A formal language is a set of strings over an alphabet.
Summary of Four Paradigms of Formal Languages

- Ad-hoc mathematical descriptions \( \{ w \in \Sigma^* \mid \ldots \} \).

1. An automaton \( M \) accepts or recognizes 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 \( L(P) \).
A language may be described mathematically using set notation as \( L = \{ w \in \Sigma^* \mid P(w) \} \).

This means \( w \in L \) if, and only, if \( P(w) \) is true.

A language may be accepted by an automaton \( M \).

\[
L(M) = \{ w \in \Sigma^* \mid d_0 \vdash^* d_f \text{ where } d_f \text{ is final state} \}
\]

A language may be denoted by an expression \( x \). We write \( L = D[x] \). A language may be denoted by an expression \( x \). We write \( L = L[x] \).

A language may be generated by a grammar \( G \).

\[
L(G) = \{ w \in \Sigma^* \mid S \Rightarrow^* w \}
\]
Automata compute

Expressions denote

Trees demonstrate

Grammars construct
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 $\alpha \rightarrow \beta$ 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.
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 \( \alpha \rightarrow \beta \) 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.
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;
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.
Example

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

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. $S \rightarrow 1 S 1$
To more effectively communicate and 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 more effectively communicate and 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. \(\ldots, X, Y, Z\) are terminal or nonterminal symbols.
4. \(\ldots, w, x, y, z\) are strings/sequences of terminals only.
5. \(\alpha, \beta, \gamma, \ldots\) are strings/sequences 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 $A \rightarrow \alpha_1 \mid \alpha_2 \mid \cdots \mid \alpha_n$.
Compact Notation

It is convenient to think of a production [in a CFG] as “belonging” to the variable [nonterminal] of its head [LHS]. We shall often use remarks like “the productions for A” or “A-productions” to refer to the production whose head [LHS] is [the] variable A. We may write the productions for a grammar by listing each variable [nonterminal] once, and then listing all the bodies of the productions for that variable [nonterminal], separated by vertical bars.

HMU 3rd, §5.1, page 175.
Derivation

[cf. Kozen, page 133]

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 we 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 $\delta \rightarrow \gamma$ is a production of $G$, then $\alpha_1\delta\alpha_2$ derives $\alpha_1\gamma\alpha_2$ in one step, and we write

$$\alpha_1\delta\alpha_2 \xrightarrow{1}_G \alpha_1\gamma\alpha_2$$

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

Definition. Let $\Rightarrow^*_G$ be the reflexive, transitive closure of the $\Rightarrow^1_G$ relation. That is:

\[
\alpha \Rightarrow^* \alpha \quad \text{for any } \alpha \\
\alpha \Rightarrow^* \beta \quad \text{if } \alpha \Rightarrow^* \gamma \text{ and } \gamma \Rightarrow^1 \beta
\]

This relation is sometimes called the “derives in zero or more steps” relation.
Derivation – Inductively Defined Relation

\[ \delta \rightarrow \gamma \in P \]

\[ \alpha_1 \delta \alpha_2 \xrightarrow{1} \alpha_1 \gamma \alpha_2 \]

\[ \alpha \in (T \cup V)^* \]

\[ \alpha \xrightarrow{*} \alpha \]

\[ \alpha \Rightarrow \gamma \quad \gamma \xrightarrow{1} \beta \]

\[ \alpha \xrightarrow{*} \beta \]
A string in \((T \cup V)^*\) that is derivable from the start symbol \(S\) is called a *sentential form*. A sentential form is called a *sentence*, if it consists only of terminal symbols.

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

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

1. Different Kinds of Grammars

2. Context-Free Grammars
   - Leftmost, Rightmost Derivations
   - Ambiguity

3. Chomsky and Greibach Normal Forms

4. Cleaning Up CFG's

5. Brute Force Membership Test

6. CYK Algorithm
Grammars

context-free
linear
regular
left linear
right linear

S → SS
S → aSb
S → bSa
S → ε
S → aS | ε
S → Sa | ε
S → aSb | ε

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 (Sudkamp)

- Context-sensitive grammar [Not defined in HMU].
  - Greibach normal form (Sudkamp, §4.8, definition 4.8.1, page 131).
  - simple-grammar, s-grammar [Not defined in HMU].
- Chomsky normal form (Sudkamp, §4.5, definition 4.5.1, page 122).
- Linear grammar (Sudkamp, chapter 7, exercise 22, page 250).
  - Regular grammar (Sudkamp, page 81).
    - right-linear grammar [Sudkamp, chapter 3,, exercise 40, page 102].
    - 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, either:
      - 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

If $\epsilon \in L(G)$ we must allow a niggly exception $S \rightarrow \epsilon$. 
The s-languages are those languages recognized by a particular restricted form of
deterministic pushdown automaton, called an s-machine. They are uniquely charac-
terized by that subset of the standard-form grammars in which each rule has the form
\[ Z \rightarrow aY_1 \ldots Y_n, \quad n \geq 0, \]
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 \quad A \to xB \text{ or } A \to x \quad A, B \in V, x \in T^*

strongly right-linear \quad A \to aB \text{ or } A \to \epsilon \quad A, B \in V, a \in T

left-linear \quad A \to Bx \text{ or } A \to x \quad A, B \in V, x \in T^*

strongly left-linear \quad A \to Ba \text{ or } A \to \epsilon \quad A, B \in V, a \in T
Outline

1 Different Kinds of Grammars

2 Context-Free Grammars
   Leftmost, Rightmost Derivations
   Ambiguity

3 Chomsky and Greibach Normal Forms

4 Cleaning Up CFG’s

5 Brute Force Membership Test

6 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; HUM 3rd, §7.4.2, page 301).
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.

Linz 6th, §5.1, definition 5.1, page 130.
HMU 3rd, §5.1.2, page 173.
Sudkamp, §3.1, definition 3.1.3, page 70.
Definition. A subset $L \subseteq T^*$ is a context-free language (CFL) if $L = L(G)$ for some context-free grammar $G$. 
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.
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 we 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 \( A \rightarrow \gamma \) 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 $\not{\Rightarrow}_G^*$ be the reflexive, transitive closure of the $\not{\Rightarrow}_G^1$ relation. That is:

- $\alpha \not{\Rightarrow}_G^* \alpha$ for any $\alpha$
- $\alpha \not{\Rightarrow}_G^* \beta$ if $\alpha \not{\Rightarrow}_G^* \gamma$ and $\gamma \not{\Rightarrow}_G^1 \beta$

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 $G$:

\[
S \xrightarrow{1} G aSa \xrightarrow{1} G aaSaa \xrightarrow{1} G aabSbaa \xrightarrow{1} G aabbaa
\]

\[
S \xrightarrow{1} G bSb \xrightarrow{1} G baSab \xrightarrow{1} G babSbab \xrightarrow{1} G babaSbab \xrightarrow{1} G babaabab
\]

Since $S \notRightarrow aabbaa$ and $S \notRightarrow babaabab$, 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 $\alpha \in (T \cup V)^*$. 
Example

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 \( R \). \textit{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.
<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>$#_L[\text{alpha}]$</th>
<th>$#_R[\text{alpha}]$</th>
<th>$#[\alpha]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varepsilon$</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>)</td>
<td>0</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>$R$</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$B$</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$R(R$</td>
<td>3</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>$RRR$</td>
<td>3</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>$RRRB$</td>
<td>3</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>$))))B$</td>
<td>0</td>
<td>3</td>
<td>-1</td>
</tr>
</tbody>
</table>
We capture some (but not all) of the notion what it means for an expression to use parentheses correctly. Theorem. For all $B \Rightarrow^* \alpha$, $\#[\alpha] = 0$.

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. $\#[\alpha R \beta] = \#[\alpha ) \beta]$
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>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( B )</td>
<td>((0+0)-0)</td>
<td>0</td>
<td>( B )</td>
<td>((0+0)-0)</td>
<td>0</td>
</tr>
<tr>
<td>( (RB)</td>
<td>((1+0)-1)</td>
<td>0</td>
<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>
<td>( ())RB)</td>
<td>((2+0)-2)</td>
<td>0</td>
</tr>
<tr>
<td>( R()B)</td>
<td>((1+1)-2)</td>
<td>0</td>
<td>( ())B)</td>
<td>((1+1)-2)</td>
<td>0</td>
</tr>
<tr>
<td>( R())</td>
<td>((1+1)-2)</td>
<td>0</td>
<td>( ()))</td>
<td>((0+2)-2)</td>
<td>0</td>
</tr>
<tr>
<td>( ()())</td>
<td>((0+2)-2)</td>
<td>0</td>
<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
\[ L = \{ ab(bbaa)^n bba(ba)^n | 0 \leq n \} \text{ is CFL} \]

Linz 6th, Section 5.1, Example 5.3, page 132
\[ L = \{ a^n b^m | n \neq m \} \text{ is CFL} \]

Linz 6th, Section 5.1, Example 5.4, page 133
\[ L = \{ w | \forall u, v \; uv = w\#([u] \geq \#)[v], \#([uv] = \#)[uv] \} \text{ is CFL} \]
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:

\[
\begin{align*}
S & \rightarrow AS_1 \mid S_1 B \\
S_1 & \rightarrow aS_1 b \mid \epsilon \\
A & \rightarrow aA \mid a \\
B & \rightarrow bB \mid b
\end{align*}
\]

Since \( L = L(G) \), \( L \) is a context-free language.
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
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

\[ \alpha \Rightarrow_{lm}^* \beta \]

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

\[ \alpha \Rightarrow_{rm}^* \beta \]
A step in a derivation will be both leftmost and rightmost if the sentential form only has one non-terminal to begin with.
Hence, a derivation can be both leftmost and rightmost at the same time.
A derivation can be neither leftmost nor rightmost. This is the case when some steps are leftmost (but not rightmost) and some steps are rightmost (but not leftmost).
Derivation

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 \( abbb \)):

\[
S \xrightarrow[1]{l} aAB \xrightarrow[1]{l} abBbB \xrightarrow[1]{l} abAbB \xrightarrow[1]{l} abbbBbB \xrightarrow[1]{l} abbbB \xrightarrow[1]{l} abbb
\]

\[
S \xrightarrow[1]{l} aAB \xrightarrow[1]{l} abBbB \xrightarrow[1]{r} abBb \xrightarrow[1]{l} abAb \xrightarrow[1]{l} abbbBb \xrightarrow[1]{l} abbb
\]

\[
S \xrightarrow[1]{r} aAB \xrightarrow[1]{r} aA \xrightarrow[1]{l} abBb \xrightarrow[1]{l} abAb \xrightarrow[1]{l} abbbBb \xrightarrow[1]{l} abbb
\]
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.
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
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  -- == abbbb
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
Linz 6th, section 5.2, definition 5.5, page 146.

Definition. A context-free language $L$ is inherently ambiguous, if there is no unambiguous context-free grammar $G$ for which $L = L(G)$. 
\[ L = \{a^n b^n c^m\} \cup \{a^n b^m c^m\} \]

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

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

This grammar is ambiguous as \( S_1 \Rightarrow^* abc \) and \( S_2 \Rightarrow^* abc \). But this does 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

1. Different Kinds of Grammars

2. Context-Free Grammars
   - Leftmost, Rightmost Derivations
   - Ambiguity

3. Chomsky and Greibach Normal Forms

4. Cleaning Up CFG’s

5. Brute Force Membership Test

6. 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
   Sudkamp 3rd, §4.5, definition 4.5.1, page 122

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
   HUM 3rd, §7.1, page 277
   Kozen, Lecture 21, definition 21.1, page 140
   Sudkamp 3rd, §4.8, definition 4.8.1, page 131

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.
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

1 Different Kinds of Grammars

2 Context-Free Grammars
   - Leftmost, Rightmost Derivations
   - Ambiguity

3 Chomsky and Greibach Normal Forms

4 Cleaning Up CFG's

5 Brute Force Membership Test

6 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.
Cleaning Up CFG's

We must get rid of all \( \varepsilon \)-productions \( A \to \varepsilon \). and unit productions \( A \to B \). 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

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

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

\begin{align*}
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
\end{align*}
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 is 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 $\epsilon$. 
A Useful Substitution Rule


Theorem. For any grammar $G = \langle T, V, P, S \rangle$, and for any nonterminal $B \in V$ such that $B \Rightarrow^*_G \gamma$, 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.

Do not overlook the special case in which $\gamma = \epsilon$ and so if there is a production $B \rightarrow \epsilon$, then there is shortcut $A \rightarrow \alpha \beta$.
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.
\[ V_n := \emptyset; \quad \text{-- assume nothing is nullable} \]

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

Compute the set \( V_o \) of all nullable nonterminals
nullableRHS :: Eq a => [a] -> [a] -> Bool
nullableRHS set rhs = all ('elem' set) rhs

close :: [Char] -> [Char]
close set |
| set == next = set |
| otherwise = close next
where next = nub $ set ++
  [non | (non, rhs) <- grammar, nullableRHS set rhs]
Algorithm 4.2.1
Construction of the Set of Nullable Variables

input: context-free grammar $G = (V, \Sigma, P, S)$

1. $\text{NULL} := \{ A \mid A \rightarrow \lambda \in P \}$
2. repeat
   
   2.1. $\text{PREV} := \text{NULL}$
   
   2.2. for each variable $A \in V$ do
       
       if there is an $A$ rule $A \rightarrow w$ and $w \in \text{PREV}^*$, then
       
       $\text{NULL} := \text{NULL} \cup \{ A \}$
       
   until $\text{NULL} = \text{PREV}$
The set \textsc{Nullable} is an inductively defined set.

\[
\begin{array}{c}
A \rightarrow \epsilon \in P \\
A \in \textsc{Nullable}
\end{array} \quad \quad \\
\begin{array}{c}
A \rightarrow \alpha \in P \\
A \in \textsc{Nullable}
\end{array} \quad \quad \\
\begin{array}{c}
A \rightarrow \alpha \in \textsc{Nullable}^* \\
A \in \textsc{Nullable}
\end{array}
\]
Figure 5.4: An algorithm for testing which variables in a CFG derive $\Lambda$. 

Floyd, 1994, Section 5.3, page 338.
\( V_n := \{S\}; \quad -- \text{Assume just } S \text{ is reachable} \)

\[
\begin{align*}
\text{loop} & \quad V_o := V_n; \\
& \text{for } X \rightarrow \alpha \in P \text{ loop} \\
& \quad \text{if } X \in V_n \text{ then} \\
& \quad \quad \text{for } N \in V \text{ in } \alpha \text{ loop} \\
& \quad \quad \quad \text{add } N \text{ to } V_n; \\
& \quad \quad \text{end loop}; \\
& \quad \text{end if;} \\
& \text{end loop;} \\
& \text{exit when } V_n = V_o \\
\text{end loop;}
\]

Compute the set of all reachable nonterminals
The set $\text{REACHABLE}$ is an inductively defined set.

\[
\begin{array}{c}
\frac{S \in \text{REACHABLE}}{
\frac{A \rightarrow \alpha B \beta \in P \quad A \in \text{REACHABLE}}{B \in \text{REACHABLE}}
}
\end{array}
\]
\[ V_n := \emptyset; \quad -- \text{assume nothing is productive} \]

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

Compute the set of all productive nonterminals
Algorithm 4.4.2
Construction of the Set of Variables That Derive Terminal Strings

input: context-free grammar $G = (V, \Sigma, P, S)$

1. $TERM := \{A \mid \text{there is a rule } A \rightarrow w \in P \text{ with } w \in \Sigma^*\}$
2. repeat
   2.1. $PREV := TERM$
   2.2. for each variable $A \in V$ do
       if there is an $A$ rule $A \rightarrow w$ and $w \in (PREV \cup \Sigma)^*$ then
           $TERM := TERM \cup \{A\}$
   until $PREV = TERM$
Additional and Pointless Algorithm

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

\text{loop}
\quad Z_o := Z_n ;
\quad \text{for} \ A \rightarrow B \in P \ \text{loop} \quad -- \quad \text{unit production}
\quad \quad \text{for} \ \langle B, C \rangle \in Z_n \ \text{loop} \quad -- \quad B \Rightarrow C
\quad \quad \quad \text{add} \ \langle A, C \rangle \ \text{to} \ Z_n ;
\quad \quad \text{end loop} ;
\quad \text{end loop} ;
\quad \text{exit when} \ Z_n = Z_o ;
\text{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).
\( P_n := P; \quad -- \text{keep all the original productions} \)

\begin{verbatim}
loop
  \( P_o := P_n; \quad -- \text{remember previous productions} \)
  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 \( P_o = P_n; \quad -- \text{repeat until no changes} \)
end loop;
\end{verbatim}

Make \( \epsilon \) and unit productions superfluous

Slightly more explicit version
\[ P_n := P; \quad -- \, \text{keep all the original productions} \]
\begin{verbatim}
loop
    \[ P_o := P_n; \quad -- \, \text{remember previous productions} \]
    for \( X \rightarrow \alpha N \beta \in P_n \) loop
        add \( X \rightarrow \alpha \beta \) to \( P_n \) if \( N \rightarrow \epsilon \in P_n \);
    end loop;
    for \( X \rightarrow B \in P_n \) (where \( B \in V \)) loop
        add \( X \rightarrow \beta \) to \( P_n \) if \( B \rightarrow \beta \in P_n \);
    end loop;
    exit when \( P_o = P_n \); \quad -- \, \text{repeat until no changes} 
end loop;
\end{verbatim}

Make \( \epsilon \) and unit productions superfluous
1 Examples first.
2 Proof of correctness afterward.
Remove the $\epsilon$ and unit productions from the following grammar:

- $S \rightarrow AB$
- $A \rightarrow aAA \mid \epsilon$
- $B \rightarrow bBB \mid \epsilon$

- $S \rightarrow A \mid B \mid AB$
- $A \rightarrow a \mid aA \mid aAA \mid \epsilon$
- $B \rightarrow b \mid bB \mid bBB \mid \epsilon$

- $S \rightarrow A \mid B \mid AB$
- $S \rightarrow a \mid aA \mid aAA$
- $S \rightarrow b \mid bB \mid bBB$
- $A \rightarrow a \mid aA \mid aAA \mid \epsilon$
- $B \rightarrow b \mid bB \mid bBB \mid \epsilon$

- $S \rightarrow AB$
- $S \rightarrow a \mid aA \mid aAA$
- $S \rightarrow b \mid bB \mid bBB$
- $A \rightarrow a \mid aA \mid aAA$
- $B \rightarrow b \mid bB \mid bBB$
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 \textit{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: $S \rightarrow \epsilon$. 
Chomsky Normal Form

Theorem. Every context free grammar can be put in Chomsky Normal Form. In other words, for every context grammar $G$, there is a context free grammar $G'$ in Chomsky Normal Form such that $L(G) = L(G')$

HMU 3rd, §7.1.5, page 272.
Kozen, Lecture 21, page 140.
First, eliminate $\epsilon$ productions and unit productions. One might as well eliminate useless nonterminals (and their 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.
Algorithm to Put a Grammar in CNF

For productions of length $k$ greater than two $X \rightarrow A_1 A_2 \ldots A_k$, add a cascade of $k - 2$ new nonterminals and productions.

\[
\begin{align*}
X & \rightarrow B_1 X_1 \\
X_1 & \rightarrow B_2 X_2 \\
X_2 & \rightarrow B_3 X_3 \\
& \quad \vdots \\
X_{k-3} & \rightarrow B_{k-2} X_{k-2} \\
X_{k-2} & \rightarrow B_{k-1} B_k
\end{align*}
\]
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.

\[
S \rightarrow [S] \mid SS \mid \epsilon
\]

\[
S \rightarrow ASB \mid SS \mid AB, \quad A \rightarrow [, \quad B \rightarrow ]
\]

\[
S \rightarrow AC \mid SS, \quad A \rightarrow [, \quad B \rightarrow ], \quad C \rightarrow SB
\]
Put in Chomsky Normal Form:

\[
\begin{align*}
S & \rightarrow ASA | aB \\
A & \rightarrow B | S \\
B & \rightarrow b | \epsilon
\end{align*}
\]

\[
B \rightarrow \epsilon
\]

\[
\begin{align*}
S & \rightarrow ASA | aB | a \\
A & \rightarrow B | S | \epsilon \\
B & \rightarrow b | \epsilon
\end{align*}
\]

\[
A \rightarrow \epsilon
\]

\[
\begin{align*}
S & \rightarrow ASA | aB | a | AS | SA \\
A & \rightarrow B | S | \epsilon \\
B & \rightarrow b | \epsilon
\end{align*}
\]
$A \rightarrow B$ and $A \rightarrow S$

$S \rightarrow ASA \mid aB \mid a \mid AS \mid SA$
$S \rightarrow BSB \mid aB \mid a \mid BS \mid SB$
$S \rightarrow SSS \mid aB \mid a \mid SS \mid SS$
$A \rightarrow B \mid S \mid \epsilon$
$B \rightarrow b \mid \epsilon$

Does not new unit rules like $S \rightarrow \epsilon$ matter?

$S \rightarrow ASA \mid aB \mid a \mid AS \mid SA$
$S \rightarrow BSB \mid aB \mid a \mid BS \mid SB$
$S \rightarrow SSS \mid aB \mid a \mid SS$
$B \rightarrow b$

Do we now have useless nonterminals?
Outline

1. Different Kinds of Grammars

2. Context-Free Grammars
   - Leftmost, Rightmost Derivations
   - Ambiguity

3. Chomsky and Greibach Normal Forms

4. Cleaning Up CFG's

5. Brute Force Membership Test

6. 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 $\beta \Rightarrow^1 G \gamma$, 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 $\#_+(w) = 2|w|$. 
Example of Exhaustive Search

Find a derivation of $aabb$ in the grammar $S \rightarrow SS \mid aSb \mid bSa \mid \epsilon$. 
Outline

1. Different Kinds of Grammars
2. Context-Free Grammars
   - Leftmost, Rightmost Derivations
   - Ambiguity
3. Chomsky and Greibach Normal Forms
4. Cleaning Up CFG’s
5. Brute Force Membership Test
6. CYK Algorithm
Cocke-Younger-Kasami Algorithm (CYK)

Linz 6th, section 6.3, page 178
Busch’s notes, class10 cf, page 22ff
HMU 3rd, section 7.4.4, page 303
Kozen, Lecture 27, page 191
Floyd, Section 5.11, Figure 5.20, page 390.

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 $A \Rightarrow^* s[i : j]$.

A classic dynamic programming algorithm.
## 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>s[0 : i]</td>
<td>s[0 : n − 1]</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>r</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>s[r : i]</td>
<td>s[r : n − 1]</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>s[n − 1 : n − 1]</td>
<td></td>
</tr>
</tbody>
</table>
for $i$ in $0, \ldots, n-1$ loop
    $M[i, i] := \emptyset$
    for $N \to a$ in $P$ loop
        add $N$ to $M[i, i]$ if $a = a_i$ -- $N \Rightarrow^* s[i : i]$
    end loop
end loop
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
      -- \( s[r : r+g] = s[r : m] \, \text{++} \, s[m+1 : r+g] \)
      -- 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 \( B \in L \text{ and } C \in R \)
      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
        $T_{i,i+m} := \emptyset$; /* 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

$N \leftarrow \text{LENGTH}(\text{words})$

$M \leftarrow$ the number of nonterminal symbols in grammar

$P \leftarrow$ 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 ($X \rightarrow \text{words}_i \ [p]$) do

    $P[X, i, 1] \leftarrow p$

  /* 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 $N - \text{length} + 1$ do

      for len1 = 1 to $N - 1$ do

        len2 $\leftarrow$ length $-$ len1

        for each rule of the form ($X \rightarrow Y \ Z \ [p]$) do

          $P[X, \text{start}, \text{length}] \leftarrow \max(P[X, \text{start}, \text{length}],$

          $P[Y, \text{start}, \text{len1}] \times P[Z, \text{start} + \text{len1}, \text{len2}] \times p)$

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 $P[X, \text{start}, \text{len}]$ is the probability of the most probable $X$ of length $\text{len}$ starting at position $\text{start}$. 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. Use the CYK method to determine if the string \( w = aabbb \) is in the language generated by the following grammar \( G \) in Chomsky Normal Form.

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

\[
\begin{array}{cccccc}
\{A\} \\
aa & ab & bb & bb \\
aab & abbb & bbb \\
aabb & abbb
\end{array}
\]

Since \( S \in M[0,4] \), the string \( w = aabbb \) is in \( L(G) \).
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string \( w = aabbb \) is in the language generated by the following grammar \( G \) in Chomsky Normal Form.

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

Since \( S \in M[0,4] \), the string \( w = aabbb \) is in \( L(G) \).
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string \( w =aabbb \) is in the language generated by the following grammar \( G \) in Chomsky Normal Form.

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

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

Since \( S \in M[0,4] \), the string \( w =aabbb \) is in \( L(G) \).
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string $w = aabbb$ is in the language generated by the following grammar $G$ in Chomsky Normal Form.

$$
S \rightarrow AB \\
A \rightarrow BB \mid a \\
B \rightarrow AB \mid b
$$

Since $S \in M[0,4]$, the string $w = aabbb$ is in $L(G)$. 

\[
\begin{array}{cccccc}
\{A\} & \{A\} & \{B\} & \{B\} \\
aa & ab & bb & bb \\
 aab & abb & bbb \\
aabb & abbb
\end{array}
\]
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string \( w = aabbb \) is in the language generated by the following grammar \( G \) in Chomsky Normal Form.

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

\[
\begin{array}{ccccccc}
 & a & a & b & b & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{B\} \\
aa & ab & bb & bb \\
\end{array}
\]

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

Since \( S \in M[0, 4] \), the string \( w = aabbb \) is in \( L(G) \).
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string \( w = aabbb \) is in the language generated by the following grammar \( G \) in Chomsky Normal Form.

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

\[
\begin{array}{cccccc}
  a & a & b & b & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{B\} \\
  aa & ab & bb & bb \\
\emptyset & \\
  aab & abb & bbb \\
  aabb & abbb
\end{array}
\]

Since \( S \in M[0,4] \), the string \( w = aabbb \) is in \( L(G) \).
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string $w = aabbb$ is in the language generated by the following grammar $G$ in Chomsky Normal Form.

$$S \rightarrow AB$$
$$A \rightarrow BB \mid a$$
$$B \rightarrow AB \mid b$$

| $a$ | $a$ | $b$ | $b$ | $b$ |
| {A} | {A} | {B} | {B} | {B} |
| aa | ab | bb | bb |
| Ø | {S, B} |
| aab | abb | bbb |
| aabb | abbb |

Since $S \in M[0, 4]$, the string $w = aabbb$ is in $L(G)$. 

Ryan Stansifer (CS, Florida Tech)
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string \( w = aabbb \) is in the language generated by the following grammar \( G \) in Chomsky Normal Form.

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

Since \( S \in M[0, 4] \), the string \( w = aabbb \) is in \( L(G) \).
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string $w = aabbb$ is in the language generated by the following grammar $G$ in Chomsky Normal Form.

$$S \rightarrow AB$$
$$A \rightarrow BB \mid a$$
$$B \rightarrow AB \mid b$$

Since $S \in M[0, 4]$, the string $w = aabbb$ is in $L(G)$.
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string \( w = aabbb \) is in the language generated by the following grammar \( G \) in Chomsky Normal Form.

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

Since \( S \in M[0,4] \), the string \( w = aabbb \) is in \( L(G) \).
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string $w = aabbb$ is in the language generated by the following grammar $G$ in Chomsky Normal Form.

$$
S \rightarrow AB \\
A \rightarrow BB \mid a \\
B \rightarrow AB \mid b
$$

Since $S \in M[0,4]$, the string $w = aabbb$ is in $L(G)$. 

```
   a     a     b     b     b
  {A}    {A}    {B}    {B}    {B}
  aa     ab     bb     bb
  Ø      {S, B}  {A}    {A}
 aab    abb     bbb
  {S, B}  {A}
 aabb    abbb
```
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string $w = aabbb$ is in the language generated by the following grammar $G$ in Chomsky Normal Form.

$$
S \to AB \\
A \to BB \mid a \\
B \to AB \mid b
$$

<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></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>{A}</td>
<td>{S, B}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabb</td>
<td>abbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Since $S \in M[0, 4]$, the string $w = aabbb$ is in $L(G)$. 

Ryan Stansifer (CS, Florida Tech)
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string \( w = aabbb \) is in the language generated by the following grammar \( G \) in Chomsky Normal Form.

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

Since \( S \in M[0,4] \), the string \( w = aabbb \) is in \( L(G) \).
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string \( w = aabbb \) is in the language generated by the following grammar \( G \) in Chomsky Normal Form.

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

Since \( S \in \mathcal{M}_{[0,4]} \), the string \( w = aabbb \) is in \( L(G) \).
Example

Linz 6th, §6.3, example 6.11, page 179. Use the CYK method to determine if the string \( w = aabbb \) is in the language generated by the following grammar \( G \) in Chomsky Normal Form.

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

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

Since \( S \in M [0, 4] \), the string \( w = aabbb \) is in \( L(G) \).
Example

Linz 6th, §6.3, exercise 4, page 180 (solution page 421). Use the CYK method to determine if the string \( w = aaabbbb \) is in the language generated by the grammar \( S \rightarrow aSb | b \).

First, convert the grammar to CNF.

\[
S \rightarrow AC | b \\
C \rightarrow SB \\
A \rightarrow a \\
B \rightarrow b
\]
Example (Continued)


\[
\begin{array}{cccccc}
  a & a & a & b & b & b \\
  \{A\} & & & & & \\
  aa & aa & ab & bb & bb & bb \\
  aaa & aab & abb & bbb & bbb & \\
  aaab & aabb & abbb & bbb & \\
  aaabb & aabb & abbb & bbb & \\
  aaabbb & aabbb & bbbb & & \\
  aaabbbb & aabbbb & & & \\
\end{array}
\]
**Example (Continued)**


<table>
<thead>
<tr>
<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></td>
<td></td>
<td></td>
<td></td>
<td></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>a</td>
<td>aabb</td>
<td>abb</td>
<td>bbb</td>
<td>bbb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabb</td>
<td>aabb</td>
<td>abbb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaabb</td>
<td>aabbb</td>
<td>abbb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaabbb</td>
<td>aabbbb</td>
<td>aabbbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example (Continued)


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


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


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


<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>b</th>
<th></th>
<th></th>
<th>b</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>
<td></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></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th>b</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>aa</td>
<td>aa</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td></td>
</tr>
</tbody>
</table>

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

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

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>aaabb</td>
<td>aabbb</td>
<td>bbbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>aaabbb</td>
<td>aabbbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Ryan Stansifer (CS, Florida Tech)
Example (Continued)


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


<table>
<thead>
<tr>
<th></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>
</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></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaaa</td>
<td>aab</td>
<td>abb</td>
<td>bbb</td>
<td>bbb</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>aabb</td>
<td>abbb</td>
<td>bbbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaabb</td>
<td>aabb</td>
<td>abbb</td>
<td>bbbb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaabb</td>
<td>aabbb</td>
<td>bbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaabbb</td>
<td>aabbbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabbbb</td>
<td>aabbbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<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>
</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>Ø</td>
<td>Ø</td>
<td>ab</td>
<td>bb</td>
<td>bb</td>
<td>bb</td>
<td>bb</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>aabb</td>
<td>aabbb</td>
<td>abbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
</tr>
<tr>
<td>aaabb</td>
<td>aabbb</td>
<td>bbbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
</tr>
<tr>
<td>aaabb</td>
<td>aabbbb</td>
<td>bbbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
</tr>
<tr>
<td>aaabbb</td>
<td>aabbbb</td>
<td>bbbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
</tr>
</tbody>
</table>
Example (Continued)


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


\[
\begin{array}{cccccccc}
a & a & a & b & b & b & b \\
\{A\} & \{A\} & \{A\} & \{S, B\} & \{S, B\} & \{S, B\} & \{S, B\} \\
\emptyset & \emptyset & \emptyset & \{C\} & \\
\text{aaa} & \text{aab} & \text{abb} & \text{bbb} & \text{bbb} \\
\text{aaab} & \text{aabb} & \text{abbb} & \text{bbb} \\
\text{aaabb} & \text{aabb} & \text{abbb} & \text{bbb} \\
\text{aaabbb} & \text{aabbb} & \text{bbb} \\
\end{array}
\]
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>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</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>
</tr>
<tr>
<td>$aaab$</td>
<td>$aabb$</td>
<td>$abbb$</td>
<td>$bbb$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$aaabb$</td>
<td>$aabbb$</td>
<td>$bbbb$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$aaabbb$</td>
<td>$aabbbb$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Ryan Stansifer (CS, Florida Tech)
Example (Continued)


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


\[
\begin{array}{cccccccc}
  a & a & a & b & b & b & b \\
  \{A\} & \{A\} & \{A\} & \{S, B\} & \{S, B\} & \{S, B\} & \{S, B\} \\
  aa & aa & ab & bb & bb & bb & bb \\
  \emptyset & \emptyset & \emptyset & \{C\} & \{C\} & \{C\} & \\
  aaa & aab & abb & bbb & bbb & bbb & \\
  \emptyset & \\
  aaab & aabb & abbb & bbb & \\
  aaabb & aabbb & bbbb & \\
  aaabbb & aabbbb &
\end{array}
\]
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>
<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>
<td></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>
<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>{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>bbb</td>
<td></td>
</tr>
<tr>
<td>\ø</td>
<td>\ø</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>bbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
<td></td>
</tr>
<tr>
<td>aaabb</td>
<td>aabbb</td>
<td>bbbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaabbb</td>
<td>aabbbb</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></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>{C}</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>Ø</td>
<td>Ø</td>
<td>{S}</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>aaab</td>
<td>aabb</td>
<td>abb</td>
<td>bbbb</td>
<td>bbb</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

|   | aaabb| aabbb| bbbb|     |     |     |     |
|   | aaabbb| aabbbb|     |     |     |     |     |
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></td>
</tr>
<tr>
<td>$aaa$</td>
<td>$aab$</td>
<td>$abb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td></td>
</tr>
<tr>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>${S}$</td>
<td>$\emptyset$</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>$aaabb$</td>
<td>$aabbb$</td>
<td>$bbb$</td>
</tr>
<tr>
<td></td>
<td>$aaabb$</td>
<td>$aabbb$</td>
<td>$bbb$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$aaabbb$</td>
<td>$aabbbb$</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>
</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>
<tr>
<td>$aa$</td>
<td>$aa$</td>
<td>$ab$</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>
</tr>
<tr>
<td>$aaa$</td>
<td>$aab$</td>
<td>$abb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td></td>
</tr>
<tr>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>${S}$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td></td>
</tr>
<tr>
<td>$aaab$</td>
<td>$aabb$</td>
<td>$abbb$</td>
<td>$bbb$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$aaabb$</td>
<td>$aabbb$</td>
<td>$bbbbb$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$aaabbb$</td>
<td>$aabbbb$</td>
<td>$bbbb$</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>
<tr>
<td>${C}$</td>
<td>${C}$</td>
<td>${C}$</td>
<td>${C}$</td>
<td>${C}$</td>
<td>${C}$</td>
<td>${C}$</td>
</tr>
<tr>
<td>$aaabb$</td>
<td>$aabbb$</td>
<td>$abbbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
</tr>
<tr>
<td>$aaabbb$</td>
<td>$aabbbb$</td>
<td>$abbbbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
</tr>
<tr>
<td>$aaabbbb$</td>
<td>$aabbbbb$</td>
<td>$abbbbbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
<td>$bbb$</td>
</tr>
</tbody>
</table>
Example (Continued)


\[
\begin{array}{cccccccc}
  a & a & a & b & b & b & b \\
  \{A\} & \{A\} & \{A\} & \{S, B\} & \{S, B\} & \{S, B\} & \{S, B\} \\
  aa & aa & ab & bb & bb & bb & bb \\
  \emptyset & \emptyset & \emptyset & \{C\} & \{C\} & \{C\} & \\
  aaa & aab & abb & bbb & bbb & \\
  \emptyset & \emptyset & \{S\} & \emptyset & \emptyset & \\
  aaabb & aabb & abbb & bbb & \\
  \{C\} & \emptyset & & \\
  aaabbb & aabbb & bbbb & \\
  aaabbb & aabbb & bbbb & \\
\end{array}
\]
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>{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>abbb</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>Ø</td>
<td>Ø</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaabb</td>
<td>aabbb</td>
<td>bbbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aaabbb</td>
<td>aabbbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Example (Continued)


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


<table>
<thead>
<tr>
<th></th>
<th></th>
<th></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>{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>{S, B}</td>
<td>{C}</td>
<td>{C}</td>
<td>{C}</td>
</tr>
<tr>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>Ø</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>Ø</td>
<td>Ø</td>
<td>{S}</td>
<td>Ø</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>bbb</td>
<td>bbb</td>
<td>bbb</td>
</tr>
<tr>
<td>{C}</td>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
<td>Ø</td>
</tr>
<tr>
<td>aaabb</td>
<td>aabbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
<td>bbb</td>
</tr>
<tr>
<td>{}</td>
<td>{}</td>
<td>{}</td>
<td>{}</td>
<td>{}</td>
<td>{}</td>
<td>{}</td>
</tr>
<tr>
<td>aaabbb</td>
<td>aabbbb</td>
<td>{}</td>
<td>aabbbb</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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 $L(G)$. 

Example 7.34 from HMU, 3rd, page 306.
Example (Continued)

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

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

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

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

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

\[
\begin{array}{cccc}
\{B\} & \{A, C\} & \{A, C\} \\
ba & aa & ab & ba \\
baa & aab & aba \\
baab & aaba \\
baaba
\end{array}
\]
Example (Continued)

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

<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></td>
</tr>
</tbody>
</table>

ba  aa  ab  ba

baa  aab  aba

baab  aaba

baaba
Example (Continued)

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

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

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

\[
\begin{array}{ccccccc}
\text{b} & \text{a} & \text{a} & \text{b} & \text{a} \\
\{B\} & \{A, C\} & \{A, C\} & \{B\} & \{A, C\}
\end{array}
\]

\[
\begin{array}{cccc}
\text{ba} & \text{aa} & \text{ab} & \text{ba} \\
\{S, A\}
\end{array}
\]

\[
\begin{array}{ccc}
\text{baa} & \text{aab} & \text{aba} \\
\text{baab} & \text{aaba}
\end{array}
\]

\[
\text{baaba}
\]
Example (Continued)

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

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

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

\[
\begin{array}{ccc}
baa & aab & aba \\
baab & aaba
\end{array}
\]

baaba
Example (Continued)

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

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></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 (Continued)

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

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

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

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

baaba
Example (Continued)

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

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>{B}</td>
<td>{A, C}</td>
<td>{A, C}</td>
<td>{B}</td>
</tr>
<tr>
<td>a</td>
<td>{A, C}</td>
<td>{B}</td>
<td>{S, C}</td>
<td>{S, A}</td>
</tr>
<tr>
<td>ba</td>
<td>{S, A}</td>
<td>{B}</td>
<td>{S, C}</td>
<td>{S, A}</td>
</tr>
<tr>
<td>aab</td>
<td>{B}</td>
<td>{S, C}</td>
<td>{S, A}</td>
<td></td>
</tr>
<tr>
<td>aba</td>
<td>{S, A}</td>
<td>{B}</td>
<td>{S, A}</td>
<td></td>
</tr>
<tr>
<td>Ø</td>
<td>{B}</td>
<td>{S, C}</td>
<td>{S, A}</td>
<td></td>
</tr>
<tr>
<td>baab</td>
<td>{B}</td>
<td>{S, C}</td>
<td>{S, A}</td>
<td></td>
</tr>
<tr>
<td>aaba</td>
<td>{S, A}</td>
<td>{B}</td>
<td>{S, A}</td>
<td></td>
</tr>
</tbody>
</table>

baaba
Example (Continued)

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

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

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

\[
\begin{array}{c}
baa & aab & aba \\
\emptyset & \{B\} & \{B\}
\end{array}
\]

\[
baaba
\]
Example (Continued)

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

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

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

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

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

\[
\begin{array}{cccc}
baa & aab & aba \\
\emptyset & \{B\} & \{B\}
\end{array}
\]

\[
\begin{array}{cccc}
baab & aaba \\
\emptyset & \{S, A, C\}
\end{array}
\]

\[
baaba
\]
Example (Continued)

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

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

\[baa\]
\[aab\]
\[aba\]
\[\emptyset\]
\[baab\]
\[aaba\]
\[\emptyset\]
<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>{S, A}</td>
<td>{B}</td>
<td>{S, C}</td>
<td>{S, A}</td>
<td></td>
</tr>
<tr>
<td>{S, A}</td>
<td>{B}</td>
<td>{B}</td>
<td>{B}</td>
<td></td>
</tr>
<tr>
<td>{S, A}</td>
<td>{S, A}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{S, A}</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{S, A}</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{S, A}</td>
<td>{S, A}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{S, A}</td>
<td>{S, A}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{S, A}</td>
<td>{S, A}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{S, A}</td>
<td>{S, A}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{S, A}</td>
<td>{S, A}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{S, A}</td>
<td>{S, A}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{S, A}</td>
<td>{S, A}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{S, A}</td>
<td>{S, A}</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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.
\begin{align*}
  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 \\
  aabbab
\end{align*}
\[
\begin{align*}
\text{a} & \quad \text{a} & \quad \text{b} & \quad \text{b} & \quad \text{a} & \quad \text{b} \\
\{\text{A}\} & \quad \{\text{A}\} \\
\text{aa} & \quad \text{ab} & \quad \text{bb} & \quad \text{ba} & \quad \text{ab} \\
\text{aab} & \quad \text{abb} & \quad \text{bba} & \quad \text{bab} \\
\text{aabb} & \quad \text{abba} & \quad \text{bbab} \\
\text{aabba} & \quad \text{abbab} \\
\text{aabbab} \\
\end{align*}
\]
\text{a} \quad \text{a} \quad \text{b} \quad \text{b} \quad \text{a} \quad \text{b} \\
\{A\} \quad \{A\} \quad \{B\} \\
\text{aa} \quad \text{ab} \quad \text{bb} \quad \text{ba} \quad \text{ab} \\
\text{aab} \quad \text{abb} \quad \text{bba} \quad \text{bab} \\
\text{aabb} \quad \text{abba} \quad \text{bbab} \\
\text{aabba} \quad \text{abbab} \\
\text{aabbab}
\[
\begin{array}{cccc}
a & a & b & b \\
\{A\} & \{A\} & \{B\} & \{B\} \\
\end{array}
\]

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

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

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

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

\[
\begin{array}{cccc}
aabbab \\
\end{array}
\]
\[ a \quad a \quad b \quad b \quad a \quad b \]
\[ \{A\} \quad \{A\} \quad \{B\} \quad \{B\} \quad \{A\} \]
\[ aa \quad ab \quad bb \quad ba \quad ab \]

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

\[
\begin{array}{cccc}
aab & abb & bba & bab \\
aabb & abba & bbab \\
aabba & abbab \\
aabbrab \\
aabbbab
\end{array}
\]
\begin{align*}
a & \quad a \quad b \quad b \quad a \quad b \\
\{A\} & \quad \{A\} \quad \{B\} \quad \{B\} \quad \{A\} \quad \{B\} \\
\phantom{a} & \quad \text{aa} \quad ab \quad \text{bb} \quad ba \quad \text{ab} \\
\emptyset & \quad \{S\} \quad \emptyset \quad \{S\} \\
aab & \quad abb \quad bba \quad bab \\
aaabb & \quad abba \quad bbab \\
aabba & \quad abbab \\
aabbab & \end{align*}
\begin{align*}
    & 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 \\
    & aabbab
\end{align*}
\[
\begin{array}{cccccc}
a & a & b & b & a & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{A\} & \{B\} \\
\emptyset & ab & bb & ba & ab \\
\emptyset & \{S\} & \emptyset & \{S\} & \{S\} \\
aab & abb & bba & bab \\
\emptyset \\
aabb & abba & bbab \\
aabba & abbab \\
aabbbab
\end{array}
\]
\[
\begin{array}{cccccc}
a & a & b & b & a & b \\
\{A\} & \{A\} & \{B\} & \{B\} & \{A\} & \{B\} \\
\end{array}
\]
\[
\begin{array}{cccccc}
aa & ab & bb & ba & ab \\
\emptyset & \{S\} & \emptyset & \{S\} & \{S\} \\
\end{array}
\]
\[
\begin{array}{cccccc}
aab & abb & bba & bab \\
\emptyset & \{C\} & \\
\end{array}
\]
\[
\begin{array}{cccccc}
aabb & abba & bbab \\
\end{array}
\]
\[
\begin{array}{cccccc}
aabba & abbab & \\
\end{array}
\]
\[
\begin{array}{cccccc}
aabbb & \\
\end{array}
\]
<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} &amp; {A} &amp; {B} &amp; {B} &amp; {A} &amp; {B}</td>
<td></td>
<td></td>
<td></td>
<td></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>\Ø</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 | abbab |
|   | aabbab |
\begin{align*}
a &\quad a &\quad b &\quad b &\quad a &\quad b \\
\{A\} &\quad \{A\} &\quad \{B\} &\quad \{B\} &\quad \{A\} &\quad \{B\} \\
\emptyset &\quad \{S\} &\quad \emptyset &\quad \{S\} &\quad \{S\} \\
aab &\quad ab &\quad bb &\quad ba &\quad ab \\
\emptyset &\quad \{C\} &\quad \emptyset &\quad \{C\} \\
aabb &\quad abba &\quad bbab \\
aabba &\quad abbab \\
aabbbab &\end{align*}
\[ a \quad a \quad b \quad b \quad a \quad b \]
\[ \{A\} \quad \{A\} \quad \{B\} \quad \{B\} \quad \{A\} \quad \{B\} \]
\[ aa \quad ab \quad bb \quad ba \quad ab \]
\[ \emptyset \quad \{S\} \quad \emptyset \quad \{S\} \quad \{S\} \]
\[ aab \quad abb \quad bba \quad bab \]
\[ \emptyset \quad \{C\} \quad \emptyset \quad \{C\} \]
\[ aabbb \quad abba \quad bbab \]
\[ \{S\} \]
\[ aabba \quad abbab \]
\[ aabbab \]
\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 \{C\} \quad \emptyset \quad \{C\} \\
\text{aabb} & \quad \text{abba} \quad \text{bbab} \\
\{S\} & \quad \{S\} \\
\text{aabba} & \quad \text{abbab} \\
\end{align*}

\text{aabbab}
<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>Ø</td>
<td>{C}</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>
<tr>
<td>{S}</td>
<td>{S}</td>
<td>Ø</td>
<td></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>

aabbab
\begin{align*}
a & \quad a & \quad b & \quad b & \quad a & \quad b \\
\{A\} & \quad \{A\} & \quad \{B\} & \quad \{B\} & \quad \{A\} & \quad \{B\} \\
aa & \quad ab & \quad bb & \quad ba & \quad ab \\
\emptyset & \quad \{S\} & \quad \emptyset & \quad \{S\} & \quad \{S\} \\
aab & \quad abb & \quad bba & \quad bab \\
\emptyset & \quad \{C\} & \quad \emptyset & \quad \{C\} \\
aaBB & \quad abba & \quad bbab \\
\{S\} & \quad \{S\} & \quad \emptyset \\
\text{aabba} & \quad \text{abbab} \\
\{D\} \\
aabbbaBBa \\
\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>Ø</td>
<td>{C}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabbb</td>
<td>abba</td>
<td>bbab</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>{S}</td>
<td>{S}</td>
<td>Ø</td>
<td></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>
<tr>
<td>{D}</td>
<td>{C}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aabbbab</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>
</tr>
<tr>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>$a$</td>
<td>$a$</td>
<td>$b$</td>
<td>$b$</td>
<td>$a$</td>
<td>$b$</td>
<td></td>
</tr>
<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>$\emptyset$</td>
<td>${S}$</td>
<td>$\emptyset$</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>$\emptyset$</td>
<td>${C}$</td>
<td>$\emptyset$</td>
<td>${C}$</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>
<tr>
<td>${S}$</td>
<td>${S}$</td>
<td>$\emptyset$</td>
<td></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>
<tr>
<td>${D}$</td>
<td>${C}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$aabbab$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>${S}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Earley’s algorithm and CYK both run in time $O(n^3)$ on general grammars. However, Earley’s algorithm does much better on grammars of practical interest. Although we will not prove it, Early’s algorithm runs in time $O(n^2)$ on unambiguous grammars and in time $O(n)$ on LR(1) grammars. (We will not define LR(1) grammars, but they are equivalent to DCFLs. Furthermore, the syntax of almost every programming language is essentially an LR(1) grammar. For more information, see a textbook on compiler design.) In contrast, the CYK algorithm always takes time bounded above and below by multiples of $n^3$, regardless of the grammar.

Floyd, Section 5.12, page 392–393.
Earley’s algorithm and CYK both run in time $O(n^3)$ on general grammars. However, Earley’s algorithm does much better on grammars of practical interest. Although we will not prove it, Early’s algorithm runs in time $O(n^2)$ on unambiguous grammars and in time $O(n)$ on LR(1) grammars. (We will not define LR(1) grammars, but they are equivalent to DCFLs. Furthermore, the syntax of almost every programming language is essentially an LR(1) grammar. For more information, see a textbook on compiler design.) In contrast, the CYK algorithm always takes time bounded above and below by multiples of $n^3$, regardless of the grammar.

Floyd, Section 5.12, page 392–393.
Theorem For every fixed $k \geq 1$: A language has an LR(k) grammar iff it is DCFL. Knuth, Section V on page 628.
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 $a^n cb^n \cup a^n db^{2n}$.

Theorem. $L$ is accepted by a DPDA.

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

---

\textsuperscript{1}Lehtinen and Okhotin, LNCS, 2012, page 156.
Example of a Context-Sensitive Grammar

Linz 6th, §11.3, Example 11.2, page 301

1. $S \rightarrow aAbc$
2. $S \rightarrow abc$
3. $Ab \rightarrow bA$
4. $Ac \rightarrow Bbcc$
5. $bB \rightarrow Bb$
6. $aB \rightarrow aa$
7. $aB \rightarrow aaA$

$A$ means you must add an $a$, $b$, and $c$. $B$ means you have added a $b$ and a $c$ and you must add an $A$. 
Theorem. The family of context-free languages is a proper subset of the family of context-sensitive languages.

\[ L = \{a^n b^n c^n\} \text{ is not context free.} \]

Linz 6th,
Sudkamp, §7.4, Example 7.4.1, page 242.

\[ L = \{a^n b^n c^n\} \text{ is context sensitive.} \]
We have seen so many restrictions on grammars, it becomes necessary to define:

Definition. An unrestricted grammar is a grammar without restrictions.

So an unrestricted grammar is simply a grammar. Although CFGs are the most common, useful, and intuitive of the grammars, we now turn our attention to unfettered grammars.
Example of an Unrestricted Grammar

Sudkamp, §10.1, Example 10.1.1, page 327

1. $S \rightarrow aAbc$
2. $S \rightarrow \epsilon$
3. $A \rightarrow aAbC$
4. $A \rightarrow \epsilon$
5. $Cb \rightarrow bc$
6. $Cc \rightarrow cc$

$L(G) = \{a^n b^n c^n | 0 \leq n\}$

$S \xrightarrow{1} aAbc \Rightarrow a^iA(bC)^{i-1}bc \Rightarrow a^i(bC)^{i-1}bc \Rightarrow a^i b^i C^{i-1}c \Rightarrow a^i b^i c^i$
Example of an Unrestricted Grammar

Sudkamp, §10.1, Example 10.1.2, page 327

\[
S \rightarrow aT[a] \mid bT[b] \mid []
\]

\[
T[a] \rightarrow aT[a] \mid bT[B] \mid []
\]

\[
T[b] \rightarrow aT[a] \mid bT[B] \mid []
\]

\[
Aa \rightarrow aA
\]

\[
Ab \rightarrow bA
\]

\[
A \rightarrow a
\]

\[
Ba \rightarrow aB
\]

\[
B \rightarrow b
\]

\[
Bb \rightarrow bB
\]

\[
L(G) = \{w[w] \mid w \in \{a, b\}^*\}
\]
\[
A \xrightarrow{1} aT[a] \\
1 \xrightarrow{} aaT[Aa] \\
1 \xrightarrow{} aaT[aA] \\
1 \xrightarrow{} aaT[aa] \\
1 \xrightarrow{} aabT[Baa] \\
1 \xrightarrow{} aabT[aba] \\
1 \xrightarrow{} aabT[aaB] \\
1 \xrightarrow{} aabT[aab] \\
1 \xrightarrow{} aab[aab]
\]
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

$S$

$\Rightarrow SV_{\square}$

$S \rightarrow SV_{\square}$

1 (11.6) add trailing blanks

$\Rightarrow TV_{\square}$

$S \rightarrow T$

2 (11.6) generate input

$\Rightarrow TV_{aa}V_{\square}$

$T \rightarrow TV_{aa}$

3 (11.7) an input character

$\Rightarrow V_{a0a}V_{aa}V_{\square}$

$V_{a0a}V_{aa} \rightarrow V_{aa}V_{a0a}$

4 (11.7) start state, 1st character

$\Rightarrow V_{aa}V_{a0a}V_{\square}$

$V_{a0a}V_{\square} \rightarrow V_{aa}V_{\square} V_{\square}$

5 (11.8) $\langle q_0, a, q_0, a, R \rangle \in \Delta$

$\Rightarrow V_{aa}V_{aa}V_{\square}$

$V_{aa}V_{\square} \rightarrow V_{aa}V_{\square} V_{\square}$

6 (11.8) $\langle q_0, a, q_0, a, R \rangle \in \Delta$

$\Rightarrow V_{aa}V_{a1a}V_{\square}$

$V_{a1a} \rightarrow a$

7 (11.9) $\langle q_0, \square, q_1, a, L \rangle \in \Delta$

$\Rightarrow V_{aa}aV_{\square}$

$aV_{\square} \rightarrow a_{\square}$

8 (11.10) restore input

$\Rightarrow V_{aa}a_{\square}$

$V_{aa}a \rightarrow a$

9 (11.11) restore input

$\Rightarrow aa_{\square}$

10 (11.12) restore input

$\Rightarrow aa$

$\square \rightarrow \epsilon$

11 (11.13) restore input
Example 11.1

\[ S \Rightarrow SV \]
\[ \Rightarrow TV \]
\[ \Rightarrow TV_{aa} \]
\[ \Rightarrow V_{a0a}V_{aa} \]
\[ \Rightarrow V_{aa}V_{a0a} \]
\[ \Rightarrow V_{aa}V_{aa}V_{00} \]
\[ \Rightarrow V_{aa}V_{a1a} \]
\[ \Rightarrow V_{aa}aV \]
\[ \Rightarrow V_{aa}a \]
\[ \Rightarrow aa \]

\[ S \Rightarrow SV \quad (11.6) \]

\[ \quad \vdash \langle \epsilon, q_0, aa \rangle \]
\[ \quad \vdash \langle a, q_0, a \rangle \]
\[ \quad \vdash \langle aa, q_0, \square \rangle \]
\[ \quad \vdash \langle a, q_1, a \square \rangle \]

\[ \square \Rightarrow \epsilon \quad (11.13) \]
References I

