER$. Finally we get the CNF grammar:

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

Homework

Exercises 7.1.3, 7.1.9(a)