CSE-233 : Section A

Summer 2020

Context Free Grammar

Reference:

Book2 Chapter 2.1

Md. Saidul Hoque Anik
anik@cse.uiu.ac.bd

Can we design a DFA/NFA to detect this language?

$$L = \{0^n 1^n \mid n \geq 0\}$$

Will the design have **finite** states?

Can we design a DFA/NFA to detect the following?

$$L = \{0^n 1^n \mid n \geq 0\}$$

Will the design have **finite** states?

This language is not a regular language!

Regular Grammar

We can use production rules/grammar to describe a language

$$S \rightarrow 0A$$
$$A \rightarrow 1B$$
$$B \rightarrow \in$$

What will the following rules generate?

Regular Grammar

We can use production rules/grammar to describe a language

Start symbol, we must start from this variable

$$S \rightarrow 0A$$
$$A \rightarrow 1B$$
$$B \rightarrow \in$$

What will the following rules generate?

Regular Grammar

We can use production rules/grammar to describe a language

$$S \rightarrow 0A$$
$$A \rightarrow 1B$$
$$B \rightarrow \in$$

What will the following rules generate? Ans: 01

Regular Grammar

Grammar for the string “01”:

$$S \rightarrow 0A$$
$$A \rightarrow 1B$$
$$B \rightarrow \in$$

Regular Grammar

Grammar for the string “01” or “11”:

$$S \rightarrow 0A \mid 1A$$
$$A \rightarrow 1B$$
$$B \rightarrow \in$$

Regular Grammar

What will be the grammar for the string staring with “0”?

$$S \rightarrow 0A \mid 1A$$
$$A \rightarrow 1B$$
$$B \rightarrow \in$$

Regular Grammar

Grammar for the string staring with “0”:

$$S \rightarrow 0A$$
$$A \rightarrow 1A \mid 0A \mid B$$
$$B \rightarrow \in$$

Regular Grammar

What will be the grammar for the string <u>ending</u> with “0”?

$$S \rightarrow 0A$$
$$A \rightarrow 1A \mid 0A \mid B$$
$$B \rightarrow \in$$

Regular Grammar

Grammar for the string <u>ending</u> with "0"

$$S \rightarrow A$$
$$A \rightarrow 1A \mid 0A \mid 0$$

Regular Grammar

Another Grammar for the string <u>ending</u> with “0”

$$S \rightarrow A0$$
$$A \rightarrow A0 \mid A1 \mid \in$$

Regular Grammar

Grammar for the string ending with “010”

$$S \rightarrow A$$
$$A \rightarrow 1A \mid 0A \mid 010$$

Task

What will be the grammar for the strings <u>having substring 010?</u>

Solution

What will be the grammar for the strings having substring 010?

$$S \rightarrow A$$
$$A \rightarrow 1A \mid 0A \mid 010B$$
$$B \rightarrow 1B \mid 0B \mid \in$$

Regular Grammar

A grammar is regular if in the production rule the variables are always on the left or always on the right

Right Regular Grammar:

$$S \to A$$
$$A \to 1A \mid 0A \mid 010B$$
$$B \to 1B \mid 0B \mid \epsilon$$

Left Regular Grammar:

$$S \to A0$$
$$A \to A0 \mid A1 \mid \epsilon$$

Limitation of Regular Grammar

A regular grammar cannot keep count

$$L = \{0^n 1^n \mid n \geq 0\}$$

Context Free Grammar

A grammar where in the production rule, the variable can be in any where (not just strictly left/right).

$$L = \{0^n 1^n \mid n \geq 0\}$$

Can you write the CF grammar for the language mentioned above?

Context Free Grammar

A grammar where in the production rule, the variable can be anywhere (not just strictly on left/right).

$$L = \{0^n 1^n \mid n \geq 0\}$$

Grammar for the language mentioned above:

$$S \rightarrow 0S1$$
$$S \rightarrow \epsilon$$

Context Free Grammar

- A **different model** for describing languages
- The language is specified by production/substitution rules (Grammar) that tell how strings can be obtained, e.g.

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$

A, B **are variables/non-terminal**

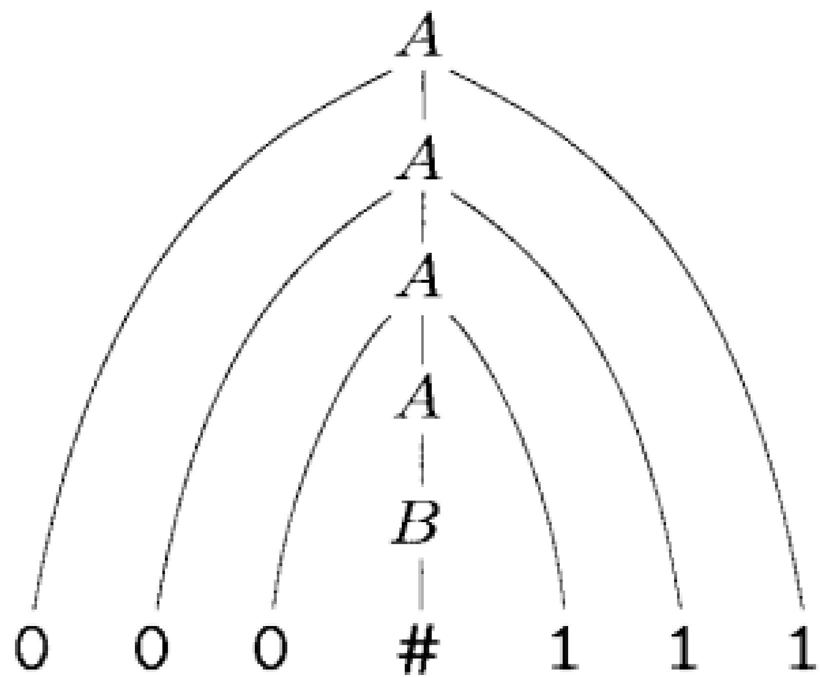
$0, 1, \#$ **are terminals**

A **is the start variable**

- Using these rules, we can derive strings like this:

$$A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 000A111 \Rightarrow 000B111 \Rightarrow 000\#111$$

Parse Tree for 000#000

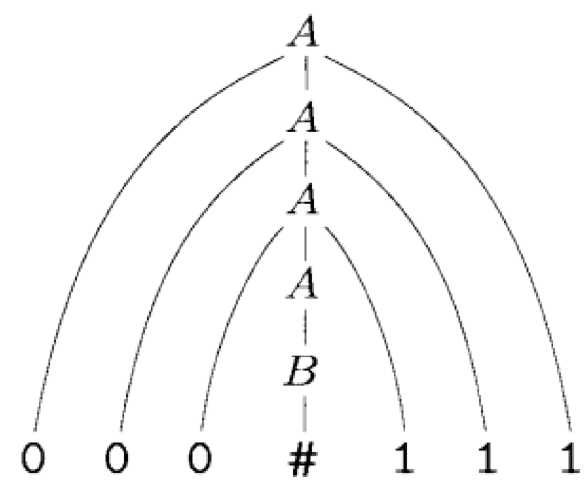


$$A \longrightarrow 0A1$$
$$A \longrightarrow B$$
$$B \longrightarrow \#$$

Derivation for 000#000

```
A  = 0A1
   = 00A11
   = 000A111
   = 000B111
   = 000#111
```

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$



Similar to Human Grammar

$$
\begin{aligned}
\langle \text{SENTENCE} \rangle &\to \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
\langle \text{NOUN-PHRASE} \rangle &\to \langle \text{CMPLX-NOUN} \rangle \mid \langle \text{CMPLX-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{VERB-PHRASE} \rangle &\to \langle \text{CMPLX-VERB} \rangle \mid \langle \text{CMPLX-VERB} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{PREP-PHRASE} \rangle &\to \langle \text{PREP} \rangle \langle \text{CMPLX-NOUN} \rangle \\
\langle \text{CMPLX-NOUN} \rangle &\to \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \\
\langle \text{CMPLX-VERB} \rangle &\to \langle \text{VERB} \rangle \mid \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \\
\langle \text{ARTICLE} \rangle &\to \texttt{a} \mid \texttt{the} \\
\langle \text{NOUN} \rangle &\to \texttt{boy} \mid \texttt{girl} \mid \texttt{flower} \\
\langle \text{VERB} \rangle &\to \texttt{touches} \mid \texttt{likes} \mid \texttt{sees} \\
\langle \text{PREP} \rangle &\to \texttt{with}
\end{aligned}
$$

```
a boy sees
the boy sees a flower
```

Derivation of “a boy sees”

$$
\begin{aligned}
\langle\text{SENTENCE}\rangle &\Rightarrow \langle\text{NOUN-PHRASE}\rangle\langle\text{VERB-PHRASE}\rangle\\
&\Rightarrow \langle\text{CMPLX-NOUN}\rangle\langle\text{VERB-PHRASE}\rangle\\
&\Rightarrow \langle\text{ARTICLE}\rangle\langle\text{NOUN}\rangle\langle\text{VERB-PHRASE}\rangle\\
&\Rightarrow \texttt{a}\ \langle\text{NOUN}\rangle\langle\text{VERB-PHRASE}\rangle\\
&\Rightarrow \texttt{a boy}\ \langle\text{VERB-PHRASE}\rangle\\
&\Rightarrow \texttt{a boy}\ \langle\text{CMPLX-VERB}\rangle\\
&\Rightarrow \texttt{a boy}\ \langle\text{VERB}\rangle\\
&\Rightarrow \texttt{a boy sees}
\end{aligned}
$$

Definition

A ***context-free grammar*** is a 4-tuple (V, Σ, R, S), where

1. V is a finite set called the ***variables***,
2. Σ is a finite set, disjoint from V, called the ***terminals***,
3. R is a finite set of ***rules***, with each rule being a variable and a string of variables and terminals, and
4. $S \in V$ is the start variable.

Note:

- *R* is a set of productions or substitution rules of the form-

  $$A \rightarrow \alpha$$

  where A is a symbol in *V* and a is a string over $V \cup T$

- Usually Capital letter used for non-terminals and Small letters for terminals.

Term: ‘yield’

If u, v, and w are strings of variables and terminals, and $A \rightarrow w$ is a rule of the grammar, we say that uAv ***yields*** uwv, written $uAv \Rightarrow uwv$.

Term: ‘derive’

If u = v

or, u => u_1 => u_2 => … => u_k => v

We say u derives v written $u \overset{*}{\Rightarrow} v$

Definition

A ***context-free grammar*** is a 4-tuple (V, Σ, R, S), where

1. V is a finite set called the ***variables***,
2. Σ is a finite set, disjoint from V, called the ***terminals***,
3. R is a finite set of ***rules***, with each rule being a variable and a string of variables and terminals, and
4. $S \in V$ is the start variable.

The ***language of the grammar*** is $\{w \in \Sigma^* \mid S \stackrel{*}{\Rightarrow} w\}$.

Usage of grammar

Essential for understanding the meaning of computer instructions. Used in compilers.

code: $(2 + 3) * 5$

meaning: "add 2 and 3, and then multiply by 5"

If we understand which rule it's matching, we can follow the sequence.

Example

Consider grammar $G_3 = (\{S\}, \{\mathtt{a}, \mathtt{b}\}, R, S)$. The set of rules, R, is

$$S \rightarrow \mathtt{a}S\mathtt{b} \mid SS \mid \varepsilon.$$

This grammar generates strings such as abab, aaabbb, and aababb. You can see more easily what this language is if you think of a as a left parenthesis "(" and b as a right parenthesis ")". Viewed in this way, $L(G_3)$ is the language of all strings of properly nested parentheses.

Checking if a string is in grammar

1. Start with the start symbol and choose the closest production that matches to the given string.
2. Replace the variables with its most appropriate production. Repeat the process until the string is generated or until no other production matches.

Example

Verify if the following grammar

$$S \to 0B \mid 1A$$
$$A \to 0 \mid 0S \mid 1AA \mid \varepsilon$$
$$B \to 1 \mid 1S \mid 0BB$$

generates the string: 00110101

Example 2

Verify if the following grammar

$S \rightarrow aAb$
$A \rightarrow aAb \mid \varepsilon$

generates the string: aabbb

Example 3

$A \to 0A1 \mid B$
$B \to \#$

variables: A, B
terminals: 0, 1, #
start variable: A

- Is the string 00#11 in L?
- How about 00#111, 00#0#1#11?

- The language of this CFG?

$$L = \{0^n\#1^n : n \geq 0\}$$

Example 4

$S \to SS \mid (S) \mid \varepsilon$

convention: variables in uppercase, terminals in lowercase, start variable first

- Give derivations of (), (()())

$$
\begin{aligned}
S &\Rightarrow (S) && \text{(rule 2)}\\
&\Rightarrow () && \text{(rule 3)}
\end{aligned}
$$

$$
\begin{aligned}
S &\Rightarrow (S) && \text{(rule 2)}\\
&\Rightarrow (SS) && \text{(rule 1)}\\
&\Rightarrow ((S)S) && \text{(rule 2)}\\
&\Rightarrow ((S)(S)) && \text{(rule 2)}\\
&\Rightarrow (()(S)) && \text{(rule 3)}\\
&\Rightarrow (()()) && \text{(rule 3)}
\end{aligned}
$$

- How about ())?

Example 5

When we have multiple productions with the same variable on the left like

$E \to E + E$

$E \to E * E$

$E \to (E)$

$E \to N$

$N \to 0N$

$N \to 1N$

$N \to 0$

$N \to 1$

Variables: E, N

Terminals: +, *, (,), 0, 1

Start variable: E

we can write this in shorthand as

$E \to E + E \mid E * E \mid (E) \mid N$

$N \to 0N \mid 1N \mid 0 \mid 1$

Task : Show derivation for (1 + 10) * 1

Example 5

Task : Show derivation for (1 + 10) * 1

$$
\begin{aligned}
E &\Rightarrow E * E \\
&\Rightarrow (E) * E \\
&\Rightarrow (E) * N \\
&\Rightarrow (E + E) * N \\
&\Rightarrow (E + E) * 1 \\
&\Rightarrow (E + N) * 1 \\
&\Rightarrow (N + N) * 1 \\
&\Rightarrow (N + 1N) * 1 \\
&\Rightarrow (N + 10) * 1 \\
&\Rightarrow (1 + 10) * 1
\end{aligned}
$$

derivation

$$\alpha \Rightarrow \beta$$

means β can be obtained from α with one production

$$\alpha \stackrel{*}{\Rightarrow} \beta$$

means β can be obtained from α after zero or more productions

Practice

Write the grammar for the following languages-

1. Numbers without leading zeros, e.g., 109, 0 but not 019
2. The language $L = \{a^n b^n c^m d^m \mid n \geq 0, m \geq 0\}$
3. The language $L = \{a^n b^m c^m d^n \mid n \geq 0, m \geq 0\}$
4. Linear equations over x, y, z, like:
    - $x + 5y - z = 9$
    - $11x - y = 2$

Task

- Write a CFG for the language (0 + 1)*111

Task

- Write a CFG for the language (0 + 1)*111

$$S \rightarrow A111$$
$$A \rightarrow \varepsilon \mid 0A \mid 1A$$

- Can you do so for every regular language?

Task

- Write a CFG for the language (0 + 1)*111

$$S \to A111$$
$$A \to \varepsilon \mid 0A \mid 1A$$

- Can you do so for every regular language?

Yes. Every regular language is context-free

Conversion from RE to CFG

regular expression	CFG
$\emptyset$	grammar with no rules
ε	$S \to \varepsilon$
a (alphabet symbol)	$S \to a$
$E_1 + E_2$	$S \to S_1 \mid S_2$
$E_1 E_2$	$S \to S_1 S_2$
E_1^*	$S \to S S_1 \mid \varepsilon$

In all cases, S becomes the new start symbol

Can we convert any CFG to RE?

CFG Design Practice

- Design CFG for a^n|n>=0

S-> aS | ϵ

- Design CFG for $(a+b)^*$
- What about $(a+b)^+$ or $(a+b)^3$?

CFG Design Practice

- Design CFG for length of all string with 2.

S-> aa | ab | ba |bb

S-> AA

A-> a | b

- Design CFG for length at least 2.

CFG Design Practice

(Solution) Design CFG for length at least 2.

- S -> AAB
- A -> a|b
- B -> aB | bB | ε

- Now design for at most 2.

CFG Design Practice

Write CFG with all string starting with a end with b.

S -> aAb
A -> aA | bA | ε

Design CFG for even length string.

CFG Design Practice

(Solution) Design CFG for even length string.

S -> AS | ε

A -> BB

B -> a | b

CFG Design Practice

Design CFG for $a^n b^n c^m |\, n, m \geq 1$

CFG Design Practice

(Solution) Design CFG for $a^n b^n c^m | \, n, m \geq 1$

- S -> AB
- A -> aAb | ab
- B -> cB | c