10/07/2025

Context Free Grammer

What we have learnt so far

- DFA/NFA computers with extremely limited memory.
- Regular Expressions
  Recipe to describe languages that are regular.

How about creating a slightly powerful computational model?

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

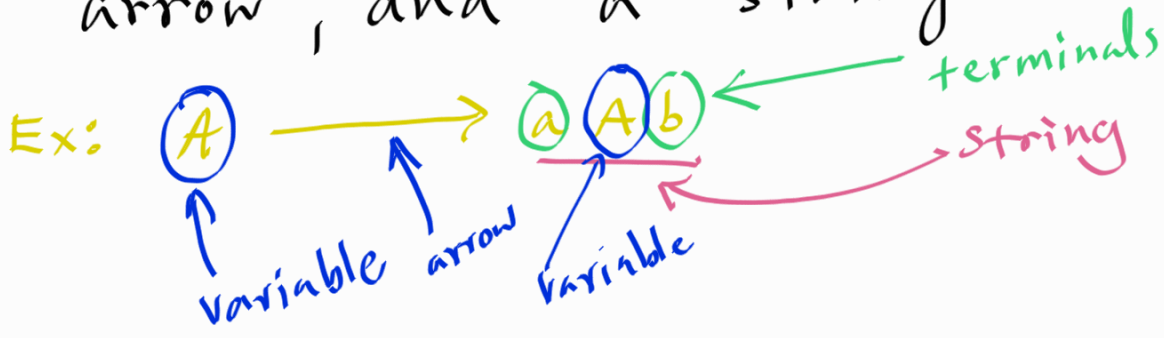
Instead going directly to the new computational model, let's try to learn about something called context free grammar.

context free grammar to context free languages is what regular expressions are to regular languages.

☆ context free grammar is a collection of substitution rules also known as productions.

☆ Each rule appears as a line in the grammar.

☆ Each line comprises of a symbol, arrow, and a string.



☆ The string can consist of variables, and other symbols called terminals.

☆ Variables are represented using capital letters, terminals are represented using lowercase letters, numbers or special symbols.

☆ one variable is designated as the starting variable.

we know that following language A is not regular.

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

1. $S \longrightarrow 0S1$
2. $S \longrightarrow B$
3. $B \longrightarrow \varepsilon$

$$S \Rightarrow 0\underline{S}1 \Rightarrow 00\underline{S}11 \Rightarrow 000\underline{S}111 \Rightarrow 000\underline{B}111$$
$$\Downarrow$$
$$000111$$

The sequence of substitutions to obtain a string is called a derivation.

$$S \Rightarrow B \Rightarrow \varepsilon$$

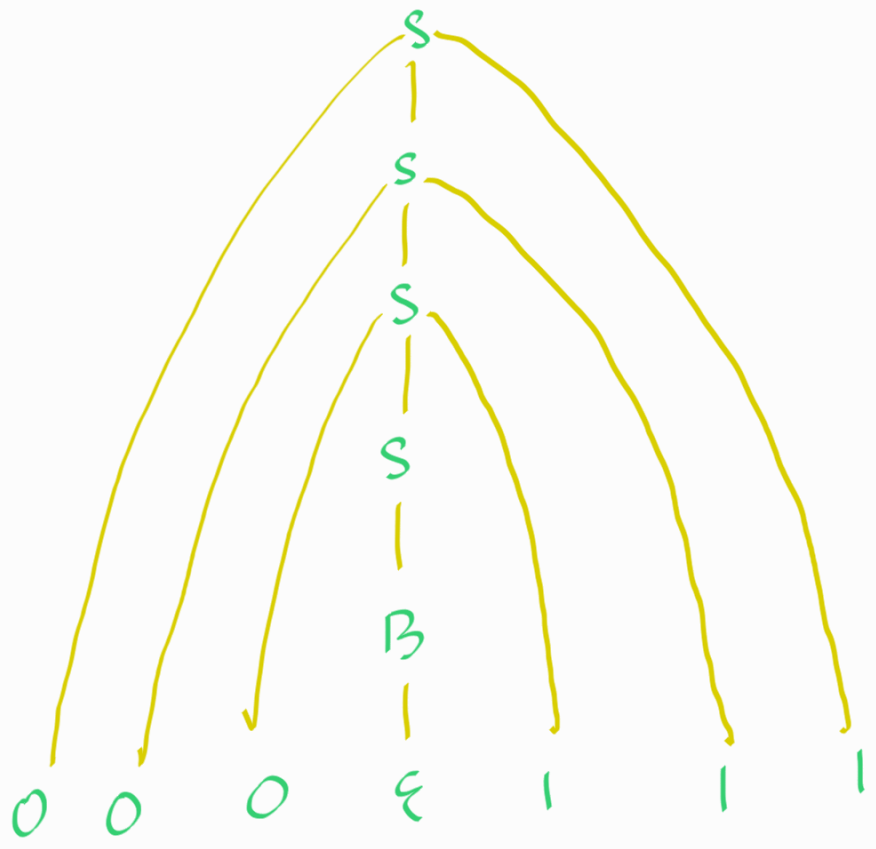
we can represent this using a parse tree as well.

$S \rightarrow 0S1$

$S \rightarrow B$

$B \rightarrow \varepsilon$

000111



Any language that can be generated by some context free grammar is called a context free language.

Just like regular expressions and regular languages,

CFG is a recipe, CFL is the meal

The language created by a context free grammar G is denoted as L(G)

Definition

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

1. V is the set of variables.
2. Σ is a finite set, called terminals.
3. R is a set of rules, with each rule being a variable and a string of variables and terminals.
4. S is the starting variable.

Example

$G_3 = (\{S\}, \{a, b\}, R, S)$

R: 1. $S \longrightarrow aSb$

2. $S \longrightarrow SS$

3. $S \longrightarrow \varepsilon$

$$S \longrightarrow aSb \mid SS \mid \varepsilon$$

$$S \overset{①}{\Rightarrow} aSb \overset{②}{\Rightarrow} aSSb \overset{③}{\Rightarrow} aa\underline{S}bSb \overset{③}{\Rightarrow} aaabSb \overset{③}{\Rightarrow} aaabb$$

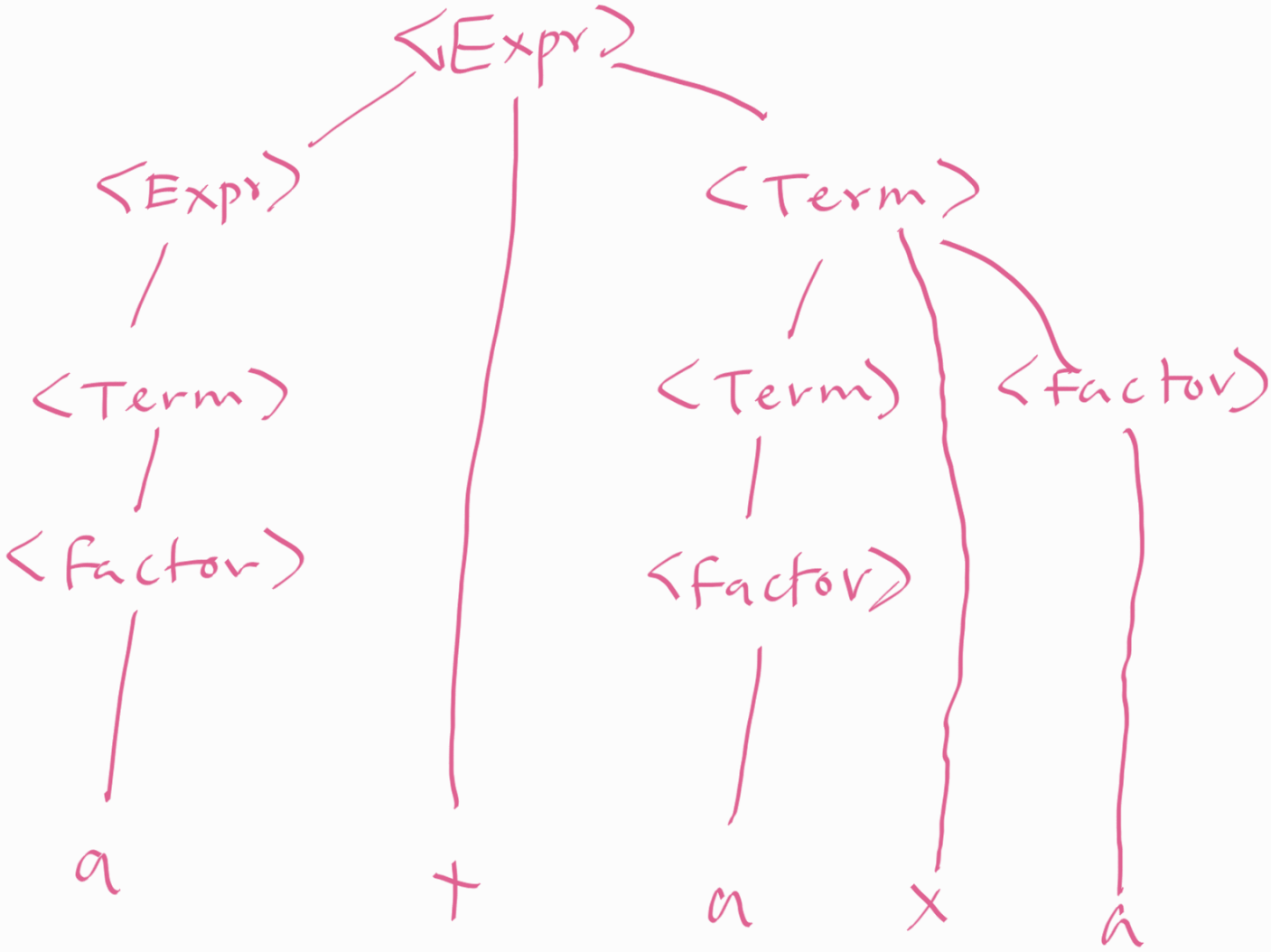
This grammar generates all strings with proper nested parenthesis

$G_4 = (V, \Sigma, R, \langle Expr \rangle)$

$V = \{\langle Expr \rangle, \langle Term \rangle, \langle Factor \rangle\}$

$\Sigma = \{a, +, \times, (,)\}$

1. $\langle Expr \rangle \rightarrow \langle Expr \rangle + \langle Term \rangle \mid \langle Term \rangle$
2. $\langle Term \rangle \rightarrow \langle Term \rangle \times \langle factor \rangle \mid \langle factor \rangle$
3. $\langle Factor \rangle \rightarrow (\langle Expr \rangle) \mid a$

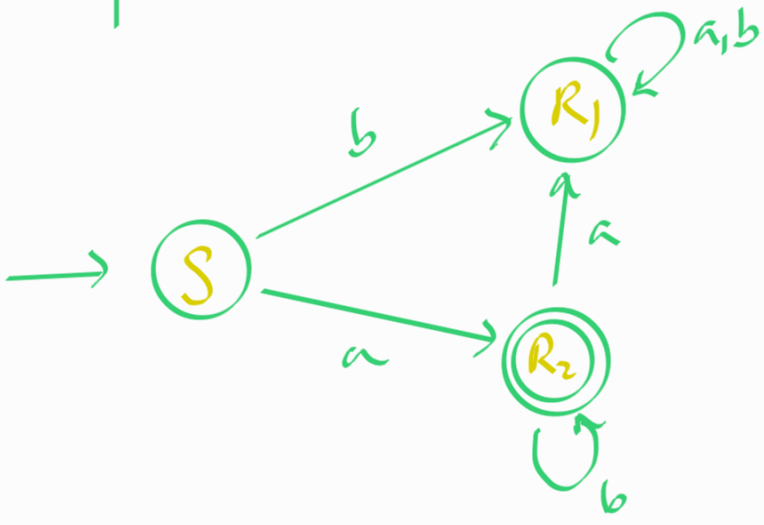


Side Note!

We can easily represent any regular language using context free grammar.

Suppose you have a DFA, you can easily convert it to a CFG.

Example: DFA ⟶ Context Free Grammar



$S \longrightarrow bR_1$

$S \longrightarrow aR_2$

$R_1 \longrightarrow aR_1 \mid bR_1$

$R_2 \longrightarrow bR_2 \mid aR_1$

$R_2 \longrightarrow \varepsilon$

* make a variable for each state q_i
* Create a grammar rule for each transition in the DFA.
* Use the variable that you created for the start state as your start variable.
* Add $R_i \longrightarrow \varepsilon$ for each accept state R_i

- Designing context-free grammar is tricky.

- You need to be able to identify recursive structure in your language.

- Some CFL are union of simpler CFLs.

- we can start by constructing CFG for smaller CFL and then merge them together.

Design a context free grammar.

$\sum = \{a, b\}$

Ex 01: $A = \{ w \mid$ the length of w is even $\}$

ε, ax, bX, $aa\checkmark$, $bb\checkmark$, ab, ba

$S \longrightarrow aSa \mid aSb \mid bSa \mid bSb \mid \varepsilon$

$S \longrightarrow aaS \mid abS \mid baS \mid bbS \mid \varepsilon$

Ex 02: $B = \{ w \mid w$ starts and ends with the same symbol $\}$

$\Sigma = \{a, b\}$

$\varepsilon, a, b, a^*, b^*, a\_bX, b\_aX$

a_a ✓

b_b ✓

$S \longrightarrow aBa \mid bBb \mid a \mid b \mid \varepsilon$

$B \longrightarrow aB \mid bB \mid \varepsilon$