

23MT2014

THEORY OF COMPUTATION

Topic:

INTRODUCTION TO GRAMMAR

Session - 10



AIM OF THE SESSION



To introduce students to the concept of grammar in automata theory and enable them to understand and apply grammar rules and formal languages.

INSTRUCTIONAL OBJECTIVES



This Session is designed to:

- 1. To familiarize students with the fundamental concepts of grammar in automata theory, including formal languages, production rules, and derivations.
- 2. To provide students with a comprehensive understanding of different types of grammars, such as regular grammars, context-free grammars, and context-sensitive grammars.

LEARNING OUTCOMES



At the end of this session, you should be able to:

- 1. Define and explain the concepts of formal languages, production rules, and derivations in the context of grammar theory.
- 2. Identify and classify different types of grammars, including regular grammars, context-free grammars, and context-sensitive grammars.











Grammars

Grammars express languages

Example: the English language

$$\langle sentence \rangle \rightarrow \langle noun_phrase \rangle \langle predicate \rangle$$

$$\langle noun_phrase \rangle \rightarrow \langle article \rangle \langle noun \rangle$$







$$\langle article \rangle \rightarrow a$$

 $\langle article \rangle \rightarrow the$

$$\langle noun \rangle \rightarrow cat$$

 $\langle noun \rangle \rightarrow dog$

$$\langle verb \rangle \rightarrow runs$$

 $\langle verb \rangle \rightarrow walks$











• A derivation of "the dog walks":

 $\langle sentence \rangle \Rightarrow \langle noun_phrase \rangle \langle predicate \rangle$

 $\Rightarrow \langle noun_phrase \rangle \langle verb \rangle$

 $\Rightarrow \langle article \rangle \langle noun \rangle \langle verb \rangle$

 \Rightarrow the $\langle noun \rangle \langle verb \rangle$

 \Rightarrow the dog $\langle verb \rangle$

 \Rightarrow the dog walks











A derivation of "a cat runs":

 $\langle sentence \rangle \Rightarrow \langle noun_phrase \rangle \langle predicate \rangle$

 $\Rightarrow \langle noun_phrase \rangle \langle verb \rangle$

 $\Rightarrow \langle article \rangle \langle noun \rangle \langle verb \rangle$

 $\Rightarrow a \langle noun \rangle \langle verb \rangle$

 $\Rightarrow a \ cat \ \langle verb \rangle$

 \Rightarrow a cat runs











Language of the grammar:

```
L = \{ \text{``a cat runs''}, 
      "a cat walks",
      "the cat runs",
      "the cat walks",
      "a dog runs",
      "a dog walks",
      "the dog runs",
      "the dog walks" }
```



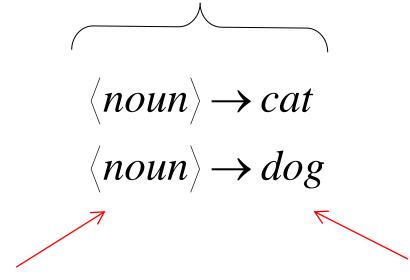






Notation

Production Rules



Variable

Terminal











Another Example

• Grammar:

$$S \rightarrow aSb$$

$$S \rightarrow \lambda$$

• Derivation of sentence

$$S \Rightarrow aSb \Rightarrow ab$$

$$S \rightarrow aSb \qquad S \rightarrow \lambda$$











• Grammar:

$$S \rightarrow aSb$$

$$S \rightarrow \lambda$$

• Derivation of sentence

aabb

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$$

$$S \rightarrow aSb \qquad S \rightarrow \lambda$$











Other derivations:

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbb$$

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb$$

$$\Rightarrow aaaaSbbbb \Rightarrow aaaabbbb$$











Language of the grammar

$$S \rightarrow aSb$$

$$S \to \lambda$$

$$L = \{a^n b^n : n \ge 0\}$$











More Notation



$$G = (V, T, S, P)$$

V: Set of variables

T: Set of terminal symbols

S: Start variable

P: Set of Production rules











Example

Grammar

$$S \rightarrow aSb$$

$$S \rightarrow \lambda$$

$$G = (V, T, S, P)$$

$$V = \{S\} \qquad T = \{a, b\}$$

$$P = \{S \to aSb, S \to \lambda\}$$





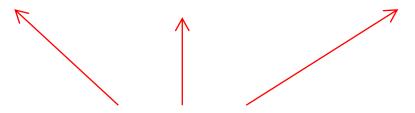


More Notation

- Sentential Form:
- A sentence that containsvariables and terminals

• Example:

 $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbb$



Sentential Forms









• We write:

$$S \Rightarrow aaabbb$$

• Instead of:

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbb$$











• In general we write:

$$w_1 \implies w_n$$

• If:

$$w_1 \Rightarrow w_2 \Rightarrow w_3 \Rightarrow \cdots \Rightarrow w_n$$











• By default:



 $w \implies w$











Example

Grammar

$$S \rightarrow aSb$$

$$S \to \lambda$$

Derivations

$$S \Rightarrow \lambda$$

*

$$S \Rightarrow ab$$

*

$$S \Rightarrow aabb$$

*

$$S \Rightarrow aaabbb$$











Example

Grammar

$$S \rightarrow aSb$$

$$S \rightarrow \lambda$$

Derivations

$$s \Rightarrow aaSbb$$









Another Grammar Example

• Grammar

$$S \rightarrow Ab$$

$$A \rightarrow aAb$$

$$A \rightarrow \lambda$$

Derivations:

$$S \Longrightarrow b \Longrightarrow$$

$$S \Longrightarrow ab \Longrightarrow abb \Longrightarrow abb$$

$$S \Longrightarrow Ab \Longrightarrow Abb \Longrightarrow aAbbb \Longrightarrow abbb$$





More Derivations

$$^{\bullet}S \Rightarrow Ab \Rightarrow aAbb \Rightarrow aaAbbb \Rightarrow aaaAbbbb$$

$$\Rightarrow$$
 aaaaAbbbbbb \Rightarrow aaaabbbbbb

$$S \Rightarrow aaaabbbbb$$

*

$$S \Rightarrow aaaaaaabbbbbbbb$$

$$S \Rightarrow a^n b^n b$$













Language of a Grammar

- For a grammar
- with start variable $\stackrel{\smile}{:}$:

$$L(G) = \{w: S \Longrightarrow w\}$$

String of terminals







Example

• For grammar

$$S \rightarrow Ab$$

$$A \rightarrow aAb$$

$$A \rightarrow \lambda$$

$$L(G) = \{a^n b^n b: n \ge 0\}$$

*

Since: $S \Rightarrow a^n b^n b$













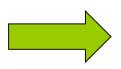
A Convenient Notation

$$\begin{array}{cccc}
A \to aAb \\
A \to \lambda
\end{array}$$

$$A \to aAb \mid \lambda$$

$$\langle article \rangle \rightarrow a$$

 $\langle article \rangle \rightarrow the$



 $\langle article \rangle \rightarrow a \mid the$











Linear Grammars











Linear Grammars

- Grammars with
- at most one variable at the right side
- of a production

• Examples:

$$S \rightarrow aSb$$

$$S \rightarrow \lambda$$

$$S \rightarrow Ab$$

$$A \rightarrow aAb$$

$$A \rightarrow \lambda$$











A Non-Linear Grammar

Grammar
$$G:$$

$$S \rightarrow SS$$

$$S \to \lambda$$

$$S \rightarrow aSb$$

$$S \rightarrow bSa$$

$$L(G) = \{w: n_a(w) = n_b(w)\}$$











Another Linear Grammar

• Grammar

$$\dot{G}$$
 $S \to A$ $A \to aB \mid \lambda$ $B \to Ab$

$$L(G) = \{a^n b^n : n \ge 0\}$$











Right-Linear Grammars

• All productions have form:

$$A \rightarrow xB$$

or

$$A \rightarrow x$$

 $S \rightarrow abS$

$$S \rightarrow a$$

string of terminals





• Example:







Left-Linear Grammars

• All productions have form:

$$A \rightarrow Bx$$

or

$$A \rightarrow x$$

• Example:

$$S \rightarrow Aab$$

$$A \rightarrow Aab \mid B$$

$$B \rightarrow a$$

string of terminals











Regular Grammars











Regular Grammars

- A regular grammar is any
- right-linear or left-linear grammar
- Examples:

$$G_1$$
 G_2 $S \rightarrow abS$ $S \rightarrow Aab$ $A \rightarrow Aab \mid B$ $B \rightarrow a$











Observation

Regular grammars generate regular languages

• Examples:

$$G_2$$

$$G_1$$

$$S \rightarrow Aab$$

$$S \rightarrow abS$$

$$A \rightarrow Aab \mid B$$

$$S \rightarrow a$$

$$B \rightarrow a$$

$$L(G_1) = (ab) * a$$

$$L(G_$$

$$L(G_2) = aab(ab) *$$





What is the purpose of using grammar in automata theory?

- a) To define the input alphabet for an automaton.
- b) To describe the set of strings accepted by an automaton.
- c) To specify the transition function of an automaton.
- d) To determine the number of states in an automaton.

Answer: b) To describe the set of strings accepted by an automaton.











Which of the following statements is true about context-free grammars?

- a) Context-free grammars can generate all types of languages.
- b) Context-free grammars can only generate regular languages.
- c) Context-free grammars can generate regular as well as non-regular languages.
- d) Context-free grammars cannot generate any languages.

Answer: c) Context-free grammars can generate regular as well as non-regular languages.











What is the role of non-terminal symbols in a grammar?

- a) They represent the terminal symbols of the language.
- b) They define the set of allowable productions in the grammar.
- c) They represent the starting symbol of the grammar.
- d) They specify the language accepted by the grammar.

Answer: b) They define the set of allowable productions in the grammar.a











Which of the following is true regarding the derivation process in a grammar?

- a) The derivation process starts from the start symbol and proceeds left to right.
- b) The derivation process starts from the start symbol and proceeds right to left.
- c) The derivation process can start from any non-terminal symbol.
- d) The derivation process involves only the terminal symbols of the grammar.

Answer: a) The derivation process starts from the start symbol and proceeds left to right.













Which of the following is used to describe the language generated by a grammar?

- a) Transition diagram
- b) Production rules
- c) State table
- d) Regular expression

Answer: b) Production rules











Question 1:

What is the role of terminals in the grammar of Theory of Computation?

Answer:

Terminals in the grammar of Theory of Computation represent the basic units or symbols of the language being defined. They are the elements that cannot be further decomposed within the grammar and typically correspond to specific symbols or tokens in the language.

Question 2:

What is the purpose of non-terminals in the grammar of Theory of Computation?

Answer:

Non-terminals in the grammar of Theory of Computation represent syntactic variables or placeholders that can be expanded into a sequence of terminals and/or non-terminals. They provide the structure and rules for generating valid sentences or expressions in the language.

Question 3:

What is the significance of production rules in the grammar of Theory of Computation?

Answer:

Production rules in the grammar of Theory of Computation define the transformations or derivations that can be applied to the non-terminals to generate valid expressions or sentences in the language. They specify how non-terminals can be replaced by a sequence of terminals and non-terminals allowing for the generation of valid language constructs.















Team - TOC







