Formal Languages
Grammar

Ryan Stansifer

Department of Computer Sciences
Florida Institute of Technology
Melbourne, Florida USA 32901

http://www.cs.fit.edu/~ryan/

March 15, 2018
A **formal language** is a set of strings over an alphabet.
String Conventions

We consider ordered sequences (lists) to be polymorphic data structures. Often, but not always, we are interested in ordered sequences of symbols (letters) from some alphabet (character set). We call these kinds of ordered sequences *strings*. (We call other kinds of ordered sequences strings, too.)

In Haskell, this datatype is built-in, but also easily definable

```haskell
data List a = Null | Cons a (List a)
data Symbol = A | B | C
type String = List Symbol
```
There is a unique list of length zero called null and denoted $\epsilon$. Null, sometimes called the the empty string, is the unique string of length zero no matter which alphabet $\Sigma$ we are talking about.

We make no notational distinction between a symbol $a \in \Sigma$ and the string of length one containing $a$. Thus we regard $\Sigma$ as a subset of $\Sigma^*$. Indeed, we make no notational distinction between a symbol $a$ the corresponding regular expression $a$. (Some authors use a bold weight font for regular expressions, e.g., $(ab + c)^*$. Some authors enclose syntactic terms in double brackets $[(ab + c)^*]$. )
String Conventions

Ø, {ε}, and ε are three different things.

Ø  the set with no elements
{ε}  the set with one element (the null string)
ε  the ordered sequence (string) of length zero

Finally the length of a string $x \in \Sigma^*$ is denoted $|x|$. Also the cardinality of a set $S$ is denoted $|S|$ — the number of members of the set.
Summary of Four Computational Models

1. An automaton $M$ accepts a formal language $L(M)$.
2. An expression $x$ denotes a formal language $L[x]$.
3. A grammar $G$ generates a formal language $L(G)$.
4. A Post system $P$ derives a formal language $L(P)$. 
Automata compute

Expressions denote

Trees demonstrate

Grammars construct
Grammar

A grammar is another formalism for defining and generating a formal language (a set of strings over an alphabet).

In the context of grammars it is traditional to call the alphabet a set of terminal symbols. Ordered sequences of terminal symbols are sometimes called sentences in this context, instead of strings.

Context-free grammars play an important role in defining programming languages and in the construction of compilers for programming languages.
Definition of Grammar

A grammar is a 4-tuple \( \langle T, V, P, S \rangle \):

- \( T \) is the finite set of terminal symbols;

- \( V \) is the finite set of nonterminal symbols, also called variables or syntactic categories, \( T \cap V = \emptyset \);

- \( S \in V \), is the start symbol;

- \( P \) is the finite set of productions. A production has the form \( \alpha \to \beta \) where \( \alpha \) and \( \beta \) are ordered sequences (strings) of terminals and nonterminals symbols. (We write \((T \cup V)^*\) for the set of ordered sequences of terminals and nonterminals symbols.) The LHS \( \alpha \) of a production can't be the empty sequence, but \( \beta \) might be.
Definition of Grammar

A grammar is a 4-tuple \( \langle T, V, P, S \rangle \):

- \( T \) is the finite set of terminal symbols;
- \( V \) is the finite set of nonterminal symbols, also called variables or syntactic categories, \( T \cap V = \emptyset \);
A grammar is a 4-tuple $\langle T, V, P, S \rangle$:

- $T$ is the finite set of terminal symbols;
- $V$ is the finite set of nonterminal symbols, also called variables or syntactic categories, $T \cap V = \emptyset$;
- $S \in V$, is the start symbol;
A grammar is a 4-tuple \( \langle T, V, P, S \rangle \):

- \( T \) is the finite set of terminal symbols;
- \( V \) is the finite set of nonterminal symbols, also called variables or syntactic categories, \( T \cap V = \emptyset \);
- \( S \in V \), is the start symbol;
- \( P \) is the finite set of productions.

A production has the form \( \alpha \rightarrow \beta \) where \( \alpha \) and \( \beta \) are ordered sequences (strings) of terminals and nonterminals symbols. (We write \( (T \cup V)^* \) for the set of ordered sequences of terminals and nonterminals symbols.) The LHS \( \alpha \) of a production can't be the empty sequence, but \( \beta \) might be.
Example

We are primarily interested in context-free grammars in which the LHS $\alpha$ is a single nonterminal (which we formally introduce later).

The following five productions are for a grammar with $T = \{0, 1\}$, $V = \{S\}$, and start symbol $S$.

1. $S \rightarrow \epsilon$
2. $S \rightarrow 0$
3. $S \rightarrow 1$
4. $S \rightarrow 0\ S\ 0$
5. $S \rightarrow 1\ S\ 1$
Common Notational Conventions

To spare tedious repetition it is convenient to establish some notational conventions.

1. Lower-case letters near the beginning of the alphabet, \( a, b, \) and so on, are terminal symbols. We shall also assume that non-letter symbols, like digits, \( +, \) are terminal symbols.

2. Upper-case letters near the beginning of the alphabet, \( A, B, \) and so on, are nonterminal symbols. Often the start symbol of a grammar is assumed to be named \( S, \) or sometimes it is the nonterminal on the left-hand side of the first production.

3. Lower-case letters near the end of the alphabet, such as \( w \) or \( z, \) are strings of terminals.

4. Upper-case letters near the end of the alphabet, such as \( X \) or \( Y, \) are a single symbol, either a terminal or a nonterminal.

5. Lower-case Greek letters, such as \( \alpha \) and \( \beta, \) are sequences (possibly empty) of terminal and nonterminal symbols.
Common Notational Conventions (Recap)

To spare tedious repetition it is convenient to establish some notational conventions.

1. $a, b, c, \ldots$ are terminals.
2. $A, B, C, \ldots$ are nonterminals.
3. $X, Y, Z$ are terminal or nonterminal symbols.
4. $w, x, y, z$ are strings of terminals only.
5. $\alpha, \beta, \gamma, \ldots$ are strings of terminals or nonterminals.
Compact Notation

If the RHS of a production is a sequence with no symbols in it, we use $\epsilon$ to communicate that fact clearly. So, for example, $A \rightarrow \epsilon$ is a production.

Definition. A production in a grammar in which the RHS of a production is a sequence of length zero is called an $\epsilon$-production.

The productions with the same LHS, $A \rightarrow \alpha_1$, $A \rightarrow \alpha_2$, \ldots, $A \rightarrow \alpha_n$ can be written using the notation $A \rightarrow \alpha_1 \mid \alpha_2 \mid \cdots \mid \alpha_n$.
It is convenient to think of a production as “belonging” to the variable \([\text{nonterminal}]\) of its head \([\text{LHS}]\). We shall often use remarks like “the productions for \(A\)” or “\(A\)-productions” to refer to the production whose head is \([\text{the}]\) variable \(A\). We may write the productions for a grammar by listing each variable once, and then listing all the bodies of the productions for that variable, separated by vertical bars.

HMU 3rd, §5.1, page 175.
A grammar $G = \langle T, V, P, S \rangle$ gives rise naturally to a method of constructing strings of terminal and nonterminal symbols by application of the productions.

**Definition.** If $\alpha, \beta \in (T \cup V)^*$, we say that $\alpha$ derives $\beta$ in one step and write

$$\alpha \xrightarrow{1} G \beta$$

if $\beta$ can be obtained from $\alpha$ by replacing some occurrence of the substring $\delta$ in $\alpha$ with $\gamma$, where $\delta \rightarrow \gamma$ is a production of $G$. In other words, $\alpha = \alpha_1 \delta \alpha_2$ and $\beta = \alpha_1 \gamma \alpha_2$ for some $\alpha_1 \alpha_2 \in (T \cup V)^*$. 
We usually omit the grammar $G$ writing

$$\alpha \xrightarrow{1} \beta$$

and leave it to the reader to figure out which grammar is meant.

We often omit the 1, writing

$$\alpha \Rightarrow \beta$$

except to emphasize the distinction with other derivability relations.
Perhaps we can recast the previous definition slightly more perspicuously as follows:

**Definition.** If $\delta \to \gamma$ is a production of $G$, then $\alpha_1 \delta \alpha_2$ derives $\alpha_1 \gamma \alpha_2$ *in one step* and write

$$\alpha_1 \delta \alpha_2 \Rightarrow_G \alpha_1 \gamma \alpha_2$$

for all $\alpha_1, \alpha_2 \in (T \cup V)^*$. 

Definition. Let $\Rightarrow^*_G$ be the reflexive, transitive closure of the $\Rightarrow_G$ relation. That is:

$$\alpha \Rightarrow^*_G \alpha \quad \text{for any } \alpha$$

$$\alpha \Rightarrow^*_G \beta \quad \text{if } \alpha \Rightarrow^*_G \gamma \text{ and } \gamma \Rightarrow^*_G \beta$$

This relation is sometimes called the “derives in zero or more steps” relation.
A string in \((T \cup V)^*\) that is derivable from the start symbol \(S\) is called a \textit{sentential form}. A sentential form is called a \textit{sentence}, if it consists only of terminal symbols.

\textit{Definition}. The \textit{language generated by} \(G\), denoted \(L(G)\), is the set of all sentences:

\[
L(G) = \{ x \in T^* \mid S \Rightarrow_G^* x \}
\]
Outline

Different Kinds of Grammars

Context-Free Grammars
  Leftmost, Rightmost Derivations
  Ambiguity

Chomsky and Greibach Normal Forms

Cleaning Up CFG’s

Brute Force Membership Test

CYK Algorithm
Grammars

- **context-free**
  - **linear**
    - **regular**
      - left linear
        - $S \rightarrow Sa \mid \epsilon$
      - right linear
        - $S \rightarrow aS \mid \epsilon$
    - $S \rightarrow aSb \mid \epsilon$
  - $S \rightarrow SS$
  - $S \rightarrow aSb$
  - $S \rightarrow bSa$
  - $S \rightarrow \epsilon$

Grammars (not languages)
Different Kinds of Grammars

Unrestricted grammar

- Context-sensitive grammar
- Context-free grammar
  - Grammar in Greibach normal form
    - simple-grammar, s-grammar
  - Grammar in Chomsky normal form
  - Linear grammar
    - Regular grammar
      - right-linear grammar.
      - left-linear grammar.
- Compiler theory: LL(k), LR(k) etc.
Restrictions on Grammars

Grammar (Linz 6th, §1.2, definition 1.1, page 21).
  ▶ Context-sensitive grammar (Linz 6th, §11.4, definition 11.4, page 300).
  ▶ Context-free grammar (Linz 6th, §5.1, definition 5.1, page 130).
      ▶ simple-grammar, s-grammar (Linz 6th, §5.3, definition 5.4, page 144).
  ▶ Linear grammar (Linz 6th, §3.3, page 93).
    ▶ Regular grammar (Linz 6th, §3.3, definition 3.3, page 92).
      • right-linear grammar.
      • left-linear grammar.

  ▶ Compiler theory: LL(k) (Linz 6th, §7.4, definition 7.5, page 210); LR(k) [Not defined in Linz 6th.]
Restrictions on Grammars (HMU)

- Context-sensitive grammar [Not defined in HMU].
    - simple-grammar, s-grammar [Not defined in HMU].
  - Chomsky normal form (HMU 3rd, §7.1.5, page 272).
- Linear grammar [Not defined in HMU].
  - Regular grammar [Not defined in HMU].
    - right-linear grammar [HMU 3rd, §5.1.7, exercise 5.1.4, page 182].
    - left-linear grammar.
- Compiler theory: LL(k), LR(k) etc. [Not defined in HMU].
Restrictions on Grammars

- Context-sensitive grammar. Each production $\alpha \rightarrow \beta$ restricted to $|\alpha| \leq |\beta|$
- Context-free grammar. Each production $A \rightarrow \beta$ where $A \in N$
  - Grammar in Greibach normal form. $A \rightarrow a\gamma$ where $a \in T$ and $\gamma \in V^*$
    - simple-grammar or s-grammar. Any pair $\langle A, a \rangle$ occurs at most once in the productions
  - Grammar in Chomsky normal form. $A \rightarrow BC$ or $A \rightarrow a$
  - Linear grammar. RHS $\beta$ contains at most one nonterminal
    - Regular grammar.
      - right-linear grammar. The nonterminal (if any) occurs to the right of (or after) any terminals
      - left-linear grammar. The nonterminal (if any) occurs to the left of (or before) any terminals
The s-languages are those languages recognized by a particular restricted form of deterministic pushdown automaton, called an s-machine. They are uniquely characterized by that subset of the standard-form grammars in which each rule has the form $Z \rightarrow aY_1 \ldots Y_n$, $n \geq 0$, and for which the pairs $(Z, a)$ are distinct among the rules. It is shown that the s-languages have the prefix property, and that they include the regular sets with end-markers. Finally, their closure properties and decision problems are examined, and it is found that their equivalence problem is solvable.

Korenja, Hopcroft, Simple deterministic languages, 1968.
right-linear $A \rightarrow xB$ or $A \rightarrow x$ $A, B \in V$, $x \in T^*$
strongly right-linear $A \rightarrow aB$ or $A \rightarrow \epsilon$ $A, B \in V$, $a \in T$
left-linear $A \rightarrow Bx$ or $A \rightarrow x$ $A, B \in V$, $x \in T^*$
strongly left-linear $A \rightarrow Ba$ or $A \rightarrow \epsilon$ $A, B \in V$, $a \in T$
Outline

Different Kinds of Grammars

Context-Free Grammars
  Leftmost, Rightmost Derivations
  Ambiguity

Chomsky and Greibach Normal Forms

Cleaning Up CFG’s

Brute Force Membership Test

CYK Algorithm
Context-Free Grammars (Overview)

- Definitions of a grammar and, specifically, a context-free grammar (Linz 6th, definition 5.1, page 130; HMU §5.1.2).
- Definition of a context-free language (Linz 6th, definition 5.1, page 130; HMU §5.1.5).
- Normal Forms
Context-Free Languages (Overview)

- Pushdown automata (Linz 6th, chapter 7, page 181; HUM 3rd, chapter 6, page 225).
- Decision properties of CFL (Linz 6th, §8.2, page 227; HMU 3rd, §7.4.2, page 301).
Context-Free Grammars

*Definition.* A *context-free grammar* is a grammar, if all productions are of the form \( A \rightarrow \alpha \) where \( A \in V \) and (as always) \( \alpha \in (T \cup V)^* \).

It is called context-free because there is no context surrounding the LHS nonterminal.
Definition. A subset $L \subseteq T^*$ is a context-free language (CFL) if $L = L(G)$ for some context-free grammar $G$. 
regular languages
context-free languages
context-sensitive languages
recursively enumerable languages
Context-Free Grammars

Context-free grammars are so common and useful that when speaking of grammars one often assumes that context-free grammars are meant as opposed to unrestricted grammars.

The definitions of derivations and sentential forms, and so on, are, of course, the same for all grammars. However, they bear repeating in the special case of context-free grammars.
Derivation (CFG)

A context-free grammar \( G = \langle T, V, S, P \rangle \) gives rise naturally to a method of constructing strings in \( T^* \) by applying the productions. If \( \alpha, \beta \in (T \cup V)^* \), we say that \( \alpha \) derives \( \beta \) in one step and write

\[
\alpha \xrightarrow{1}_G \beta
\]

if \( \beta \) can be obtained from \( \alpha \) by replacing some occurrence of a nonterminal \( A \) in the string \( \alpha \) with \( \gamma \), where \( A \to \gamma \) is a production of \( G \). In other words, \( \alpha = \alpha_1 A \alpha_2 \) and \( \beta = \alpha_1 \gamma \alpha_2 \) for some \( \alpha_1 \alpha_2 \in (T \cup V)^* \).

We usually omit the grammar \( G \) writing

\[
\alpha \xrightarrow{1} \beta
\]

and leave it to the reader to figure out which grammar is meant.
Definition. Let $\Rightarrow^*_G$ be the reflexive, transitive closure of the $\Rightarrow_G$ relation. That is:

- $\alpha \Rightarrow^* \alpha$ for any $\alpha$
- $\alpha \Rightarrow^* \beta$ if $\alpha \Rightarrow^* \gamma$ and $\gamma \Rightarrow_1 \beta$

This relation is sometimes called the “derives in zero or more steps” or simply the “derives” relation.
Consider the context-free grammar $G = \langle \{S\}, \{a, b\}, S, P \rangle$ with productions

$$
S \rightarrow aSa \\
S \rightarrow bSb \\
S \rightarrow \epsilon
$$

Here are some derivations for $G$:

$$
S \xrightarrow{1} G \quad aSa \xrightarrow{1} G \quad aaSa \xrightarrow{1} G \quad aabSbaa \xrightarrow{1} G \quad aabbaa
$$

$$
S \xrightarrow{1} G \quad bSb \xrightarrow{1} G \quad baSab \xrightarrow{1} G \quad babSbab \xrightarrow{1} G \quad babaSabab \xrightarrow{1} G \quad babaabab
$$

Since $S \xrightarrow{*} aabbaa$ and $S \xrightarrow{*} babaabab$, we have $aabbaa \in L(G)$ and $babaabab \in L(G)$. It is clear that this is the language of even length palindromes $L(G) = \{ww^R \mid w \in \{a, b\}^*\}$. Observe that this language was shown earlier not to be regular.
Derivations are inductively defined sets and we prove properties of derivations and hence about the languages they construct.

Next we give an example inductive proof about the language constructed by a CFG. We will need variations on the following notation.

For all $X \in T \cup V$ we write either $n_X(\alpha)$ or $\#_X(\alpha)$ for the number of occurrences of the symbol $X$ in the sentential form $\alpha \in (T \cup V)^*$. 

Notation
Consider the following grammar $G$:

1. $B \rightarrow \epsilon$
2. $B \rightarrow (RB$
3. $R \rightarrow )$
4. $R \rightarrow (RR$

$L(G) = \{w \in T^* \mid B \Rightarrow^* w\}$
Example Inductive Proof of CFG

(For the purposes of this example, it is slightly less awkward to write \( \#_L[\alpha] \) for \( \#([\alpha]). \)

Definition. Let \( \#_L[\alpha] \) be the number of left parentheses in \( \alpha \). And, let \( \#_R[\alpha] \) be the number of occurrences of the nonterminal \( R \). plus the number of right parentheses in \( \alpha \).

Definition. Let \( \#[\alpha] \) be \( \#_R[\alpha] - \#_L[\alpha] \). In words, count the closing parentheses and the occurrences of the nonterminal \( R \) and subtract the number of opening parentheses.

Theorem. For all \( B \xrightarrow{*} \alpha, \#[\alpha] = 0. \)

Corollary. For all \( w \in L(G), \#[w] = 0. \)
Proof. Clearly each production preserves the count $\#$, i.e., the count on the RHS of the production is the same as the count on the LHS.

1. $\#[B] = \#[\epsilon] = 0$
2. $\#[B] = \#[(R)B] = 0$
3. $\#[R] = \#[()] = 1$
4. $\#[R] = \#[(R)R] = 1$

So, any step in the derivation preserves the count.

1. $\#[\alpha B \beta] = \#[\alpha \beta]$
2. $\#[\alpha B \beta] = \#[\alpha (R) B \beta]$
3. $\#[\alpha R \beta] = \#[\alpha (R) \beta]$
4. $\#[\alpha R \beta] = \#[\alpha (R) R \beta]$

Given these observations, we now show that for all derivations $B \Rightarrow^* \gamma$ it is the case that $\#[\gamma] = 0$. 
In each step of these derivations (read from top to bottom),

\[ R + ) - ( \]

occurrences of \( R + \) right parens minus left parens

is preserved and is zero.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( B )</td>
<td>(0+0)-0</td>
<td>0</td>
</tr>
<tr>
<td>( RB )</td>
<td>(1+0)-1</td>
<td>0</td>
</tr>
<tr>
<td>( R(RB )</td>
<td>(2+0)-2</td>
<td>0</td>
</tr>
<tr>
<td>( R()B )</td>
<td>(1+1)-2</td>
<td>0</td>
</tr>
<tr>
<td>( ()() )</td>
<td>(0+2)-2</td>
<td>0</td>
</tr>
</tbody>
</table>
The proof is by induction on the derivation. The base case is a zero step derivation from \( B \). Obviously, for \( B \Rightarrow B \), we have \( \#[B] = 0 \). Assume all the previous steps in the derivation preserve the count. In other words, the induction hypothesis says for all sentential forms \( \gamma' \) if \( B \Rightarrow^* \gamma' \) we have \( \#[\gamma'] = 0 \). The last step of the derivation is

\[
\gamma' \xrightarrow{1} \gamma
\]

Since each of the four productions preserves the count, \( \#[\gamma] = 0 \).

QED
Linz 6th, Section 5.1, Example 5.2, page 132
Linz 6th, Section 5.1, Example 5.3, page 132
Linz 6th, Section 5.1, Example 5.4, page 133
Show the language $L = \{a^n b^m | n \neq m\}$ is context free. Observe

$$L = \{a^n b^m | n < m\} \cup \{a^n b^m | n > m\}$$

So, $G$ be:

$$S \rightarrow AS_1 \mid S_1B$$
$$S_1 \rightarrow aS_1b \mid \epsilon$$
$$A \rightarrow aA \mid a$$
$$B \rightarrow bB \mid b$$

Since $L = L(G)$, $L$ is a context-free language.
Concrete Syntax Trees

Every derivation gives rise to a figure called a concrete syntax tree. A node labeled with a nonterminal occurring on the left side of a production has children consisting of the symbols on the right side of that production.
Linz 6th, Section 5.1, Example 5.6, page 136
Derivation

In a grammar that is not linear, sentential forms with more than one nonterminal in them can be derived. In such cases, we have a choice in which production to use.

Definition. A derivation is said to be leftmost if in each step the leftmost nonterminal in the sentential form is replaced. We write

\[ \alpha \Rightarrow^{*}_{\text{lm}} \beta \]

Definition. A derivation is said to be rightmost if in each step the rightmost nonterminal in the sentential form is replaced. We write

\[ \alpha \Rightarrow^{*}_{\text{rm}} \beta \]
Theorem. All derivations in linear grammars are both leftmost and rightmost.

Theorem. If there is a derivation at all, then there is a leftmost derivation and a rightmost derivation.

Theorem. If there is a rightmost derivation, then there is a leftmost derivation, and vice versa.

Theorem. Some derivations are neither leftmost nor rightmost.
Consider the grammar with productions

1. $S \rightarrow aAB$
2. $A \rightarrow bBb$
3. $B \rightarrow A$
4. $B \rightarrow \epsilon$

Here are some distinct derivations (of the same string $abbbb$):

$S \xrightarrow{1} aAB \xrightarrow{1} abBbB \xrightarrow{1} abAbB \xrightarrow{1} abbbBbb \xrightarrow{1} abbbb$

$S \xrightarrow{1} aAB \xrightarrow{1} abBbB \xrightarrow{1} abBb \xrightarrow{1} abAb \xrightarrow{1} abbbBbb \xrightarrow{1} abbbb$

$S \xrightarrow{1} aAB \xrightarrow{1} aA \xrightarrow{1} abBb \xrightarrow{1} abAb \xrightarrow{1} abbbBb \xrightarrow{1} abbbb$
The derivations are not essentially different. This can be easily seen if we view a derivation as a tree. A tree can be built from a derivation by taking LHS nonterminal of each production $A \rightarrow \alpha$ used in the derivation as an internal node with children for each of the $|\alpha|$ grammar symbols in the RHS. Terminal symbols are leaves in the tree (they have no children).

The derivations on the previous frame/slide are instructions for building a tree using the productions 1, 2, 3, 2, 4, 4 with inconsequential differences in order.

Effectively leftmost is an building the tree depth-first with the internal nodes (nonterminals) ordered left to right. While right most is building a tree with the internal nodes (nonterminals) ordered right to left.
[Picture]
Three nonterminals and four productions lead to three mutually recursive, algebraic type declarations in Haskell with four constructors.

```
class Yield a where
    yield :: a -> String
```
```haskell
data S = P1 A B -- S -> a A B
data A = P2 B -- A -> b B b
data B = P3 A | P4 -- B -> A / [epsilon]

instance Yield S where
    yield (P1 a b) = 'a' : (yield a) ++ (yield b)
instance Yield A where
    yield (P2 b) = 'b' : yield b ++ ['b']
instance Yield B where
    yield (P3 a) = yield a
    yield (P4) = ""

d :: S
d = P1 (P2 (P3 (P2 P4))) P4

y :: String
y = yield d -- == abbb
```
data  S = P1 S | P2 S S | P3

instance  Yield S where
  yield (P1 s)   = 'a' : (yield s) ++ ['b']
  yield (P2 s1 s2) = (yield s1) ++ (yield s1)
  yield (P3)     = ""

d1, d2 :: S
  d1 = P1 (P1 P3)
  d2 = P2 P3 (P1 (P1 P3))

-- (yield d1) == (yield d2) == "aabb"
Definition. A context-free grammar $G$ is ambiguous, if there is more than one leftmost (or rightmost) derivation of the same sentence.

Different leftmost (or rightmost) derivations arise from different choices in productions for a single nonterminal in a sentential form, not just in the order to expand different occurrences of a nonterminals in a sentential form.
Consider the grammar with productions $S \rightarrow aSb \mid SS \mid \epsilon$

The sentence $aabb$ has two distinct leftmost derivations.

$$S \xrightarrow{1} aSb \xrightarrow{1} aaSbb \xrightarrow{1} aabb$$

$$S \xrightarrow{1} SS \xrightarrow{lm} S \xrightarrow{1} aSb \xrightarrow{1} aaSbb \xrightarrow{1} aabb$$

The first derivation is a leftmost derivation (and a rightmost derivation); the second derivation is a leftmost derivation of the same string ($aabb$). The derivations are different. The first derivation does not use the production $S \rightarrow SS$ while the second one does.

Hence, the grammar is ambiguous.
(Linz 6th, Section 5.2, Example 5.10, page 146)
Linz 6th, Section 5.2, Example 5.11, page 146–147
Linz 6th, Section 5.2, Example 5.12, page 147
Definition. A context-free language \( L \) is inherently ambiguous, if there is no unambiguous context-free grammar \( G \) for which \( L = L(G) \).
Linz 6th, Section 5.2, Example 5.13, page 149

\[ L = \{ a^n b^n c^m \} \cup \{ a^n b^m c^m \} \]

Notice that \( L = L_1 \cup L_2 \) where \( L_1 \) and \( L_2 \) are generated by the following context-free grammar with \( S \to S_1 | S_2 \) where

\[
\begin{align*}
S_1 & \to S_1 c | A \\
A & \to aAb | \epsilon \\
S_2 & \to aS_2 | B \\
B & \to bBc | \epsilon
\end{align*}
\]

This grammar is ambiguous as \( S_1 \Rightarrow^* abc \) and \( S_2 \Rightarrow^* abc \). But this not not mean the language is ambiguous. A rigorous argument that \( L \) is inherently ambiguous is technical and can be found, for example, in Harrison 1978.
Exhaustive Search

Linz 6th, Section 5.2, page 141.

Algorithm. Given a grammar $G$ without $\epsilon$-productions and unit productions and a string $w$, systematically construct all possible leftmost (or rightmost) derivations to determine if $w \in L(G)$.

We postpone this until we have we give all the necessary definitions.
Outline

Different Kinds of Grammars

Context-Free Grammars
  Leftmost, Rightmost Derivations
  Ambiguity

Chomsky and Greibach Normal Forms

Cleaning Up CFG’s

Brute Force Membership Test

CYK Algorithm
1. **Chomsky Normal Form**

   *Definition*: A grammar is said to be in CNF if all its productions are in one of two forms: \( A \rightarrow BC \) or \( A \rightarrow a \).

   Linz 6th, §6.2, definition 6.4, page 171
   HUM 3rd, §7.1.5, page 272
   Kozen, Lecture 21, definition 21.1, page 140

2. **Greibach Normal Form**

   *Definition*: A grammar is said to be in GNF if all its productions are in the form: \( A \rightarrow aX \) where \( a \in T \) and \( X \in V^* \).

   Linz 6th §6.2, definition 6.5, page 174
   HMU 3rd, §7.1, page 277
   Kozen, Lecture 21, definition 21.1, page 140

NB \( \epsilon \notin L(G) \).
There is another interesting normal form for grammars. This form, called Greibach Normal Form, after Sheila Greibach, has several interesting consequences. Since each use of a production introduces exactly one terminal into a sentential form, a string of length $n$ has a derivation of exactly $n$ steps. Also, if we apply the PDA construction to a Greibach-Normal grammar, then we get a PDA with no $\epsilon$-rules, thus showing that it is always possible to eliminate such transitions of a PDA.

HMU 3rd, section 7.1, page 277
Sheila Adele Greibach (b. 1939)

Sheila Greibach was born in New York City and received the A.B. degree from Radcliffe College in Linguistics and Applied Mathematics *summa cum laude* in 1960. She received the Ph.D. in Applied Mathematics from Harvard University in 1963. She joined the UCLA Faculty in 1969 and the Computer Science Department in 1970 and is now Emeritus Professor.
We will return to GNF after we introduce PDA’s.

Going back to CNF, it is easy to put a grammar in Chomsky Normal Form. And, because of that, there is a very convenient and efficient algorithm to determine membership in any CFL. This algorithm is known as the CYK Algorithm and has many uses, e.g., in natural language processing.
Before we put grammars in Chomsky normal form, we wish to address some petty annoyances in “untidy” or “meandering” grammars.
Outline

Different Kinds of Grammars

Context-Free Grammars
  Leftmost, Rightmost Derivations
  Ambiguity

Chomsky and Greibach Normal Forms

Cleaning Up CFG’s

Brute Force Membership Test

CYK Algorithm
Before we can study context-free languages in greater depth, we must attend to some technical matters. The definition of a context-free grammar imposes no restriction whatsoever on the right side of a production. However, complete freedom is not necessary and, in fact, is a detriment in some arguments.

Linz 6th, Chapter 6, page 156.
We must get rid of all $\varepsilon$-productions $A \rightarrow \varepsilon$. and unit productions $A \rightarrow B$. These are bothersome because they make it hard to determine whether applying a production makes any progress toward deriving a string of terminals. For instance, with unit productions, there can be loops in the derivation, and with $\varepsilon$-productions, one can generate very long strings of nonterminals and then erase them all.

Kozen, page 141.
An Example of a “Meandering” Grammar

\[ S \to AS \mid B \]
\[ A \to B \mid \epsilon \]
\[ B \to A \mid b \]

Needless looping:

\[ S \Rightarrow B \Rightarrow b \]
\[ S \Rightarrow B \Rightarrow A \Rightarrow B \Rightarrow b \]
\[ S \Rightarrow B \Rightarrow A \Rightarrow B \Rightarrow A \Rightarrow B \Rightarrow b \]

Vanishing nonterminals:

\[ S \Rightarrow AS \Rightarrow^* AAAS \Rightarrow AAAB \Rightarrow^* B \Rightarrow b \]
\[ S \Rightarrow AS \Rightarrow^* AAAAAS \Rightarrow AAAAAB \Rightarrow^* B \Rightarrow b \]
Another Example

\[ S \rightarrow W \mid X \mid Z \]
\[ W \rightarrow A \]
\[ A \rightarrow B \]
\[ B \rightarrow C \]
\[ C \rightarrow Aa \]
\[ X \rightarrow C \]
\[ Y \rightarrow aY \mid a \]
\[ Z \rightarrow \epsilon \]
1. The nonterminal $Y$ can never be derived by the start symbol $S$. It is useless.

2. The production $S \rightarrow X$ can easily be made redundant by skipping directly to $C$ since $X \rightarrow C$.

3. The nonterminal $A$ cannot derive any string in $T^*$ because any sentential form with $A$ cannot get rid of the $A$.

4. We do not need the nonterminal $Z$, because it can generate nothing—the empty string. It seems counterproductive to generate a long sentential form only to cut it back later.

The grammar is too complicated—the only string in $L(G)$ is $\epsilon$. 
Theorem. For any grammar \( G = \langle T, V, P, S \rangle \), and for any nonterminal \( B \in V \) such that \( B \stackrel{*}{\Rightarrow}_G \gamma \), if \( A \rightarrow \alpha B \beta \in P \), then \( L(G) = L(G') \) for the grammar \( G' = \langle T, V, P \cup \{ A \rightarrow \alpha \gamma \beta \}, S \rangle \).

Adding a shortcut does not change the grammar.
Summary of the Process

It is possible to systematically modify grammars to eliminate productions that are not helpful making them simpler to use. A two-step process gets rid of the most blatant problems:

1. Remove $\epsilon$ and unit productions,
2. Remove unreachable and nonproductive nonterminals.

In what follows we elaborate and give algorithms.
Definition

For a context-free grammar $G = \langle T, V, P, S \rangle$:

- A nonterminal $A \in N$ is said to be **nullable** if $A \Rightarrow^* \epsilon$.
- A nonterminal $A \in N$ is said to be **reachable** if $S \Rightarrow^* \alpha A \beta$ for some $\alpha, \beta \in (\Sigma \cup V)^*$.
- A nonterminal $A \in N$ is said to be **productive** if $A \Rightarrow^* w$ for some $w \in \Sigma^*$.
- A nonterminal $A \in N$ is said to be **useful** if it is both reachable and productive.
- A pair of nonterminals $A, B \in V$ is said to be a **unit pair** if $A \Rightarrow^* B$. 
Nullability. Any nonterminal $A$ is nullable if there is a production of the form $A \rightarrow \epsilon$. $A$ is also nullable, if there is a production of the form $A \rightarrow V_1 V_2 \cdots V_k$ and all $V_i$ is nullable for $1 \leq i \leq k$.

Reachability. $A$ is reachable if there is a production of the form $S \rightarrow \alpha A \beta$, or if there is any $B \in N$ where $B \rightarrow \alpha A \beta$ and $B$ is reachable.

Productiveness/Generating. A nonterminal $A \in N$ is productive if there is some production of the form $A \rightarrow w$ for some $w \in \Sigma^*$ or if there is some production $A \rightarrow \alpha$ for which all the nonterminals in $\alpha$ are productive.
\( V_n := \emptyset; \quad -- \textit{assume nothing is nullable} \)

\textbf{loop}
\[
V_o := V_n;
\]
\textbf{for} \( X \rightarrow \alpha \in P \) \textbf{loop}
\[
\text{add } X \text{ to } V_n \text{ if } \alpha \in V_o^*;
\]
\textbf{end loop;}
\textbf{exit when} \( V_n = V_o; \)
\textbf{end loop;}

Compute the set \( V_o \) of all nullable nonterminals
\[ V_n = \emptyset; \quad -- \text{assume nothing is nullable} \]

loop
  \[ V_o = V_n; \]
  for \( X \rightarrow \alpha \in P \) loop
    if \( \alpha \in V_o^* \) then
      \[ V_n := V_n \cup X; \]
    end if;
  end loop;
  exit when \( V_n = V_o; \)
end loop;

ALTERNATE: Compute the set \( V_o \) of all nullable nonterminals
Compute the set of all reachable nonterminals
Compute the set of all productive nonterminals
Additional and Pointless Algorithm

\[
Z_n = \{\langle Y, Y \rangle \}; \quad -- \quad Y \Rightarrow^* Y \quad for \quad all \quad Y \in V
\]

loop
\[
Z_o := Z_n;
\]
for \( A \rightarrow B \in P \) loop \quad -- \quad unit \ production
\[
\text{for} \quad \langle B, C \rangle \in Z_n \ \text{loop} \quad -- \quad B \Rightarrow^* C
\]
add \( \langle A, C \rangle \) to \( Z_n \);
\]
end loop;
end loop;
exit when \( Z_n = Z_o \);
end loop;

Compute the set of all units pairs \( A \Rightarrow^* B \)
An $\epsilon$ production is one of the form $A \rightarrow \epsilon$ for some nonterminal $A$. All $\epsilon$ productions can be removed from a grammar (except possibly $S \rightarrow \epsilon$).

A nonterminal is said to be nullable if $N \Rightarrow^* \epsilon$.

Find all nullable nonterminals. Replace every production with a nullable nonterminal in the RHS with two productions: one with and one without the nullable nonterminal. If the leaving out the nonterminal results in a $\epsilon$ production, then do not add it.

If a production has $n$ nullable nonterminals in the RHS, then it is replaced by $2^n$ productions (or $2^n - 1$ productions).
\[ P_n := P; \quad \text{-- keep all the original productions} \]

loop
  \[ P_o := P_n; \]
  for \( X \rightarrow \alpha N \beta \in P_n \) loop
    if \( N \rightarrow \epsilon \in P_n \) then
      add \( X \rightarrow \alpha \beta \) to \( P_n \);
    end if;
  end loop;
  for \( X \rightarrow B \in P_n \) (where \( B \in V \)) loop
    if \( B \rightarrow \beta \in P_n \) then
      add \( X \rightarrow \beta \) to \( P_n \);
    end if;
  end loop;
  exit when \( P_o = P_n \);
end loop;

Make \( \epsilon \) and unit productions superfluous
\[ P_n := P; \quad \text{-- keep all the original productions} \]

loop
\[ P_o := P_n; \]
for \( X \rightarrow \alpha N \beta \in P_n \) loop
\[ \text{add } X \rightarrow \alpha \beta \text{ to } P_n \text{ if } N \rightarrow \epsilon \in P_n; \]
end loop;
for \( X \rightarrow B \in P_n \) (where \( B \in V \)) loop
\[ \text{add } X \rightarrow \beta \text{ to } P_n \text{ if } B \rightarrow \beta \in P_n; \]
end loop;
exit when \( P_o = P_n \);
end loop;

Make \( \epsilon \) and unit productions superfluous
1. Examples first.
2. Proof of correctness afterward.
The previous algorithm terminates. The RHS of each new production added is not longer than the longest production of the original grammar. There is only a finite number of such possible productions.
Theorem. No unit production is required in a minimal step derivation any grammar constructed by the previous algorithm. Suppose a unit production $A \rightarrow B$ is used in the shortest possible derivation of $S \Rightarrow^* x$. Then

\[
S \Rightarrow^m \alpha A \beta \xrightarrow{1} \alpha B \beta \Rightarrow^n \eta B \theta \xrightarrow{1} \eta \gamma \delta \Rightarrow^k x
\]

\[
S \Rightarrow^m \alpha A \beta \xrightarrow{1} \alpha \gamma \theta \Rightarrow^n \eta \gamma \theta \Rightarrow^k x
\]
Theorem. No epsilon production is required in a minimal step derivation any grammar constructed by the previous algorithm. Suppose an $\epsilon$ production $B \rightarrow \epsilon$ is used in the shortest possible derivation of $S \Rightarrow^* x$. Then

$$S \Rightarrow^m \eta A \theta \xrightarrow{1} \eta \alpha B \beta \theta \Rightarrow^n \gamma B \delta \xrightarrow{1} \gamma \delta \Rightarrow^k x$$

$$S \Rightarrow^m \eta A \theta \xrightarrow{1} \eta \alpha \beta \theta \Rightarrow^n \gamma \delta \Rightarrow^*^k x$$
Theorem: Let $G'$ be the grammar after running the “eu” algorithm on context-free grammar $G$. If $\epsilon \in L(G)$, then $S \rightarrow \epsilon$ in the productions of $G'$. 
If a nonterminal $A$ is useful, then $S \Rightarrow^* \alpha A \beta \Rightarrow^* w$ for some $\alpha, \beta \in (T \cup V)^*$ and $w \in T^*$

Theorem. Deleting all useless nonterminals and any production containing them on the LHS or RHS results in a grammar that generates the same language as originally.
Theorem

Every CFG $G = \langle \Sigma, N, P, S \rangle$ can be (effectively) transformed to one without cycles (unit productions), non-productive, or unreachable nonterminals. (This means that unit productions are unnecessary.)

- A nullable nonterminal $N$ is one for which $N \Rightarrow^* \epsilon$.
- A productive nonterminal $N$ is one for which $N \Rightarrow^* w$ for some $w \in \Sigma^*$.
- A reachable nonterminal $N$ is one for which $S \Rightarrow^* \alpha N \beta$ for some $\alpha, \beta \in (\Sigma \cup N)^*$.

All epsilon productions may also (effectively) be eliminated from a CFG, if the language does not contain the empty string. If the language contains the empty string, no epsilon productions are necessary save one: $S \rightarrow \epsilon$. 
Algorithm to Put a Grammar in CNF

First, eliminate $\epsilon$ productions and unit productions. One might as well eliminate useless productions.

1. See that all RHS of length two or more consist only of nonterminals by introducing a nonterminal $T_a$ and adding productions like $T_a \rightarrow a$ to the grammar.

2. For productions of length $k$ greater than two, add a cascade of $k - 2$ new nonterminals and productions.

HMU 3rd, §7.1.5, page 272.
Kozen, Lecture 21, page 140.
Convert to CNF.
Linz, example 6.8, page 173. $S \rightarrow ABa, A \rightarrow aab, B \rightarrow Ac$.
Kozen, example 21.4. $L = \{a^n b^n \mid n \geq 1\}$
Kozen, example 21.5. Balanced parentheses.

$$S \rightarrow [S] \mid SS \mid \epsilon$$
$$S \rightarrow ASB \mid SS \mid AB, \quad A \rightarrow [, \quad B \rightarrow ]$$
$$S \rightarrow AC \mid SS, \quad A \rightarrow [, \quad B \rightarrow ], \quad C \rightarrow SB$$
Outline

Different Kinds of Grammars

Context-Free Grammars
  Leftmost, Rightmost Derivations
  Ambiguity

Chomsky and Greibach Normal Forms

Cleaning Up CFG’s

Brute Force Membership Test

CYK Algorithm
Exhaustive Search

Linz 6th, Section 5.2, page 141.

Algorithm. Given a grammar $G$ without $\epsilon$-productions and unit productions and a string $w$, systematically construct all possible leftmost (or rightmost) derivations and see if the sentential forms are consistent with $w$.

Lemma. Given a grammar $G$ without $\epsilon$-productions. If $S \Rightarrow_G \alpha \Rightarrow_G x$, then the $|\alpha| \leq |x|$. Proof. For all nonterminals $N$, if $N \Rightarrow x$, then $1 \leq |x|$.

Lemma. Given a grammar $G$ without $\epsilon$-productions and unit productions. Let $\#_+(\alpha)$ be the number of terminal symbols in $\alpha$ added to the length of $w$. If $\beta \Rightarrow_G \gamma$, then $\#_+(\beta) < \#_+(\gamma)$. And, hence, if $\beta \Rightarrow_G \gamma$, then $\#_+(\beta) < \#_+(\gamma)$.

Corollary. If $\alpha \Rightarrow_G w$, then the number steps is bounded by $\#_+(w) = 2|w|$.
Linz 6th, Section 5.2, Example 5.7, page 141.
Busch’s notes, class09, page 12ff.
Outline

Different Kinds of Grammars

Context-Free Grammars
  Leftmost, Rightmost Derivations
  Ambiguity

Chomsky and Greibach Normal Forms

Cleaning Up CFG’s

Brute Force Membership Test

CYK Algorithm
Cocke-Kasami-Younger Algorithm (CYK)

Linz 6th, section 6.3, page 178
Busch’s notes, class10כ, page 22ff
HMU 3rd, section 7.4.4, page 303
Kozen, Lecture 27, page 191

Kasami 1965; Younger 1967; Cocke and Schwartz 1970
Let $G = \langle T, V, P, S \rangle$ be a CFG in Chomsky normal form. We can determine if a string $s$ is in $L(G)$. Suppose $s = a_0a_1, \ldots, a_{n-1}$. We write $s[i : j]$ where $0 \leq i \leq j \leq n - 1$ for the substring of $s$ of length $j - i + 1$ beginning at position $i$ and ending at position $j$.

We require a mutable, two dimensional, triangular array $M$ containing sets of nonterminals with the intention that $A \in M[i, j]$ iff $A \Rightarrow^* s[i : j]$. 
### CYK Algorithm

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>...</th>
<th>i</th>
<th>...</th>
<th>n-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>s[0 : 0]</td>
<td>s[0 : 1]</td>
<td>s[0 : 2]</td>
<td>...</td>
<td>s[0 : i]</td>
<td>...</td>
<td>s[0 : n - 1]</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>s[1 : 1]</td>
<td>s[1 : 2]</td>
<td>...</td>
<td>s[1 : i]</td>
<td>...</td>
<td>s[1 : n - 1]</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>s[2 : 2]</td>
<td>...</td>
<td>s[2 : i]</td>
<td>...</td>
<td>s[2 : n - 1]</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td>...</td>
<td></td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>r</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>s[n - 1 : n - 1]</td>
</tr>
</tbody>
</table>
for $i$ in $0, \ldots, n-1$ loop
    $M[i, i] := \emptyset$
    for $N \rightarrow a$ in $P$ loop
        add $N$ to $M[i, i]$ if $a = a_i$ -- $N \Rightarrow^*_G s[i : i]$
    end loop
end loop
CYK Algorithm

for $g$ in $1\ldots,n-1$ loop -- every substring length
  for $r$ in $0\ldots,n-g-1$ loop -- starting at $r$
    for $m$ in $r\ldots,r+g-1$ loop -- every midpoint
      -- $s[r:r+g] = s[r:m] + s[m+1:r+g]$
      -- the nonterminals generating $s[r:m]$
      $L = M[r,m]$
      -- the nonterminals generating $s[m+1:r+g]$
      $R = M[m+1,r+g]$
    for $A \rightarrow BC$ in $P$ loop
      -- $A \Rightarrow_{G}^{*} s[r:r+g]$
      add $A$ to $M[r,r+g]$ if $B \in L$ and $C \in R$
    end loop
  end loop
end loop
end loop
return $S \in M[0,n-1]$
for $i := 0$ to $n - 1$ do
  begin
    $T_{i,i+1} := \emptyset$; /* initialize to $\emptyset$ */
    for $A \rightarrow a$ a production of $G$ do
      if $a = x_{i,i+1}$ then $T_{i,i+1} := T_{i,i+1} \cup \{A\}$
    end;
  for $m := 2$ to $n$ do /* for each length $m \geq 2$ */
    for $i := 0$ to $n - m$ do /* for each substring */
      begin /* of length $m$ */
        $T_{i,i+m} := \emptyset$; /* initialize to $\emptyset$ */
        for $j := i + 1$ to $i + m - 1$ do /* for all ways to break */
          for $A \rightarrow BC$ a production of $G$ do /* up the string */
            if $B \in T_{i,j} \land C \in T_{j,i+m}$
              then $T_{i,i+m} := T_{i,i+m} \cup \{A\}$
        end;
      end;
function CYK-PARSE(words, grammar) returns P, a table of probabilities

N ← LENGTH(words)
M ← the number of nonterminal symbols in grammar
P ← an array of size [M, N, N], initially all 0

/* Insert lexical rules for each word */
for i = 1 to N do
    for each rule of form (X → wordsi [p]) do
        P[X, i, 1] ← p

/* Combine first and second parts of right-hand sides of rules, from short to long */
for length = 2 to N do
    for start = 1 to N - length + 1 do
        for len1 = 1 to N - 1 do
            len2 ← length - len1
            for each rule of the form (X → Y Z [p]) do
                P[X, start, length] ← MAX(P[X, start, length],
                                            P[Y, start, len1] × P[Z, start + len1, len2] × p)

return P

Figure 23.5 The CYK algorithm for parsing. Given a sequence of words, it finds the most probable derivation for the whole sequence and for each subsequence. It returns the whole table, P, in which an entry P[X, start, len] is the probability of the most probable X of length len starting at position start. If there is no X of that size at that location, the probability is 0.
Example

Linz 6th, §6.3, example 6.11, page 179.

\[ S \rightarrow AB \]
\[ A \rightarrow BB \mid a \]
\[ B \rightarrow AB \mid b \]

\[
\begin{array}{cccccc}
  a & a & b & b & b \\
\end{array}
\]

\[
\{ A \}
\]

\[
\begin{array}{cccc}
  aa & ab & bb & bb \\
\end{array}
\]

\[
\begin{array}{cccc}
  aab & abb & bbb \\
\end{array}
\]

\[
\begin{array}{cccc}
  aaba & abbb \\
\end{array}
\]

\[
\begin{array}{c}
  aabbb \\
\end{array}
\]
Example

Linz 6th, §6.3, example 6.11, page 179.

\[
S \rightarrow AB
\]
\[
A \rightarrow BB \mid a
\]
\[
B \rightarrow AB \mid b
\]

\[
\begin{array}{cccccc}
  a & a & b & b & b \\
  \{A\} & \{A\} & & & \\
  aa & ab & bb & bb & \\
  aab & abb & bbb & \\
  aaba & abbb & \\
  aabbb & \\
\end{array}
\]
Example

Linz 6th, §6.3, example 6.11, page 179.

\[
\begin{align*}
S & \to AB \\
A & \to BB \mid a \\
B & \to AB \mid b
\end{align*}
\]

\[
\begin{array}{cccccc}
a & a & b & b & b & b \\
\{A\} & \{A\} & \{B\} & \{A\} & \{B\} & \{B\}
\end{array}
\]

\[
\begin{array}{cccccc}
aa & ab & bb & bb & bbb & bbb
\end{array}
\]

\[
\begin{array}{cccccc}
aab & abb & bbb & bbb & bbb & bbb
\end{array}
\]

\[
\begin{array}{cccccc}
aaba & abbb & abbb & abbb & abbb & abbb
\end{array}
\]

\[
aabbb
\]
Example

Linz 6th, §6.3, example 6.11, page 179.

\[ S \rightarrow AB \]
\[ A \rightarrow BB \mid a \]
\[ B \rightarrow AB \mid b \]

\[
\begin{array}{cccccc}
a & a & b & b & b \\
\{A\} & \{A\} & \{B\} & \{B\} \\
\end{array}
\]

\[
\begin{array}{cccc}
aa & ab & bb & bb \\
aab & abb & bbb \\
aaba & abbb \\
aabbb \\
\end{array}
\]
Example

Linz 6th, §6.3, example 6.11, page 179.

\[ S \rightarrow AB \]
\[ A \rightarrow BB \mid a \]
\[ B \rightarrow AB \mid b \]

\[
\begin{array}{cccccc}
{A} & {A} & {B} & {B} & {B} \\
\{A\} & \{A\} & \{B\} & \{B\} & \{B\} \\
{a} & {a} & {b} & {b} & {b} \\
{a} & {a} & {b} & {b} & {b} \\
{a} & {a} & {b} & {b} & {b} \\
\end{array}
\]

{aab} {abb} {bbb}
{aab} {abb} {bbb}
{aab} {abb} {bbb}
{aab} {abb} {bbb}

aabbb
Example

Linz 6th, §6.3, example 6.11, page 179.

\[
S \rightarrow AB
\]
\[
A \rightarrow BB \mid a
\]
\[
B \rightarrow AB \mid b
\]

\[
\begin{array}{cccccc}
a & a & b & b & b & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{B\}
\end{array}
\]

\[
\begin{array}{cccc}
aa & ab & bb & bb \\
\emptyset
\end{array}
\]

\[
\begin{array}{cccc}
aab & abb & bbb
\end{array}
\]

\[
\begin{array}{cccc}
aaba & abbb
\end{array}
\]

\[
aabbb
\]
Example

Linz 6th, §6.3, example 6.11, page 179.

\[
S \rightarrow AB \\
A \rightarrow BB \mid a \\
B \rightarrow AB \mid b
\]

\[
\begin{array}{cccc}
   a & a & b & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{B\}
\end{array}
\]

\[
\begin{array}{cccc}
   aa & ab & bb & bb \\
\emptyset & \{S, B\}
\end{array}
\]

\[
\begin{array}{cccc}
   aab & abb & bbb \\
aaba & abbb \\
aabbb
\end{array}
\]
Example

Linz 6th, §6.3, example 6.11, page 179.

\[
\begin{align*}
S & \rightarrow AB \\
A & \rightarrow BB \mid a \\
B & \rightarrow AB \mid b
\end{align*}
\]

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>a</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>{A}</td>
<td>{A}</td>
<td>{B}</td>
<td>{B}</td>
<td>{B}</td>
<td>{B}</td>
</tr>
<tr>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td>{S, B}</td>
<td></td>
<td>{A}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aab</td>
<td>abb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| aaba | abbb |

| aabbb |
Example

Linz 6th, §6.3, example 6.11, page 179.

\[
S \rightarrow AB \\
A \rightarrow BB \mid a \\
B \rightarrow AB \mid b
\]

\[
\begin{array}{cccccc}
a & a & b & b & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{B\}
\end{array}
\]

\[
\begin{array}{cccc}
\emptyset & \{S, B\} & \{A\} & \{A\}
\end{array}
\]

\[
\begin{array}{cccc}
aab & abb & bbb
\end{array}
\]

\[
aaba & abbb
\]

\[
aabbb
\]
Example

Linz 6th, §6.3, example 6.11, page 179.

\[ S \rightarrow AB \]
\[ A \rightarrow BB \mid a \]
\[ B \rightarrow AB \mid b \]

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{A}</td>
<td>{A}</td>
<td>{B}</td>
<td>{B}</td>
<td>{B}</td>
</tr>
<tr>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>\Ø</td>
<td>{S, B}</td>
<td>{A}</td>
<td>{A}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aab</td>
<td>abb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{S, B}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaba</td>
<td>abbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabb</td>
<td>aabb</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example

Linz 6th, §6.3, example 6.11, page 179.

\[ S \rightarrow AB \]
\[ A \rightarrow BB \mid a \]
\[ B \rightarrow AB \mid b \]

\[
\begin{array}{cccccc}
  & a & a & b & b & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{B\} \\
  & aa & ab & bb & bb \\
\emptyset & \{S, B\} & \{A\} & \{A\} \\
  & aab & abb & bbb \\
\{S, B\} & \{A\} \\
aaba & abbb \\
aabbb
\end{array}
\]
Example

Linz 6th, §6.3, example 6.11, page 179.

\[
S \rightarrow AB \\
A \rightarrow BB \mid a \\
B \rightarrow AB \mid b
\]

\[
\begin{array}{cccccc}
 a & a & b & b & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{B\} \\
 aa & ab & bb & bb \\
\emptyset & \{S, B\} & \{A\} & \{A\} \\
 aab & abb & bbb \\
\{S, B\} & \{A\} & \{S, B\} \\
 aaba & abbb \\
aabbb
\end{array}
\]
Example

Linz 6th, §6.3, example 6.11, page 179.

\[ S \rightarrow AB \]
\[ A \rightarrow BB \mid a \]
\[ B \rightarrow AB \mid b \]

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>{A}</td>
<td>{A}</td>
<td>{B}</td>
<td>{B}</td>
<td>{B}</td>
</tr>
<tr>
<td>a</td>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td></td>
</tr>
<tr>
<td>S</td>
<td>{S, B}</td>
<td>{A}</td>
<td>{A}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>aab</td>
<td>abb</td>
<td>bbb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S</td>
<td>{S, B}</td>
<td>{A}</td>
<td>{S, B}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>aaba</td>
<td>abbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S</td>
<td>{A}</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>aabbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ \emptyset \]
Example

Linz 6th, §6.3, example 6.11, page 179.

\[
\begin{align*}
S & \rightarrow AB \\
A & \rightarrow BB \mid a \\
B & \rightarrow AB \mid b
\end{align*}
\]

\[
\begin{array}{cccccc}
a & a & b & b & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{B\}
\end{array}
\]

\[
\begin{array}{cccc}
aa & ab & bb & bb \\
\emptyset & \{S, B\} & \{A\} & \{A\}
\end{array}
\]

\[
\begin{array}{cccc}
aab & abb & bbb \\
\{S, B\} & \{A\} & \{S, B\}
\end{array}
\]

\[
\begin{array}{cccc}
aaba & abbb \\
\{A\} & \{S, B\}
\end{array}
\]

aabbb
Example

Linz 6th, §6.3, example 6.11, page 179.

\[
S \rightarrow AB \\
A \rightarrow BB \mid a \\
B \rightarrow AB \mid b
\]

\[
\begin{array}{cccccc}
a & a & b & b & b & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{B\}
\end{array}
\]

\[
\begin{array}{cccccc}
aa & ab & bb & bb & & \\
\emptyset & \{S, B\} & \{A\} & \{A\} & & \\
\end{array}
\]

\[
\begin{array}{cccccc}
aab & abb & bbb & & & \\
\{S, B\} & \{A\} & \{S, B\} & & & \\
\end{array}
\]

\[
\begin{array}{cccccc}
aaba & abbb & & & & \\
\{A\} & \{S, B\} & & & & \\
\end{array}
\]

\[
\begin{array}{cccccc}
aabbb & & & & & \\
\{S, B\} & & & & & \\
\end{array}
\]
Example

Linz 6th, §6.3, exercise 4, page 180. Use the CYK method to determine if the string $w = aaabbb$ is in the language generated by the grammar $S \rightarrow aSb \mid b$.

First, convert the grammar to CNF.

\begin{align*}
S & \rightarrow AC \mid b \\
C & \rightarrow SB \\
A & \rightarrow a \\
B & \rightarrow b
\end{align*}
Example (Continued)


\[
a \quad a \quad a \quad a \quad b \quad b \quad b \quad b
\]
\[
\{A\}
\]

\[
aa \quad aa \quad ab \quad bb \quad bb \quad bb
\]

\[
aaa \quad aab \quad abbb \quad bbb
\]

\[
aaabb \quad aabbb \quad abbb \quad bbb
\]

\[
aabbb
\]
Example (Continued)


\[
\begin{align*}
  & a & a & a & b & b & b & b \\
  \{A\} & \{A\} \\
  & aa & aa & ab & bb & bb & bb \\
  & aaa & aab & abbb & bbb & bbb \\
  & aaab & aabb & abbb & bbb \\
  & aabbb
\end{align*}
\]
Example (Continued)


\[
\begin{array}{cccccccc}
  a & a & a & b & b & b & b & b \\
  \{A\} & \{A\} & \{A\} & & & & & \\
  aa & aa & ab & bb & bb & bb & & \\
  aaa & aab & abb & bbb & bbb & & \\
  aaab & aabb & abbb & bbb & & \\
  aabbb & \end{array}
\]
Example (Continued)


\[
\begin{array}{ccccccc}
  a & a & a & b & b & b & b \\
  \{A\} & \{A\} & \{A\} & \{S, B\} & \\
  aa & aa & ab & bb & bb & bb \\
  aaa & aab & abb & bbb & bbb \\
  aaab & aabb & abbb & bbb \\
  aabbb
\end{array}
\]
Example (Continued)


\[
\begin{array}{cccccc}
  a & a & a & b & b & b \\
  \{A\} & \{A\} & \{A\} & \{S, B\} & \{S, B\} \\
  aa & aa & ab & bb & bb & bb \\
  aaa & aab & abb & bbb & bbb \\
  aaab & aabb & abbb & bbb & \\
  aabbb & \\
\end{array}
\]
Example (Continued)


\[
\begin{array}{ccccccc}
  a & a & a & b & b & b & b \\
  \{A\} & \{A\} & \{A\} & \{S, B\} & \{S, B\} & \{S, B\} \\
  aa & aa & ab & bb & bb & bb \\
  aaaa & aabb & abbb & bbbb & bbbb \\
  aaabb & aabbb & abbbb & bbb \\
  aabbb
\end{array}
\]
Example (Continued)


<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>$a$</td>
<td>$a$</td>
<td>$b$</td>
<td>$b$</td>
<td>$b$</td>
</tr>
<tr>
<td>${A}$</td>
<td>${A}$</td>
<td>${A}$</td>
<td>${S, B}$</td>
<td>${S, B}$</td>
<td>${S, B}$</td>
</tr>
</tbody>
</table>

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$aa$</td>
<td>$aa$</td>
<td>$ab$</td>
<td>$bb$</td>
<td>$bb$</td>
</tr>
</tbody>
</table>

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$aaa$</td>
<td>$aab$</td>
<td>$abb$</td>
<td>$bbb$</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$aaab$</td>
<td>$aabb$</td>
<td>$abbb$</td>
</tr>
</tbody>
</table>

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$aabb$</td>
</tr>
</tbody>
</table>

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$aabbb$</td>
</tr>
</tbody>
</table>
Example (Continued)


<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>{A}</td>
<td>{A}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
</tr>
<tr>
<td>aa</td>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaa</td>
<td>aab</td>
<td>abb</td>
<td>bbb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaab</td>
<td>aabb</td>
<td>abbb</td>
<td>bbbb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example (Continued)


\[
\begin{array}{ccccccc}
 a & a & a & b & b & b & b \\
\{A\} & \{A\} & \{A\} & \{S, B\} & \{S, B\} & \{S, B\} & \{S, B\} \\
 aa & aa & ab & bb & bb & bb & bb \\
\emptyset & \emptyset & \\
aaa & aab & abb & bbb & bbb & bbb & bbb \\
\end{array}
\]
Example (Continued)


\[
\begin{array}{ccccccc}
  a & a & a & b & b & b & b \\
  \{A\} & \{A\} & \{A\} & \{S, B\} & \{S, B\} & \{S, B\} & \{S, B\} \\
  aa & aa & ab & bb & bb & bb & bb \\
  \emptyset & \emptyset & \emptyset & \emptyset \\
  aaa & aab & abb & bbb & bbb \\
  aaab & aabb & abbb & bbb \\
  aabbb
\end{array}
\]
Example (Continued)


<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>{A}</td>
<td>{A}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
</tr>
<tr>
<td>aa</td>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>{C}</td>
<td>{C}</td>
<td>{C}</td>
<td>{C}</td>
<td></td>
</tr>
<tr>
<td>aaa</td>
<td>aab</td>
<td>abb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaab</td>
<td>aabb</td>
<td>abbb</td>
<td>bbbb</td>
<td>bbbb</td>
<td>bbbb</td>
<td>bbbb</td>
<td></td>
</tr>
</tbody>
</table>

aabbb
Example (Continued)


### Table

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{A}</td>
<td>{A}</td>
<td>{A}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
</tr>
<tr>
<td></td>
<td>aa</td>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
</tr>
<tr>
<td></td>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>{C}</td>
<td>{C}</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>aaa</td>
<td>aab</td>
<td>abb</td>
<td>bbb</td>
<td>bbb</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>aaab</td>
<td>aabb</td>
<td>abbb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>aabbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example (Continued)


\[
\begin{array}{cccccccc}
  a & a & a & b & b & b & b & b \\
  \{A\} & \{A\} & \{A\} & \{S, B\} & \{S, B\} & \{S, B\} & \{S, B\} \\
  aa & aa & ab & bb & bb & bb & & \\
  \emptyset & \emptyset & \emptyset & \{C\} & \{C\} & \{C\} & & \\
  aaa & aab & abb & bbb & bbb & & & \\
  & & & & & & & \\
  aaab & aabb & abbb & bbb & & & & \\
  & & & & & & & \\
  aabbb & & & & & & & \\
\end{array}
\]
Example (Continued)


<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{A}</td>
<td>{A}</td>
<td>{A}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
</tr>
<tr>
<td>a</td>
<td>aa</td>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>{C}</td>
<td>{C}</td>
<td>{C}</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>aaa</td>
<td>aab</td>
<td>abb</td>
<td>bbb</td>
<td>bbb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td>aab</td>
<td>abb</td>
<td>bbb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>aaab</td>
<td>aabb</td>
<td>abbb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>aabbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>aabbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>b</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example (Continued)


<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>$a$</td>
<td>$a$</td>
<td>$b$</td>
<td>$b$</td>
<td>$b$</td>
<td>$b$</td>
</tr>
<tr>
<td>{A}</td>
<td>{A}</td>
<td>{A}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
</tr>
<tr>
<td>$aa$</td>
<td>$aa$</td>
<td>$ab$</td>
<td>$bb$</td>
<td>$bb$</td>
<td>$bb$</td>
<td></td>
</tr>
<tr>
<td>\Ø</td>
<td>\Ø</td>
<td>\Ø</td>
<td>{C}</td>
<td>{C}</td>
<td>{C}</td>
<td></td>
</tr>
<tr>
<td>$aaa$</td>
<td>$aab$</td>
<td>$abb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>\Ø</td>
<td>\Ø</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$aaab$</td>
<td>$aabb$</td>
<td>$abbb$</td>
<td>$bbb$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$aabbb$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example (Continued)


<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td></td>
<td>{A}</td>
<td>{A}</td>
<td>{A}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
</tr>
<tr>
<td>aa</td>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>{C}</td>
<td>{C}</td>
<td>{C}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaa</td>
<td>aab</td>
<td>abb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>{S}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaab</td>
<td>aabb</td>
<td>abbb</td>
<td>bbbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>aabbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

aabbb
Example (Continued)


<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{A}</td>
<td>{A}</td>
<td>{A}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
</tr>
<tr>
<td>aa</td>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>{C}</td>
<td>{C}</td>
<td>{C}</td>
<td></td>
</tr>
<tr>
<td></td>
<td>aaa</td>
<td>aab</td>
<td>abbb</td>
<td>bbb</td>
<td>bbb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>{S}</td>
<td>Ø</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>aaab</td>
<td>aabb</td>
<td>abbb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>aabbb</td>
<td>aabb</td>
<td>abbb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>aabbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example (Continued)


<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>$a$</td>
<td>$a$</td>
<td>$b$</td>
<td>$b$</td>
<td>$b$</td>
<td>$b$</td>
</tr>
<tr>
<td>${A}$</td>
<td>${A}$</td>
<td>${A}$</td>
<td>${S, B}$</td>
<td>${S, B}$</td>
<td>${S, B}$</td>
<td>${S, B}$</td>
</tr>
<tr>
<td>$aa$</td>
<td>$aa$</td>
<td>$ab$</td>
<td>$bb$</td>
<td>$bb$</td>
<td>$bb$</td>
<td>$bb$</td>
</tr>
<tr>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>${C}$</td>
<td>${C}$</td>
<td>${C}$</td>
<td>${C}$</td>
</tr>
<tr>
<td>$aaa$</td>
<td>$aab$</td>
<td>$abb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
</tr>
<tr>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>${S}$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>$aaab$</td>
<td>$aabb$</td>
<td>$abbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
</tr>
</tbody>
</table>

$aabbb$
Example (Continued)


\[
\begin{array}{cccccc}
  a & a & a & b & b & b \\
  \{A\} & \{A\} & \{A\} & \{S, B\} & \{S, B\} & \{S, B\} \\
  aa & aa & ab & bb & bb & bb \\
  \emptyset & \emptyset & \emptyset & \{C\} & \{C\} & \{C\} \\
  aaa & aab & abb & bbb & bbb & bbb \\
  \emptyset & \emptyset & \{S\} & \emptyset & \emptyset & \emptyset \\
  aaab & aabb & abbb & bbb & bbb \\
  \{C\} & aabb & abbb & bbb & bbb \\
  aabbb & \\
\end{array}
\]
Example (Continued)


<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td>a</td>
<td>b</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>{A}</td>
<td>{A}</td>
<td>{A}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
</tr>
<tr>
<td>aa</td>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>{C}</td>
<td>{C}</td>
<td>{C}</td>
</tr>
<tr>
<td>aaa</td>
<td>aab</td>
<td>abb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>{S}</td>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
</tr>
<tr>
<td>aaab</td>
<td>aabb</td>
<td>abbb</td>
<td>bbb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>{C}</td>
<td>{}</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example (Continued)


\[
\begin{array}{cccccccc}
  a & a & a & b & b & b & b \\
  \{A\} & \{A\} & \{A\} & \{S, B\} & \{S, B\} & \{S, B\} & \{S, B\} \\
  aa & aa & ab & bb & bb & bb & \\
  \emptyset & \emptyset & \emptyset & \{C\} & \{C\} & \{C\} & \\
  aaa & aab & abb & bbb & bbb & \\
  \emptyset & \emptyset & \{S\} & \emptyset & \emptyset & \\
  aaab & aabb & abbb & bbb & \\
  \{C\} & \{\} & \{\} & \\
  aabbb &
\end{array}
\]

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{A}</td>
<td>{A}</td>
<td>{A}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
</tr>
<tr>
<td>aa</td>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>{C}</td>
<td>{C}</td>
<td>{C}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaa</td>
<td>aab</td>
<td>abb</td>
<td>bbb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>{S}</td>
<td>Ø</td>
<td>Ø</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaab</td>
<td>aabb</td>
<td>abbb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{C}</td>
<td>{}</td>
<td>{}</td>
<td>{}</td>
<td>{}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example (Continued)


<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{A}</td>
<td>{A}</td>
<td>{A}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
<td>{S, B}</td>
</tr>
<tr>
<td>aa</td>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>{C}</td>
<td>{C}</td>
<td>{C}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaa</td>
<td>aab</td>
<td>abb</td>
<td>bbb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>{S}</td>
<td>Ø</td>
<td>Ø</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaab</td>
<td>aabb</td>
<td>abbb</td>
<td>bbbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{C}</td>
<td>{}</td>
<td>{}</td>
<td>{}</td>
<td>{}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{S, B}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example

Consider the following CFG $G$ in CNF.

\[
S \rightarrow AB \mid BC \\
A \rightarrow BA \mid a \\
B \rightarrow CC \mid b \\
C \rightarrow AB \mid a
\]

Let us test $baaba$ for membership in $L(G)$.

\[
\begin{array}{cccccc}
\text{b} & \text{a} & \text{a} & \text{b} & \text{a} \\
\{B\} & \text{ba} & \text{aa} & \text{ab} & \text{ba} \\
\text{ba} & \text{aa} & \text{ab} & \text{ba} \\
\text{baa} & \text{aab} & \text{aba} \\
\text{baab} & \text{aaba} \\
\text{baaba}
\end{array}
\]
Example

Consider the following CFG $G$ in CNF.

$$S \rightarrow AB \mid BC$$
$$A \rightarrow BA \mid a$$
$$B \rightarrow CC \mid b$$
$$C \rightarrow AB \mid a$$

Let us test $baaba$ for membership in $L(G)$.

$$\begin{array}{cccccc}
& b & a & a & b & a \\
\{B\} & \{A, C\} & & & & \\
ba & aa & ab & ba \\
baa & aab & aba \\
baab & aaba \\
baaba
\end{array}$$
Example

Consider the following CFG $G$ in CNF.

$$
S \rightarrow AB \mid BC \\
A \rightarrow BA \mid a \\
B \rightarrow CC \mid b \\
C \rightarrow AB \mid a
$$

Let us test $baaba$ for membership in $L(G)$.

```
<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>a</td>
<td>a</td>
<td>b</td>
<td>a</td>
</tr>
<tr>
<td>{B}</td>
<td>{A, C}</td>
<td>{A, C}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ba</td>
<td>aa</td>
<td>ab</td>
<td>ba</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>baa</td>
<td>aab</td>
<td>aba</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>baab</td>
<td>aaba</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>baaba</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
Example

Consider the following CFG $G$ in CNF.

\[
S \rightarrow AB \mid BC \\
A \rightarrow BA \mid a \\
B \rightarrow CC \mid b \\
C \rightarrow AB \mid a
\]

Let us test $baaba$ for membership in $L(G)$.

\[
\begin{array}{ccccccc}
& b & a & a & b & a \\
\{B\} & \{A, C\} & \{A, C\} & \{B\} & \\
ba & aa & ab & ba \\
baa & aab & aba \\
baab & aaba \\
baaba &
\end{array}
\]
Example

Consider the following CFG $G$ in CNF.

\[
S \rightarrow AB \mid BC \\
A \rightarrow BA \mid a \\
B \rightarrow CC \mid b \\
C \rightarrow AB \mid a
\]

Let us test $baaba$ for membership in $L(G)$.

\begin{align*}
& b & a & a & b & a \\
& \{B\} & \{A, C\} & \{A, C\} & \{B\} & \{A, C\} \\
& ba & aa & ab & ba & \\
& baa & aab & aba & \\
& baab & aaba & \\
& baaba
\end{align*}
Example

Consider the following CFG $G$ in CNF.

$$
S \rightarrow AB \mid BC \\
A \rightarrow BA \mid a \\
B \rightarrow CC \mid b \\
C \rightarrow AB \mid a
$$

Let us test $baaba$ for membership in $L(G)$.

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$b$</td>
<td>$a$</td>
<td>$a$</td>
<td>$b$</td>
<td>$a$</td>
<td></td>
</tr>
<tr>
<td>${B}$</td>
<td>${A, C}$</td>
<td>${A, C}$</td>
<td>${B}$</td>
<td>${A, C}$</td>
<td></td>
</tr>
<tr>
<td>$ba$</td>
<td>$aa$</td>
<td>$ab$</td>
<td>$ba$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>${S, A}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$baa$</td>
<td>$aab$</td>
<td>$aba$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$baab$</td>
<td>$aaba$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$baaba$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example

Consider the following CFG $G$ in CNF.

\[
S \rightarrow AB \mid BC \\
A \rightarrow BA \mid a \\
B \rightarrow CC \mid b \\
C \rightarrow AB \mid a
\]

Let us test $baaba$ for membership in $L(G)$.

\[
\begin{array}{cccccc}
b & a & a & b & a \\
\{B\} & \{A, C\} & \{A, C\} & \{B\} & \{A, C\} \\
ba & aa & ab & ba \\
\{S, A\} & \{B\} & \\
baa & aab & aba \\
baab & aaba \\
baaba
\end{array}
\]
Example

Consider the following CFG $G$ in CNF.

$$
\begin{align*}
S & \rightarrow AB \mid BC \\
A & \rightarrow BA \mid a \\
B & \rightarrow CC \mid b \\
C & \rightarrow AB \mid a
\end{align*}
$$

Let us test $baaba$ for membership in $L(G)$.

<table>
<thead>
<tr>
<th>$b$</th>
<th>$a$</th>
<th>$a$</th>
<th>$b$</th>
<th>$a$</th>
</tr>
</thead>
<tbody>
<tr>
<td>${B}$</td>
<td>${A, C}$</td>
<td>${A, C}$</td>
<td>${B}$</td>
<td>${A, C}$</td>
</tr>
<tr>
<td>$ba$</td>
<td>$aa$</td>
<td>$ab$</td>
<td>$ba$</td>
<td></td>
</tr>
<tr>
<td>${S, A}$</td>
<td>${B}$</td>
<td>${S, C}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$baa$</td>
<td>$aab$</td>
<td>$aba$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$baab$</td>
<td>$aaba$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$baaba$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example

Consider the following CFG $G$ in CNF.

$$
S \rightarrow AB \mid BC \\
A \rightarrow BA \mid a \\
B \rightarrow CC \mid b \\
C \rightarrow AB \mid a
$$

Let us test $baaba$ for membership in $L(G)$.

$$
\begin{array}{ccccccc}
  b & a & a & b & a \\
  \{B\} & \{A, C\} & \{A, C\} & \{B\} & \{A, C\} \\
  ba & aa & ab & ba \\
  \{S, A\} & \{B\} & \{S, C\} & \{S, A\} \\
  baa & aab & aba \\
  baab & aaba \\
  baaba
\end{array}
$$
Example

Consider the following CFG $G$ in CNF.

\[
\begin{align*}
S & \rightarrow AB \mid BC \\
A & \rightarrow BA \mid a \\
B & \rightarrow CC \mid b \\
C & \rightarrow AB \mid a
\end{align*}
\]

Let us test $baaba$ for membership in $L(G)$.

\[
\begin{array}{cccccc}
b & a & a & b & a \\
\{B\} & \{A, C\} & \{A, C\} & \{B\} & \{A, C\} \\
ba & aa & ab & ba \\
\{S, A\} & \{B\} & \{S, C\} & \{S, A\} \\
baa & aab & aba \\
\emptyset & \\
baab & aaba \\
\end{array}
\]
Example

Consider the following CFG $G$ in CNF.

$$
S \rightarrow AB \mid BC \\
A \rightarrow BA \mid a \\
B \rightarrow CC \mid b \\
C \rightarrow AB \mid a
$$

Let us test $baaba$ for membership in $L(G)$.

$$
\begin{array}{cccccc}
b & a & a & b & a \\
\{B\} & \{A, C\} & \{A, C\} & \{B\} & \{A, C\} \\
ba & aa & ab & ba & \\
\{S, A\} & \{B\} & \{S, C\} & \{S, A\} & \\
bbaa & aab & aba & \\
\emptyset & \{B\} & & & \\
baab & aaba & & & \\
baaba & \\
\end{array}
$$
Example

Consider the following CFG $G$ in CNF.

$$
S \to AB \mid BC \\
A \to BA \mid a \\
B \to CC \mid b \\
C \to AB \mid a
$$

Let us test $baaba$ for membership in $L(G)$.

$$
\begin{array}{cccc}
b & a & a & b & a \\
\{B\} & \{A, C\} & \{A, C\} & \{B\} & \{A, C\} \\
ba & aa & ab & ba \\
\{S, A\} & \{B\} & \{S, C\} & \{S, A\} \\
baa & aab & aba & baaba \\
\emptyset & \{B\} & \{B\} & \\
baab & aaba & \\
\end{array}
$$
Example

Consider the following CFG $G$ in CNF.

$$
S \rightarrow AB \mid BC \\
A \rightarrow BA \mid a \\
B \rightarrow CC \mid b \\
C \rightarrow AB \mid a
$$

Let us test $baaba$ for membership in $L(G)$.

```
|   |   | |   | |   | |
|---|---|---|---|---|---|
| b | a | a | b | a |
| {B} | {A, C} | {A, C} | {B} | {A, C} |

ba
{S, A}

aa
{B}

ab
{S, C}

ba
{S, A}

baa
Ø

aab
{B}

aba
{B}

baab
Ø

aaba
Ø

baaba
```
Example

Consider the following CFG $G$ in CNF.

$$S \rightarrow AB \mid BC$$

$$A \rightarrow BA \mid a$$

$$B \rightarrow CC \mid b$$

$$C \rightarrow AB \mid a$$

Let us test $baaba$ for membership in $L(G)$.

\[
\begin{array}{cccccc}
  b & a & a & b & a \\
  \{B\} & \{A, C\} & \{A, C\} & \{B\} & \{A, C\} \\
  ba & aa & ab & ba \\
  \{S, A\} & \{B\} & \{S, C\} & \{S, A\} \\
  baa & aab & aba \\
  \emptyset & \{B\} & \{B\} \\
  baab & aaba \\
  \emptyset & \{S, A, C\} \\
  baaba \\
\end{array}
\]
Example

Consider the following CFG $G$ in CNF.

\[
S \rightarrow AB \mid BC \\
A \rightarrow BA \mid a \\
B \rightarrow CC \mid b \\
C \rightarrow AB \mid a
\]

Let us test $baaba$ for membership in $L(G)$.

\[
\begin{array}{cccccc}
b & a & a & b & a \\
\{B\} & \{A, C\} & \{A, C\} & \{B\} & \{A, C\} \\
ba & aa & ab & ba \\
\{S, A\} & \{B\} & \{S, C\} & \{S, A\} \\
baa & aab & aba \\
\emptyset & \{B\} & \{B\} \\
baab & aaba \\
\emptyset & \{S, A, C\} \\
baba & \\
\{S, A, C\}
\end{array}
\]
Example

Kozen, Lecture 27, page 192. \( aabbab \in L(G) \)?

\[
S \rightarrow AB \mid BA \mid SS \mid AC \mid BD \\
A \rightarrow a \\
B \rightarrow b \\
C \rightarrow SB \\
D \rightarrow SA
\]

The grammar is in Chomsky Normal Form.
\[ a \quad a \quad b \quad b \quad a \quad b \]
\[ \{A\} \]
\[ aa \quad ab \quad bb \quad ba \quad ab \]
\[ aab \quad abb \quad bba \quad bab \]
\[ aabb \quad abba \quad bbab \]
\[ aabba \quad abbab \]
\[ aabbbab \]
\[ a \quad a \quad b \quad b \quad a \quad b \]
\[
\{A\} \quad \{A\}
\]

\[ aa \quad ab \quad bb \quad ba \quad ab \]

\[ aab \quad abb \quad bba \quad bab \]

\[ aabb \quad abba \quad bbab \]

\[ aabba \quad abbab \]

\[ aabbbab \]
\[ a \quad a \quad b \quad b \quad a \quad b \]
\[ \{A\} \quad \{A\} \quad \{B\} \]

\[
\begin{array}{cccccc}
aa & ab & bb & ba & ab \\
\end{array}
\]

\[
\begin{array}{cccccc}
aab & abb & bba & bab \\
\end{array}
\]

\[
\begin{array}{cccccc}
aabb & abba & bbab \\
\end{array}
\]

\[
\begin{array}{cccccc}
aabba & abbab \\
\end{array}
\]

\[
\begin{array}{cccccc}
aabbab \\
\end{array}
\]
\begin{align*}
  a & \quad a & \quad b & \quad b & \quad a & \quad b \\
  \{A\} & \quad \{A\} & \quad \{B\} & \quad \{B\} \\
  aa & \quad ab & \quad bb & \quad ba & \quad ab \\
  aab & \quad abb & \quad bba & \quad bab \\
  aabb & \quad abba & \quad bbab \\
  aabba & \quad abbab \\
  aabbab
\end{align*}
$$\begin{array}{cccccc}
a & a & b & b & a & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{A\} & \{B\} \\
\end{array}$$

$$\begin{array}{cccc}
aa & ab & bb & ba & ab \\
\end{array}$$

$$\begin{array}{cccc}
aab & abb & bba & bab \\
\end{array}$$

$$\begin{array}{cccc}
aabb & abba & bbab \\
\end{array}$$

$$\begin{array}{cc}
aabba & abbab \\
\end{array}$$

$$\begin{array}{c}
aabbab \\
\end{array}$$
a  a  b  b  a  b
\{A\}  \{A\}  \{B\}  \{B\}  \{A\}  \{B\}

aa  ab  bb  ba  ab
Ø

aab  abb  bba  bab

aabbb  abba  bbab

aabba  abbbab

aabbab
\[
\begin{align*}
&\{A\} \quad \{A\} \quad \{B\} \quad \{B\} \quad \{A\} \quad \{B\} \\
&aa \quad ab \quad bb \quad ba \quad ab \\
&\Ø \quad \{S\} \\
&aab \quad abb \quad bba \quad bab \\
&aabb \quad abba \quad bbab \\
&aabba \quad abbab \\
&aabbab
\end{align*}
\]
\[
\begin{array}{cccccc}
  a & a & b & b & a & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{A\} & \{B\} \\
  aa & ab & bb & ba & ab \\
\emptyset & \{S\} & \emptyset \\
  aab & abb & bba & bab \\
  aabb & abba & bbbab \\
  aabba & abbab \\
  aabbab \\
\end{array}
\]
\[
\begin{array}{cccccc}
a & a & b & b & a & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{A\} & \{B\} \\
\text{aa} & \text{ab} & \text{bb} & \text{ba} & \text{ab} \\
\emptyset & \{S\} & \emptyset & \{S\} \\
aab & abb & bba & bab \\
aabb & abba & bbab \\
aabba & abbab \\
aabbbab
\end{array}
\]
\[
\begin{array}{cccccc}
a & a & b & b & a & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{A\} & \{B\} \\
aa & ab & bb & ba & ab & \\
\emptyset & \{S\} & \emptyset & \{S\} & \{S\} & \\
aab & abb & bba & bab & \\
aabb & abba & bbab & \\
aabba & abbab & \\
\end{array}
\]
\[
\begin{align*}
\text{a} & \quad \text{a} & \quad \text{b} & \quad \text{b} & \quad \text{a} & \quad \text{b} \\
\{A\} & \quad \{A\} & \quad \{B\} & \quad \{B\} & \quad \{A\} & \quad \{B\} \\
\text{aa} & \quad \text{ab} & \quad \text{bb} & \quad \text{ba} & \quad \text{ab} \\
\emptyset & \quad \{S\} & \quad \emptyset & \quad \{S\} & \quad \{S\} \\
\text{aab} & \quad \text{abb} & \quad \text{bba} & \quad \text{bab} \\
\emptyset & \quad \text{aab} & \quad \text{abba} & \quad \text{bbab} \\
\text{aabb} & \quad \text{abba} & \quad \text{bbab} \\
\text{aabba} & \quad \text{abbab} \\
\text{aabbab}
\end{align*}
\]
<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A}</td>
<td>{A}</td>
<td>{B}</td>
<td>{B}</td>
<td>{A}</td>
<td>{B}</td>
<td></td>
</tr>
<tr>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>ba</td>
<td>ab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td>{S}</td>
<td>Ø</td>
<td>{S}</td>
<td>{S}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aab</td>
<td>abb</td>
<td>bba</td>
<td>bab</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td>{C}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabb</td>
<td>abba</td>
<td>bbab</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

aabba   abbbab

aabbbab
\[ \begin{array}{cccccc}
  a & a & b & b & a & b \\
  \{ A \} & \{ A \} & \{ B \} & \{ B \} & \{ A \} & \{ B \} \\
  aa & ab & bb & ba & ab \\
  \emptyset & \{ S \} & \emptyset & \{ S \} & \{ S \} \\
  aab & abb & bba & bab \\
  \emptyset & \{ C \} & \emptyset \\
  aabb & abba & bbab \\
  aabba & abbab \\
  aabbab 
\end{array} \]
<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{A}</td>
<td>{A}</td>
<td>{B}</td>
<td>{B}</td>
<td>{A}</td>
<td>{B}</td>
</tr>
<tr>
<td></td>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>ba</td>
<td>ab</td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td></td>
<td>{S}</td>
<td>Ø</td>
<td>{S}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aab</td>
<td>abb</td>
<td>bba</td>
<td>bab</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td></td>
<td>{C}</td>
<td>Ø</td>
<td>{C}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabb</td>
<td>abba</td>
<td>bbab</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

aabba abbab

aabbab
\[
\begin{array}{cccccc}
a & a & b & b & a & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{A\} & \{B\} \\
aa & ab & bb & ba & ab & \\
\emptyset & \{S\} & \emptyset & \{S\} & \{S\} & \\
aaab & abbb & bbaa & bab & \\
\emptyset & \{C\} & \emptyset & \{C\} & \\
aabbb & abba & bbab & \\
\{S\} & \\
aabba & abbab & \\
aabbab
\end{array}
\]
\begin{align*}
\begin{array}{cccccc}
a & a & b & b & a & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{A\} & \{B\} \\
\end{array}
\end{align*}

\begin{align*}
\begin{array}{cccc}
aa & ab & bb & ba & ab \\
\emptyset & \{S\} & \emptyset & \{S\} & \{S\} \\
\end{array}
\end{align*}

\begin{align*}
\begin{array}{cccc}
aab & abb & bba & bab \\
\emptyset & \{C\} & \emptyset & \{C\} \\
\end{array}
\end{align*}

\begin{align*}
\begin{array}{ccc}
aabb & abba & bbab \\
\{S\} & \{S\} \\
\end{array}
\end{align*}

\begin{align*}
\begin{array}{cc}
aabba & abbab \\
\end{array}
\end{align*}

\begin{align*}
aabbabb
\end{align*}
<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{A}</td>
<td>{A}</td>
<td>{B}</td>
<td>{B}</td>
<td>{A}</td>
<td>{B}</td>
</tr>
<tr>
<td>aa</td>
<td></td>
<td>ab</td>
<td>bb</td>
<td>ba</td>
<td>ab</td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td></td>
<td>{S}</td>
<td>Ø</td>
<td>{S}</td>
<td>{S}</td>
<td></td>
</tr>
<tr>
<td>aab</td>
<td></td>
<td>abb</td>
<td>bba</td>
<td>bab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td></td>
<td>{C}</td>
<td>Ø</td>
<td>{C}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabb</td>
<td>abba</td>
<td>bbab</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>{S}</td>
<td>{S}</td>
<td>Ø</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabba</td>
<td>abbab</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

aabbbab
\[
\begin{array}{cccccc}
a & a & b & b & a & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{A\} & \{B\} \\

\begin{array}{cccc}
aa & ab & bb & ba & ab \\
\emptyset & \{S\} & \emptyset & \{S\} & \{S\} \\

\end{array}
\end{array}
\]

\[
\begin{array}{cccc}
aab & abb & bba & bab \\
\emptyset & \{C\} & \emptyset & \{C\} \\

\end{array}
\]

\[
\begin{array}{cccc}
aabb & abba & bbab \\
\{S\} & \{S\} & \emptyset \\

\end{array}
\]

\[
\begin{array}{cc}
aabba & abbab \\
\{D\} & \\

\end{array}
\]

\[
aabbab
\]
\[
\begin{array}{cccccc}
  a & a & b & b & a & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{A\} & \{B\} \\
  aa & ab & bb & ba & ab \\
\emptyset & \{S\} & \emptyset & \{S\} & \{S\} \\
  aab & abb & bba & bab \\
\emptyset & \{C\} & \emptyset & \{C\} \\
  aabb & abba & bbab \\
\{S\} & \{S\} & \emptyset \\
  aabba & abbbab \\
\{D\} & \{C\} \\
  aabbab \\
\{S\}
\end{array}
\]
The end. Ignore the rest.
The languages generated by LL(k) grammars is a proper subset of the DCFL accepted by DPDA’s.

Theorem. Linz 6th, §7.4, exercise 6, page 211. If $L$ is generated by a LL($k$) grammar for some $k$, then $L \in$ DCFL.

Let $L$ be the language $a^n \cup a^n b^n$. Let $L$ be the language $a^n c b^n \cup a^n d b^{2n}$.\(^1\)

Theorem. $L$ is accepted by a DPDA.

Theorem. For no $k$ is there an LL($k$) grammar generating $L$.

\(^1\)Lehtinen and Okhotin, LNCS, 2012, page 156.
Theorem 11.6, Linz, 6th, page 293. Any language generated by an unrestricted grammar is recursively enumerable.

Kozen, Miscellaneous Exercise, #104, page 343. Show that the type 0 grammars (see Lecture 36) generate exactly the r.e. sets.
Example 11.1

\[ S \]
\[ \Rightarrow SV \]
\[ \Rightarrow TV \]
\[ \Rightarrow TVa V \]
\[ \Rightarrow Vaa V_{a0} V \]
\[ \Rightarrow Vaa V_{a0} V_{\;0} \]
\[ \Rightarrow Vaa V_{a1} V \]
\[ \Rightarrow Vaa a V \]
\[ \Rightarrow Vaa a \]
\[ \Rightarrow aa \]

\[ S \rightarrow SV \]  \hspace{1cm} 1 \hspace{1cm} (11.6) \hspace{1cm} add trailing blanks
\[ S \rightarrow T \]  \hspace{1cm} 2 \hspace{1cm} (11.6) \hspace{1cm} generate input
\[ T \rightarrow TVa \]  \hspace{1cm} 3 \hspace{1cm} (11.7) \hspace{1cm} an input character
\[ T \rightarrow TVa \]  \hspace{1cm} 4 \hspace{1cm} (11.7) \hspace{1cm} start state, 1st character
\[ V_{a0} V_{aa} \rightarrow V_{aa} V_{a0} \]  \hspace{1cm} 5 \hspace{1cm} (11.8) \hspace{1cm} \langle q_0, a, q_0, a, R \rangle \in \Delta
\[ V_{a0} V_{aa} \rightarrow V_{aa} V_{a0} \]  \hspace{1cm} 6 \hspace{1cm} (11.8) \hspace{1cm} \langle q_0, a, q_0, a, R \rangle \in \Delta
\[ V_{a1} \rightarrow a \]  \hspace{1cm} 7 \hspace{1cm} (11.9) \hspace{1cm} \langle q_0, a, q_1, a, L \rangle \in \Delta
\[ a V \rightarrow a \]  \hspace{1cm} 8 \hspace{1cm} (11.10) \hspace{1cm} restore input
\[ V_{aa} a \rightarrow a \]  \hspace{1cm} 9 \hspace{1cm} (11.11) \hspace{1cm} restore input
\[ V_{aa} a \rightarrow a \]  \hspace{1cm} 10 \hspace{1cm} (11.12) \hspace{1cm} restore input
\[ \rightarrow \epsilon \]  \hspace{1cm} 11 \hspace{1cm} (11.13) \hspace{1cm} restore input
Example 11.1

\[
S \Rightarrow SV_{\underline{u}} \\
\Rightarrow TV_{\underline{u}} \\
\Rightarrow TV_{aa}V_{\underline{u}} \\
\Rightarrow V_{a0a}V_{aa}V_{\underline{u}} \\
\Rightarrow V_{aa}V_{a0a}V_{\underline{u}} \\
\Rightarrow V_{aa}V_{aa}V_{\underline{u}_{0}} \\
\Rightarrow V_{aa}V_{a1a}V_{\underline{u}} \\
\Rightarrow V_{aa}aV_{\underline{u}} \\
\Rightarrow V_{aa}a_{\underline{u}} \\
\Rightarrow aa_{\underline{u}} \\
\Rightarrow aa \\
\]

\[
S \rightarrow SV_{\underline{u}} \quad (11.6)
\]

\[
\vdash \langle \epsilon, q_0, aa \rangle \\
\vdash \langle a, q_0, a \rangle \\
\vdash \langle aa, q_0, u \rangle \\
\vdash \langle a, q_1, a_{\underline{u}} \rangle \\
\vdash \langle aa, a_{\underline{u}} \rangle \\
\vdash \langle aa \rangle \\
\vdash aa \\
\]

\[
\underline{u} \rightarrow \epsilon \quad (11.13)
\]
