Theoretical Computer Science

Tutorial Week 12

Prof. Andrey Frolov

Innopolis University

Agenda

Grammars

- **Computational linguistics**
- Chomsky Hierarchy
    - Regular grammars (type 3)
    - Context-Free grammars (type 2)
    - Context-Sensitive grammars (type 1)
    - Unrestricted grammars (type 0)

Computational linguistics

Grammars

- (proposition) $\to$ (noun)(verb)
- (noun) $\to$ a cat
- (noun) $\to$ a dog
- (noun) $\to$ a fish
- $\cdots$
- (verb) $\to$ meows
- (verb) $\to$ barks
- (verb) $\to$ swims
- $\cdots$

(proposition) $\to$ (noun)(verb) $\to$ a dog (verb) $\to$ a dog swims
(proposition) $\to$ (noun)(verb) $\to$ a cat (verb) $\to$ a cat barks
(proposition) $\to$ (noun)(verb) $\to$ a fish (verb) $\to$ a fish meows

$$A \to \alpha_1 \mid \alpha_2 \mid \cdots \mid \alpha_n$$

- $A \to \alpha_1$
- $A \to \alpha_2$
- $\cdots$
- $A \to \alpha_n$

Computational linguistics

Grammars

- (proposition) $\rightarrow$ (noun)(verb)
- (noun) $\rightarrow$ a cat | a dog | a fish | $\cdots$
- (verb) $\rightarrow$ meows | barks | swims | $\cdots$

(proposition), (noun), (verb) are called **non-terminal**
"a dog", "a cat", "meows", "barks", ... are called **terminal**

Computational linguistics

Grammars

- (proposition) $\to$ (noun)(verb)
- (noun) $\to$ a cat | a dog
- (verb) $\to$ sleeps | runs
- a cat (verb) $\to$ a cat meows
- a dog (verb) $\to$ a dog barks

(proposition) $\to$ (noun)(verb) $\to$ a dog (verb) $\to$ a dog barks
(proposition) $\to$ (noun)(verb) $\to$ a cat (verb) $\to$ a cat meows

Agenda

Grammars

- Computational linguistics
- **Chomsky Hierarchy**
  - Regular grammars (type 3)
  - Context-Free grammars (type 2)
  - Context-Sensitive grammars (type 1)
  - Unrestricted grammars (type 0)

Grammar: definition

Alphabets:

- V_N is the non-terminal alphabet;
- V_T is the terminal alphabet;
- $V = V_N \cup V_T$ the alphabet;
- $V_N \cap V_T = \emptyset$.

Grammar: definition

Rules:

- $P \subseteq (V^* \cdot V_N \cdot V^*) \times V^*$ is the (finite) set of rewriting rules of production, where $V = V_N \cup V_T$

A **production rule** $\alpha \to \beta$ is an element of P where

- $\alpha \in V^* \cdot V_N \cdot V^*$ is a sequence of symbols including at least one non-terminal symbol;
- $\beta \in V^*$ is a (potentially empty) sequence of (terminal or non-terminal) symbols.

Grammar: definition

Initial:

- $S \in V_N$ is called an initial symbol

Grammar: definition

Definition

A grammar is a tuple $\langle V_N, V_T, P, S \rangle$, where

- V_N is the non-terminal alphabet;
- V_T is the terminal alphabet;
- $P \subseteq (V^* \cdot V_N \cdot V^*) \times V^*$ is the (finite) set of rewriting rules of production, where $V = V_N \cup V_T$;
- $S \in V_N$ is a particular element called axiom or initial symbol.

Agenda

Grammars

- Computational linguistics
- Chomsky Hierarchy
  - **Regular grammars (type 3)**
  - Context-Free grammars (type 2)
  - Context-Sensitive grammars (type 1)
  - Unrestricted grammars (type 0)

Regular grammars (type 3)

Right regular grammar

A right regular grammar is a formal grammar $\langle V_N, V_T, P, S \rangle$ such that all the production rules in P are of one of the following forms:

1. $A \to b$, where $A \in V_N$ and $b \in V_T$;
2. $A \to bB$, where $A, B \in V_N$ and $b \in V_T \cup \{\epsilon\}$;
3. $A \to \epsilon$, where $A \in V_N$ and ϵ denotes the empty string.

Left regular grammar

A left regular grammar is a formal grammar $\langle V_N, V_T, P, S \rangle$ such that all the production rules in P are of one of the following forms:

1. $A \to b$, where $A \in V_N$ and $b \in V_T$;
2. $A \to B\text{b}$, where $A, B \in V_N$ and $b \in V_T \cup \{\epsilon\}$;
3. $A \to \epsilon$, where $A \in V_N$ and ϵ denotes the empty string.

Regular grammars (type 3)

Fact

Right regular = Left regular = Regular

Regular grammars (type 3)

Example 1

$L_1 = \{s \in \{a, b\}^* \mid a \text{ and } b \text{ alternating}\}$

- $S \to A$
- $S \to B$
- $A \to \mathrm{a}B$
- $A \to \epsilon$
- $B \to \mathrm{b}A$
- $B \to \epsilon$

$S \to A \to aB \to abA \to abaB \to aba\epsilon = aba$
$S \to B \to bA \to baB \to babA \to babaB \to baba$

Regular grammars (type 3)

Example 2

$$L_1 = \{a^{2n} \mid n \in \mathbb{N}\}$$

Rules

- $S \rightarrow aA$
- $A \rightarrow aS$
- $S \rightarrow \epsilon$

$$S \rightarrow aA \rightarrow aaS \rightarrow aa$$

$$S \rightarrow aA \rightarrow aaS \rightarrow aaaA \rightarrow aaaaS \rightarrow aaaa$$

Regular grammars (type 3)

Example 2

$$L_1 = \{a^{2n} \mid n \in \mathbb{N}\}$$

Rules

- $S \rightarrow aaS$
- $S \rightarrow \epsilon$

$$S \rightarrow aaS \rightarrow aa$$

$$S \rightarrow aaS \rightarrow aaaaS \rightarrow aaaa$$

Regular grammars (type 3)

Right regular grammar

A right regular grammar is a formal grammar $\langle V_N, V_T, P, S\rangle$ such that all the production rules in P are of one of the following forms:

1. $A \to b$, where $A \in V_N$ and $b \in V_T$;
2. $A \to bB$, where $A, B \in V_N$ and $b \in V_T \cup \{\epsilon\}$;
3. $A \to \epsilon$, where $A \in V_N$ and ϵ denotes the empty string.

Right regular grammar

A right regular grammar is a formal grammar $\langle V_N, V_T, P, S\rangle$ such that all the production rules in P are of one of the following forms:

1) $A \to s$, where $A \in V_N$ and $s \in V_T^*$;
2) $A \to sB$, where $A, B \in V_N$ and $s \in V_T^*$;

The same definitions?

Regular grammars (type 3)

Rules

- $A \to a_1 A'_1$
- $A'_1 \to a_2 A'_2$
- $\cdots$
- $A'_{k-1} \to a_k A'_k$
- $A'_k \to \epsilon B$

This rules equals to $A \to sB$, where $s = a_1 a_2 \ldots a_k$.

$$A \to a_1 A'_1 \to a_1 a_2 A'_2 \to \cdots \to a_1 a_2 \ldots a_k A'_k \to a_1 a_2 \ldots a_k B = sB$$

Agenda

Grammars

- Computational linguistics
- Chomsky Hierarchy
  - Regular grammars (type 3)
  - **Context-Free grammars (type 2)**
  - Context-Sensitive grammars (type 1)
  - Unrestricted grammars (type 0)

Context-Free grammars (type 2)

Context-Free grammar

Defined by rules of the form $A \to \gamma$ where A is a non-terminal and γ is a string of terminals and non-terminals.

Example of rules

1. $A \to b$, where $A \in V_N$ and $b \in V_T$ Yes
2. $A \to Bb$, where $A, B \in V_N$ and $b \in V_T$ Yes
3. $A \to BbB$, where $A, B \in V_N$ and $b \in V_T$ Yes
4. $A \to bBb$, where $A, B \in V_N$ and $b \in V_T$ Yes
5. $AB \to Bb$, where $A, B \in V_N$ and $b \in V_T$ NO

Context-Free grammars (type 2)

Example

$$L_1 = \{a^n b^n \mid n \in \mathbb{N}\}$$

Rules

$$S \rightarrow \epsilon$$

$$S \rightarrow aSb$$

$$S \rightarrow \epsilon$$

$$S \rightarrow aSb \rightarrow ab$$

$$S \rightarrow aSb \rightarrow aaSbb \rightarrow \cdots a^n S b^n \rightarrow a^n b^n$$

Agenda

Grammars

- Computational linguistics
- Chomsky Hierarchy
  - Regular grammars (type 3)
  - Context-Free grammars (type 2)
  - **Context-Sensitive grammars (type 1)**
  - Unrestricted grammars (type 0)

Context-Sensitive grammars (type 1)

The rules of the form $s_1 A s_2 \to s_1 \gamma s_2$, where A is a non-terminal and s_1, s_2 and γ are strings of terminals and non-terminals.

Example

Generate language $\{a^n b^n c^n | n > 0\}$

1. $S \to \mathrm{a}BC$
2. $S \to \mathrm{a}SBC$
3. $CB \to CZ$
4. $CZ \to WZ$
5. $WZ \to WC$
6. $WC \to BC$
7. $\mathrm{a}B \to \mathrm{ab}$
8. $\mathrm{b}B \to \mathrm{bb}$
9. $\mathrm{b}C \to \mathrm{bc}$
10. $\mathrm{c}C \to \mathrm{cc}$

$CB \to CZ \to WZ \to WC \to BC$

Context-Sensitive grammars (type 1)

The rules of the form $s_1 A s_2 \to s_1 \gamma s_2$, where A is a non-terminal and s_1, s_2 and γ are strings of terminals and non-terminals.

Example

Generate language $\{a^n b^n c^n | n > 0\}$

1. $S \to \mathrm{a}BC$
2. $S \to \mathrm{a}SBC$
3. $CB \to CZ$
4. $CZ \to WZ$
5. $WZ \to WC$
6. $WC \to BC$
7. $\mathrm{a}B \to \mathrm{ab}$
8. $\mathrm{b}B \to \mathrm{bb}$
9. $\mathrm{b}C \to \mathrm{bc}$
10. $\mathrm{c}C \to \mathrm{cc}$

$S \to aSBC \to aaBCBC \to aaBBCC \to aabBCC \to aabbCC \to aabbcC \to aabbcc$

Agenda

Grammars

- Computational linguistics
- Chomsky Hierarchy
  - Regular grammars (type 3)
  - Context-Free grammars (type 2)
  - Context-Sensitive grammars (type 1)
  - **Unrestricted grammars (type 0)**

Unrestricted grammars (type 0)

Unrestricted grammar (type 0)

A Unrestricted grammar is a formal grammar without any limitation on production rules.

The rules of the form $\alpha \to \beta$, where α and β are strings of non-terminals and terminals.

Example 1

Generate language $\{a^n b^n c^n \mid n > 0\}$

1. $S \to \mathrm{a}BC$
2. $S \to \mathrm{a}SBC$
3. $CB \to BC$
4. $\mathrm{a}B \to \mathrm{ab}$
5. $\mathrm{b}B \to \mathrm{bb}$
6. $\mathrm{b}C \to \mathrm{bc}$
7. $\mathrm{c}C \to \mathrm{cc}$

Unrestricted grammars (type 0)

Example 2

Generate language $\{a^n b^n c^n d^n \mid n > 0\}$

1. $S \to \mathrm{a}BCD$
2. $S \to \mathrm{a}SBCD$
3. $CB \to BC$
4. $DB \to BD$
5. $DC \to CD$
6. $\mathrm{a}B \to \mathrm{ab}$
7. $\mathrm{b}B \to \mathrm{bb}$
8. $\mathrm{b}C \to \mathrm{bc}$
9. $\mathrm{c}C \to \mathrm{cc}$
10. $\mathrm{c}D \to \mathrm{cd}$
11. $\mathrm{d}D \to \mathrm{dd}$

$S \to aSBCD \to aaBCDBCD \to aaBCBDCD \to aaBBCDCD \to aaBBCCDD \to aabBCCDD \to aabbCCDD \to aabbcCDD \to aabbccDD \to aabbccdD \to aabbccdd$

Chomsky Hierarchy

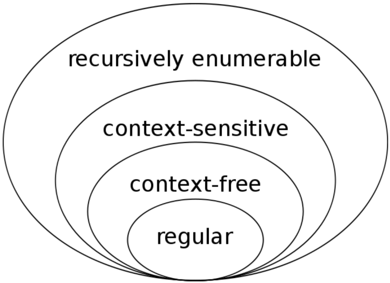
Classification of grammars by Chomsky: four types according to the form of production rules.

(type 3) Regular grammars

(type 2) Context-Free grammars

(type 1) Context-Sensitive grammars

(type 0) Unrestricted grammars



Grammars, languages and automata

Chomsky hierarchy	Grammars	Languages	Minimal automaton
Type-0	Unrestricted	Recursively enumerable	Turing machine
Type-1	Context-sensitive	Context-sensitive	LBA
Type-2	Context-free	Context-free	NDPDA
Type-3	Regular	Regular	FSA

Thank you for your attention!