Formal Languages, Automata and Codes

Oleg Gutik

Lecture 14

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form
$$A \to x,$$
where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form
$$A \to x,$$
where $A \in V$ and $x \in (V \cup T)^*$.
A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

5.1 Context-Free Grammars

The productions in a regular grammar are restricted in two ways: The left side must be a single variable, while the right side has a special form. To create grammars that are more powerful, we must relax some of these restrictions. By retaining the restriction on the left side, but permitting anything on the right, we get context-free grammars.

Definition 5.1

A grammar $G = (V, T, S, P)$ is said to be *context-free* if all productions in P have the form

$$A \to x,$$

where $A \in V$ and $x \in (V \cup T)^*$.

A language L is said to be *context-free* if and only if there is a context-free grammar G such that $L = L(G)$.

Every regular grammar is context-free, so a regular language is also a context-free one. But, as we know from simple examples such as $\{a^n b^n : n \geqslant 0\}$, there are nonregular languages. We have already shown in Example 1.11 that this language can be generated by a context-free grammar, so we see that the family of regular languages is a proper subset of the family of context-free languages.

Context-free grammars derive their name from the fact that the substitution of the variable on the left of a production can be made any time such a variable appears in a sentential form. It does not depend on the symbols in the rest of the sentential form (the context). This feature is the consequence of allowing only a single variable on the left side of the production.

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \{ww^R : w \in \{a, b\}\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \{ab(bbaa)^n bba(ba)^n : n \geqslant 0\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \{ww^R : w \in \{a, b\}\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \{ab(bbaa)^n bba(ba)^n : n \geqslant 0\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \left\{ww^R : w \in \{a, b\}\right\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \left\{ab(bbaa)^n bba(ba)^n : n \geqslant 0\right\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \left\{ww^R : w \in \{a, b\}\right\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \left\{ab(bbaa)^n bba(ba)^n : n \geqslant 0\right\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \left\{ww^R : w \in \{a, b\}\right\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \left\{ab(bbaa)^n bba(ba)^n : n \geqslant 0\right\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \left\{ww^R \colon w \in \{a, b\}\right\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \left\{ab(bbaa)^n bba(ba)^n \colon n \geqslant 0\right\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \{ww^R : w \in \{a, b\}\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \{ab(bbaa)^n bba(ba)^n : n \geqslant 0\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \{ww^R : w \in \{a, b\}\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \{ab(bbaa)^n bba(ba)^n : n \geqslant 0\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \left\{ww^R \colon w \in \{a, b\}^*\right\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \left\{ab(bbaa)^n bba(ba)^n : n \geqslant 0\right\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \{ww^R \colon w \in \{a, b\}\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \{ab(bbaa)^n bba(ba)^n : n \geqslant 0\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$
$$S \to bSb,$$
$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \left\{ww^R \colon w \in \{a, b\}\right\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$
$$A \to aaBb,$$
$$B \to bbAa,$$
$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \left\{ab(bbaa)^n bba(ba)^n : n \geqslant 0\right\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \left\{ww^R : w \in \{a, b\}\right\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \left\{ab(bbaa)^n bba(ba)^n : n \geqslant 0\right\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \left\{ww^R \colon w \in \{a, b\}\right\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \left\{ab(bbaa)^n bba(ba)^n : n \geqslant 0\right\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \left\{ww^R \colon w \in \{a, b\}\right\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \left\{ab(bbaa)^n bba(ba)^n : n \geqslant 0\right\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \left\{ww^R \colon w \in \{a, b\}\right\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \left\{ab(bbaa)^n bba(ba)^n : n \geqslant 0\right\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \left\{ww^R \colon w \in \{a, b\}\right\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \left\{ab(bbaa)^n bba(ba)^n : n \geqslant 0\right\}.$$

Examples of Context-Free Languages

Example 5.1

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$

$$S \to bSb,$$

$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \left\{ww^R \colon w \in \{a, b\}\right\}.$$

The language is context-free, but as shown in Example 4.8, it is not regular.

Example 5.2

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to abB,$$

$$A \to aaBb,$$

$$B \to bbAa,$$

$$A \to \lambda,$$

is context-free. We leave it to the reader to show that

$$L(G) = \left\{ab(bbaa)^n bba(ba)^n \colon n \geqslant 0\right\}.$$

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language
$$L = \{a^n b^m : n \neq m\}$$
is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with
$$S \to AS_1,$$
$$S_1 \to aS_1b|\lambda,$$
$$A \to aA|a.$$
We can use similar reasoning for the case $n < m$, and we get the answer
$$S \to AS_1|S_1B,$$
$$S_1 \to aS_1b|\lambda,$$
$$A \to aA|a,$$
$$B \to bB|b.$$
The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language
$$L = \{a^n b^m : n \neq m\}$$
is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with
$$S \to AS_1,$$
$$S_1 \to aS_1b|\lambda,$$
$$A \to aA|a.$$
We can use similar reasoning for the case $n < m$, and we get the answer
$$S \to AS_1|S_1B,$$
$$S_1 \to aS_1b|\lambda,$$
$$A \to aA|a,$$
$$B \to bB|b.$$
The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language.

The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \rightarrow AS_1,$$

$$S_1 \rightarrow aS_1b|\lambda,$$

$$A \rightarrow aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \rightarrow AS_1|S_1B,$$

$$S_1 \rightarrow aS_1b|\lambda,$$

$$A \rightarrow aA|a,$$

$$B \rightarrow bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language
$$L = \{a^n b^m : n \neq m\}$$
is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$
$$S_1 \to aS_1b|\lambda,$$
$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$
$$S_1 \to aS_1b|\lambda,$$
$$A \to aA|a,$$
$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language.

However, the grammar is not linear.

Both of the above examples involve grammars that are not only context-free, but linear. Regular and linear grammars are clearly context-free, but a context-free grammar is not necessarily linear.

Example 5.3

The language

$$L = \{a^n b^m : n \neq m\}$$

is context-free.

To show this, we need to produce a context-free grammar for the language. The case of $n = m$ is solved in Example 1.11 and we can build on that solution. Take the case $n > m$. We first generate a string with an equal number of a's and b's, then add extra a's on the left. This is done with

$$S \to AS_1,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a.$$

We can use similar reasoning for the case $n < m$, and we get the answer

$$S \to AS_1|S_1B,$$

$$S_1 \to aS_1b|\lambda,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

The resulting grammar is context-free, hence L is a context-free language. However, the grammar is not linear.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \text{ where } v \text{ is any prefix of } w\}. \tag{1}$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as (()) and ()()() and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \text{ where } v \text{ is any prefix of } w\}. \quad (1)$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as (()) and ()()() and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \\ \text{where } v \text{ is any prefix of } w\}. \tag{1}$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as (()) and ()()() and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \text{ where } v \text{ is any prefix of } w\}. \quad (1)$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as (()) and ()()() and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \text{ where } v \text{ is any prefix of } w\}. \quad (1)$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as (()) and ()()() and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \text{ where } v \text{ is any prefix of } w\}. \quad (1)$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as (()) and ()()() and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \text{ where } v \text{ is any prefix of } w\}. \tag{1}$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as (()) and ()()() and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \text{ where } v \text{ is any prefix of } w\}. \tag{1}$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as (()) and ()()() and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \\ \text{where } v \text{ is any prefix of } w\}. \tag{1}$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as (()) and ()()() and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \text{ where } v \text{ is any prefix of } w\}. \tag{1}$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as $(())$ and $()()()$ and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \text{ where } v \text{ is any prefix of } w\}. \tag{1}$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as $(())$ and $()()()$ and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \\ \text{where } v \text{ is any prefix of } w\}. \tag{1}$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as $(())$ and $()()()$ and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \text{ where } v \text{ is any prefix of } w\}. \quad (1)$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as (()) and ()()() and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Example 5.4

Consider the grammar with productions

$$S \to aSb|SS|\lambda.$$

This is another grammar that is context-free, but not linear. Some strings in $L(G)$ are $abaabb$, $aababb$, and $ababab$. It is not difficult to conjecture and prove that

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geqslant n_b(v), \text{ where } v \text{ is any prefix of } w\}. \tag{1}$$

We can see the connection with programming languages clearly if we replace a and b with left and right parentheses, respectively. The language L includes such strings as (()) and ()()() and is in fact the set of all properly nested parenthesis structures for the common programming languages.

Here again there are many other equivalent grammars. But, in contrast to Example 5.3, it is not so easy to see if there are any linear ones.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB.$
2. $A \to aaA.$
3. $A \to \lambda.$
4. $B \to Bb.$
5. $B \to \lambda.$

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \stackrel{1}{\Rightarrow} AB \stackrel{2}{\Rightarrow} aaAB \stackrel{3}{\Rightarrow} aaB \stackrel{4}{\Rightarrow} aaBb \stackrel{5}{\Rightarrow} aab$$

and

$$S \stackrel{1}{\Rightarrow} AB \stackrel{4}{\Rightarrow} ABb \stackrel{2}{\Rightarrow} aaABb \stackrel{5}{\Rightarrow} aaAb \stackrel{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB$.
2. $A \to aaA$.
3. $A \to \lambda$.
4. $B \to Bb$.
5. $B \to \lambda$.

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \stackrel{1}{\Rightarrow} AB \stackrel{2}{\Rightarrow} aaAB \stackrel{3}{\Rightarrow} aaB \stackrel{4}{\Rightarrow} aaBb \stackrel{5}{\Rightarrow} aab$$

and

$$S \stackrel{1}{\Rightarrow} AB \stackrel{4}{\Rightarrow} ABb \stackrel{2}{\Rightarrow} aaABb \stackrel{5}{\Rightarrow} aaAb \stackrel{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB$.
2. $A \to aaA$.
3. $A \to \lambda$.
4. $B \to Bb$.
5. $B \to \lambda$.

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \stackrel{1}{\Rightarrow} AB \stackrel{2}{\Rightarrow} aaAB \stackrel{3}{\Rightarrow} aaB \stackrel{4}{\Rightarrow} aaBb \stackrel{5}{\Rightarrow} aab$$

and

$$S \stackrel{1}{\Rightarrow} AB \stackrel{4}{\Rightarrow} ABb \stackrel{2}{\Rightarrow} aaABb \stackrel{5}{\Rightarrow} aaAb \stackrel{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB$.
2. $A \to aaA$.
3. $A \to \lambda$.
4. $B \to Bb$.
5. $B \to \lambda$.

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

and

$$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{2}{\Rightarrow} aaABb \overset{5}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB$.
2. $A \to aaA$.
3. $A \to \lambda$.
4. $B \to Bb$.
5. $B \to \lambda$.

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \stackrel{1}{\Rightarrow} AB \stackrel{2}{\Rightarrow} aaAB \stackrel{3}{\Rightarrow} aaB \stackrel{4}{\Rightarrow} aaBb \stackrel{5}{\Rightarrow} aab$$

and

$$S \stackrel{1}{\Rightarrow} AB \stackrel{4}{\Rightarrow} ABb \stackrel{2}{\Rightarrow} aaABb \stackrel{5}{\Rightarrow} aaAb \stackrel{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB$.
2. $A \to aaA$.
3. $A \to \lambda$.
4. $B \to Bb$.
5. $B \to \lambda$.

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

and

$$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{2}{\Rightarrow} aaABb \overset{5}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB$.
2. $A \to aaA$.
3. $A \to \lambda$.
4. $B \to Bb$.
5. $B \to \lambda$.

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \stackrel{1}{\Rightarrow} AB \stackrel{2}{\Rightarrow} aaAB \stackrel{3}{\Rightarrow} aaB \stackrel{4}{\Rightarrow} aaBb \stackrel{5}{\Rightarrow} aab$$

and

$$S \stackrel{1}{\Rightarrow} AB \stackrel{4}{\Rightarrow} ABb \stackrel{2}{\Rightarrow} aaABb \stackrel{5}{\Rightarrow} aaAb \stackrel{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB$.
2. $A \to aaA$.
3. $A \to \lambda$.
4. $B \to Bb$.
5. $B \to \lambda$.

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

and

$$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{2}{\Rightarrow} aaABb \overset{5}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB.$
2. $A \to aaA.$
3. $A \to \lambda.$
4. $B \to Bb.$
5. $B \to \lambda.$

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

and

$$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{2}{\Rightarrow} aaABb \overset{5}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar
$G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB.$
2. $A \to aaA.$
3. $A \to \lambda.$
4. $B \to Bb.$
5. $B \to \lambda.$

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.
Consider now the two derivations

$$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

and

$$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{2}{\Rightarrow} aaABb \overset{5}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar
$G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB.$
2. $A \to aaA.$
3. $A \to \lambda.$
4. $B \to Bb.$
5. $B \to \lambda.$

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.
Consider now the two derivations

$$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

and

$$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{2}{\Rightarrow} aaABb \overset{5}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB.$
2. $A \to aaA.$
3. $A \to \lambda.$
4. $B \to Bb.$
5. $B \to \lambda.$

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

and

$$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{2}{\Rightarrow} aaABb \overset{5}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB.$
2. $A \to aaA.$
3. $A \to \lambda.$
4. $B \to Bb.$
5. $B \to \lambda.$

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

and

$$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{2}{\Rightarrow} aaABb \overset{5}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB.$
2. $A \to aaA.$
3. $A \to \lambda.$
4. $B \to Bb.$
5. $B \to \lambda.$

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \stackrel{1}{\Rightarrow} AB \stackrel{2}{\Rightarrow} aaAB \stackrel{3}{\Rightarrow} aaB \stackrel{4}{\Rightarrow} aaBb \stackrel{5}{\Rightarrow} aab$$

and

$$S \stackrel{1}{\Rightarrow} AB \stackrel{4}{\Rightarrow} ABb \stackrel{2}{\Rightarrow} aaABb \stackrel{5}{\Rightarrow} aaAb \stackrel{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

$$1.\ S \to AB.$$

$$2.\ A \to aaA.$$

$$3.\ A \to \lambda.$$

$$4.\ B \to Bb.$$

$$5.\ B \to \lambda.$$

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

and

$$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{2}{\Rightarrow} aaABb \overset{5}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar $G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB.$
2. $A \to aaA.$
3. $A \to \lambda.$
4. $B \to Bb.$
5. $B \to \lambda.$

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.

Consider now the two derivations

$$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

and

$$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{2}{\Rightarrow} aaABb \overset{5}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar
$G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB.$
2. $A \to aaA.$
3. $A \to \lambda.$
4. $B \to Bb.$
5. $B \to \lambda.$

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.
Consider now the two derivations

$$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

and

$$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{2}{\Rightarrow} aaABb \overset{5}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Leftmost and Rightmost Derivations

In a grammar that is not linear, a derivation may involve sentential forms with more than one variable. In such cases, we have a choice in the order in which variables are replaced. Take, for example, the grammar
$G = (\{A, B, S\}, \{a, b\}, S, P)$ with productions

1. $S \to AB.$
2. $A \to aaA.$
3. $A \to \lambda.$
4. $B \to Bb.$
5. $B \to \lambda.$

This grammar generates the language $L(G) = \{a^{2n}b^m : n \geqslant 0, m \geqslant 0\}$. Carry out a few derivations to convince yourself of this.
Consider now the two derivations

$$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

and

$$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{2}{\Rightarrow} aaABb \overset{5}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab.$$

In order to show which production is applied, we have numbered the productions and written the appropriate number on the $\Rightarrow$ symbol. From this we see that the two derivations not only yield the same sentence but also use exactly the same productions. The difference is entirely in the order in which the productions are applied. To remove such irrelevant factors, we often require that the variables be replaced in a specific order.

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$

$$A \to bBb,$$

$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$

$$A \to bBb,$$

$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$

$$A \to bBb,$$

$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$

$$A \to bBb,$$

$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$

$$A \to bBb,$$

$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

Definition 5.2

A derivation is said to be *leftmost* if in each step the leftmost variable in the sentential form is replaced. If in each step the rightmost variable is replaced, we call the derivation *rightmost*.

Example 5.5

Consider the grammar with productions

$$S \to aAB,$$

$$A \to bBb,$$

$$B \to A|\lambda.$$

Then

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$$

is a leftmost derivation of the string $abbbb$. A rightmost derivation of the same string is

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$$

5.1 Context-Free Grammars

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure

shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure

shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure

shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure

shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure

shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure

shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure

shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure

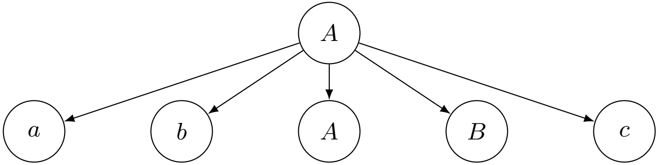
shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure



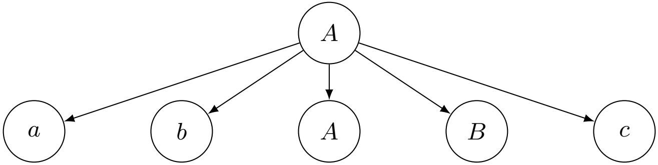
shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure



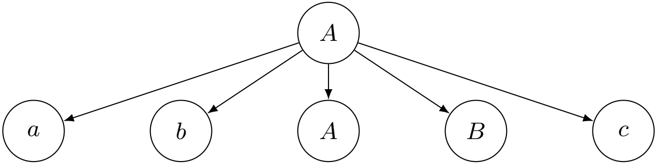
shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure



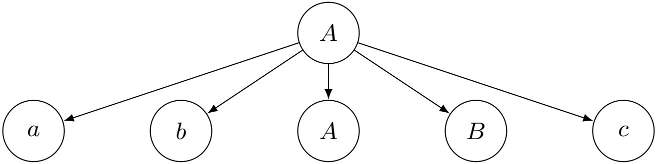
shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure



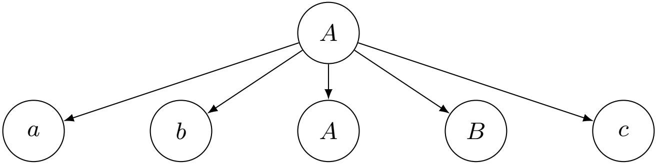
shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure



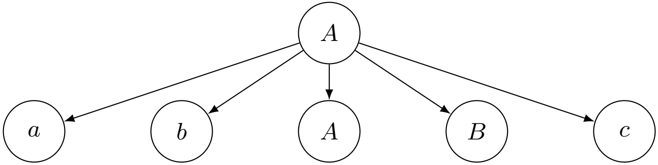
shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure



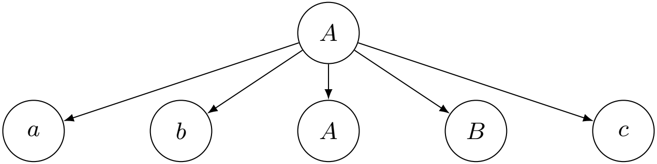
shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure



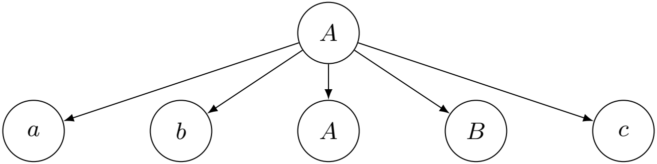
shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure



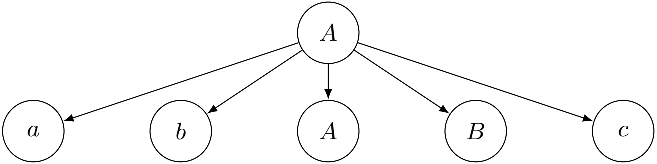
shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Derivation Trees

A second way of showing derivations, independent of the order in which productions are used, is by a derivation or parse tree. A derivation tree is an ordered tree in which nodes are labeled with the left sides of productions and in which the children of a node represent its corresponding right sides. For example, the Figure



shows part of a derivation tree representing the production

$$A \to abABc.$$

In a derivation tree, a node labeled with a variable occurring on the left side of a production has children consisting of the symbols on the right side of that production. Beginning with the root, labeled with the start symbol and ending in leaves that are terminals, a derivation tree shows how each variable is replaced in the derivation. The following definition makes this notion precise.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$,

$$A \to a_1 a_2 \cdots a_n.$$

5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$,

$$A \to a_1 a_2 \cdots a_n.$$

5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$,

$$A \to a_1 a_2 \cdots a_n.$$

5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$,

$$A \to a_1 a_2 \cdots a_n.$$

5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \rightarrow a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Definition 5.3

Let $G = (V, T, S, P)$ be a context-free grammar. An *ordered tree* is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.
2. Every leaf has a label from $T \cup \{\lambda\}$.
3. Every interior vertex (a vertex that is not a leaf) has a label from V.
4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form
$$A \to a_1 a_2 \cdots a_n.$$
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

A tree that has properties 3, 4, and 5, but in which 1 does not necessarily hold and in which property 2 is replaced by

(2a) Every leaf has a label from $V \cup T \cup \{\lambda\}$,

is said to be a *partial derivation tree*.

The string of symbols obtained by reading the leaves of the tree from left to right, omitting any λ's encountered, is said to be the *yield* of the tree. The descriptive term left to right can be given a precise meaning. The yield is the string of terminals in the order they are encountered when the tree is traversed in a depth-first manner, always taking the leftmost unexplored branch.

Example 5.6

Consider the grammar G, with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

The tree in the Figure is a partial derivation tree for G,

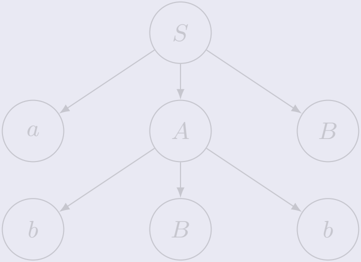
while the tree in the following Figure is a derivation tree.

Example 5.6

Consider the grammar G, with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

The tree in the Figure is a partial derivation tree for G,



while the tree in the following Figure is a derivation tree.

5.1 Context-Free Grammars

Example 5.6

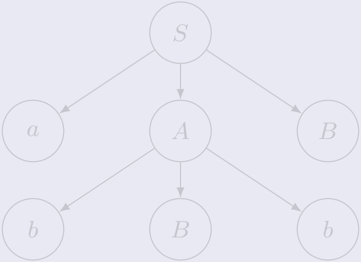
Consider the grammar G, with productions

$$S \to aAB,$$

$$A \to bBb,$$

$$B \to A|\lambda.$$

The tree in the Figure is a partial derivation tree for G,



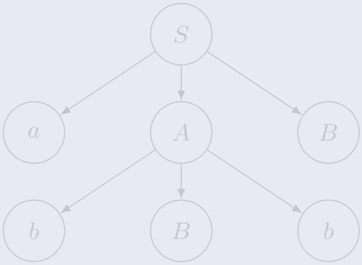
while the tree in the following Figure is a derivation tree.

Example 5.6

Consider the grammar G, with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

The tree in the Figure is a partial derivation tree for G,



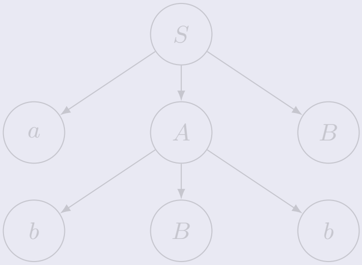
while the tree in the following Figure is a derivation tree.

Example 5.6

Consider the grammar G, with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

The tree in the Figure is a partial derivation tree for G,



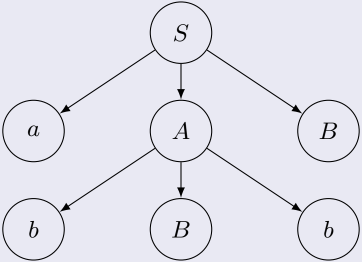
while the tree in the following Figure is a derivation tree.

Example 5.6

Consider the grammar G, with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

The tree in the Figure is a partial derivation tree for G,



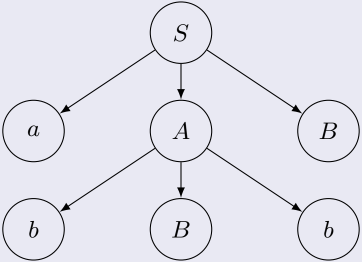
while the tree in the following Figure is a derivation tree.

Example 5.6

Consider the grammar G, with productions

$$S \to aAB,$$
$$A \to bBb,$$
$$B \to A|\lambda.$$

The tree in the Figure is a partial derivation tree for G,



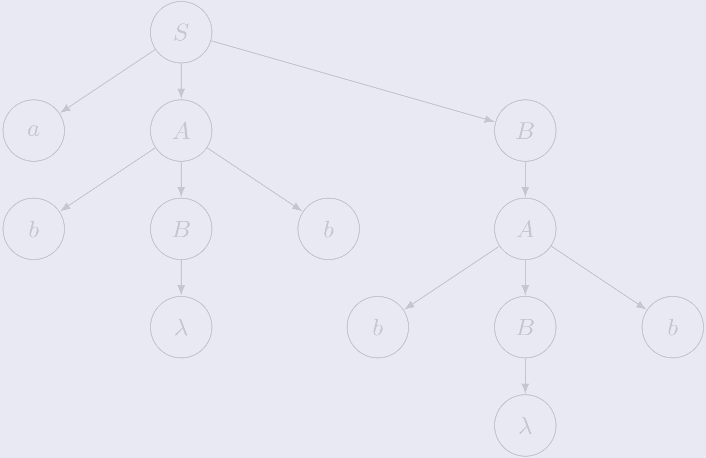
while the tree in the following Figure is a derivation tree.

5.1 Context-Free Grammars

Example 5.6 (continuation)

The string $abBbB$, which is the yield of the first tree, is a sentential form of G. The yield of the second tree, $abbbb$, is a sentence of $L(G)$.

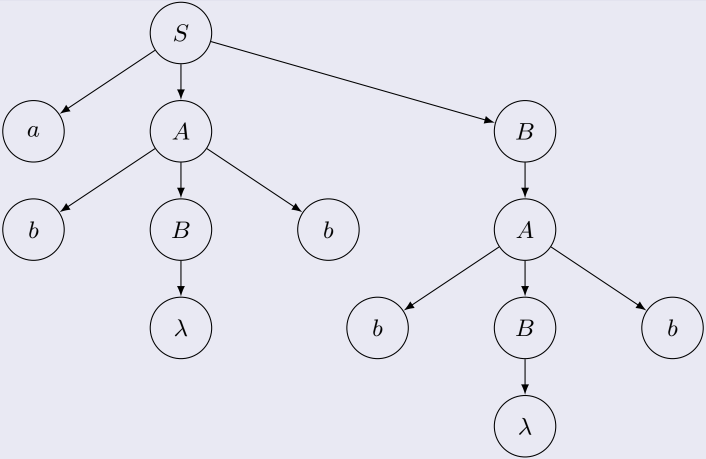
Example 5.6 (continuation)



The string $abBbB$, which is the yield of the first tree, is a sentential form of G.
The yield of the second tree, $abbbb$, is a sentence of $L(G)$.

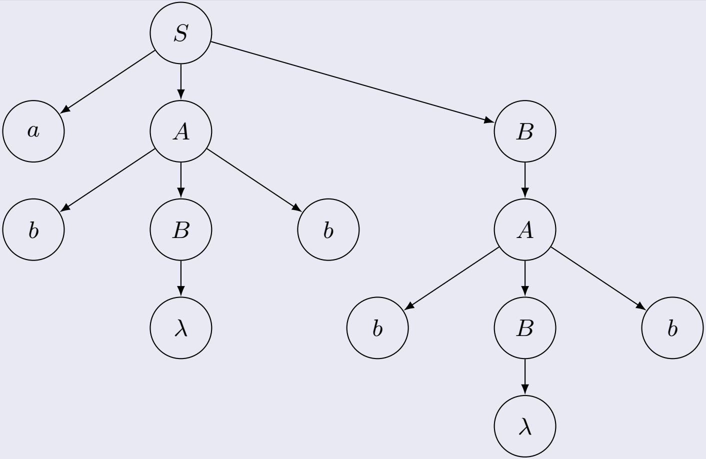
5.1 Context-Free Grammars

Example 5.6 (continuation)



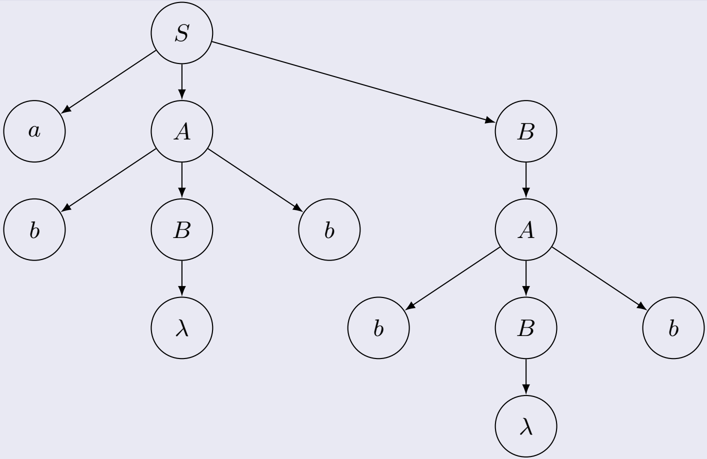
The string $abBbB$, which is the yield of the first tree, is a sentential form of G. The yield of the second tree, $abbbb$, is a sentence of $L(G)$.

Example 5.6 (continuation)



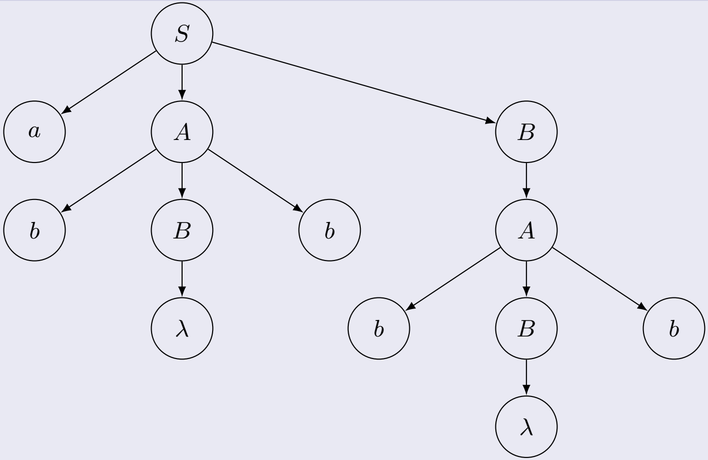
The string $abBbB$, which is the yield of the first tree, is a sentential form of G. The yield of the second tree, $abbbb$, is a sentence of $L(G)$.

Example 5.6 (continuation)



The string $abBbB$, which is the yield of the first tree, is a sentential form of G.
The yield of the second tree, $abbbb$, is a sentence of $L(G)$.

Example 5.6 (continuation)



The string $abBbB$, which is the yield of the first tree, is a sentential form of G. The yield of the second tree, $abbbb$, is a sentence of $L(G)$.

Relation Between Sentential Forms and Derivation Trees

Derivation trees give a very explicit and easily comprehended description of a derivation. Like transition graphs for finite automata, this explicitness is a great help in making arguments. First, though, we must establish the connection between derivations and derivation trees.

Relation Between Sentential Forms and Derivation Trees

Derivation trees give a very explicit and easily comprehended description of a derivation. Like transition graphs for finite automata, this explicitness is a great help in making arguments. First, though, we must establish the connection between derivations and derivation trees.

Relation Between Sentential Forms and Derivation Trees

Derivation trees give a very explicit and easily comprehended description of a derivation. Like transition graphs for finite automata, this explicitness is a great help in making arguments. First, though, we must establish the connection between derivations and derivation trees.

Relation Between Sentential Forms and Derivation Trees

Derivation trees give a very explicit and easily comprehended description of a derivation. Like transition graphs for finite automata, this explicitness is a great help in making arguments. First, though, we must establish the connection between derivations and derivation trees.

Relation Between Sentential Forms and Derivation Trees

Derivation trees give a very explicit and easily comprehended description of a derivation. Like transition graphs for finite automata, this explicitness is a great help in making arguments. First, though, we must establish the connection between derivations and derivation trees.

Relation Between Sentential Forms and Derivation Trees

Derivation trees give a very explicit and easily comprehended description of a derivation. Like transition graphs for finite automata, this explicitness is a great help in making arguments. First, though, we must establish the connection between derivations and derivation trees.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n + 1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \overset{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2\cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2\cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2\cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \overset{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2\cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2\cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2\cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n + 1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

5.1 Context-Free Grammars

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \overset{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_m y = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_m y = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_m y = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_m y = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n + 1$ steps must be such that

$$S \overset{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_m y = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_m y = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n + 1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n + 1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n + 1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n + 1$ steps must be such that

$$S \overset{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_m y = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_m y = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_m y = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_m y = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \overset{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \rightarrow u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_m y = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \rightarrow a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_m y = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2\cdots a_m y = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2\cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2\cdots a_m y = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_m y = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_m y = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \overset{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_m y = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_m y = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \stackrel{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_m y = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_m y = w$. By induction, we therefore claim that the result is true for all sentential forms.

Theorem 5.1

Let $G = (V, T, S, P)$ be a context-free grammar. Then for every $w \in L(G)$, there exists a derivation tree of G whose yield is w. Conversely, the yield of any derivation tree is in $L(G)$. Also, if t_G is any partial derivation tree for G whose root is labeled S, then the yield of t_G is a sentential form of G.

Proof. First we show that for every sentential form of $L(G)$ there is a corresponding partial derivation tree. We do this by induction on the number of steps in the derivation. As a basis, we note that the claimed result is true for every sentential form derivable in one step. Since $S \Rightarrow u$ implies that there is a production $S \to u$, this follows immediately from Definition 5.3.

Assume that for every sentential form derivable in n steps, there is a corresponding partial derivation tree. Now any w derivable in $n+1$ steps must be such that

$$S \overset{*}{\Rightarrow} xAy, \qquad x, y \in (V \cup T)^*, \quad A \in V,$$

in n steps, and

$$xAy \Rightarrow xa_1a_2 \cdots a_my = w, \quad a_i \in V \cup T.$$

Since by the inductive assumption there is a partial derivation tree with yield xAy, and since the grammar must have production $A \to a_1a_2 \cdots a_m$, we see that by expanding the leaf labeled A, we get a partial derivation tree with yield $xa_1a_2 \cdots a_my = w$. By induction, we therefore claim that the result is true for all sentential forms.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ∎

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ∎

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of G and that the yield of every derivation tree is in $L(G)$. ■

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation.

Thank You for attention!