Equivalence with context-free grammars I

- Language context free $\Leftrightarrow$ recognized by pushdown automata
- For the left-hand side, recall that by definition a language is context-free if it is constructed by some CFG
- Thus, the proof is equivalent to

  Language by CFG $\Leftrightarrow$ recognized by pushdown automata
- For the proof, one direction is easier, while the other is harder
- As usual, we do the easier one ($\Rightarrow$) first

CFL $\rightarrow$ PDA I

- Given a CFG, we find a PDA to simulate this grammar
- Two keys:

  stack

  nondeterminism: different substitutions
- We do the proof by an example
- Suppose we are given the following CFG

$$
\begin{aligned}
S &\rightarrow aTb \mid b \\
T &\rightarrow Ta \mid \epsilon
\end{aligned}
$$

CFL $\rightarrow$ PDA II

- Idea: for rule substitution, we replace the left-hand side variable with the right-hand side string
- That is, in PDA, we

  pop up the left-hand side variable

  and

  push right-hand side to stack

  in a reversed way
- For example, we have

$$S \rightarrow aTb$$

CFL $\rightarrow$ PDA III

Then

$$S \text{ is popped}$$

and

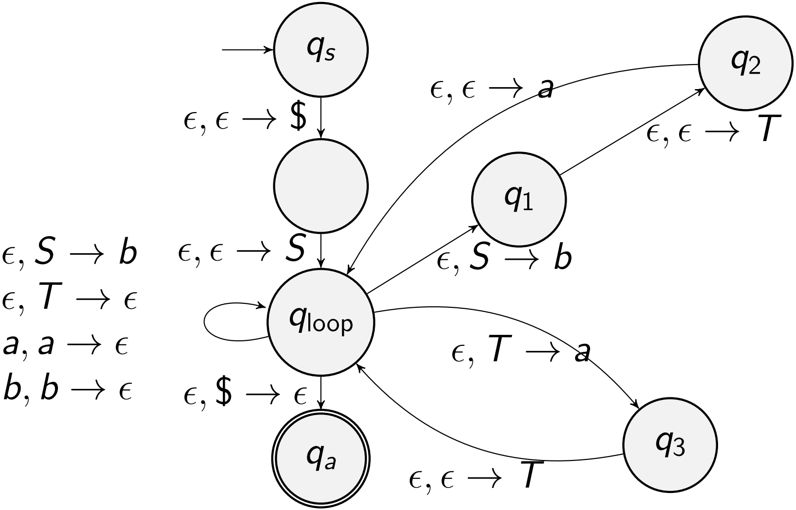
$$b, T, a$$

are sequentially pushed

CFL $\rightarrow$ PDA IV

- A PDA can be as follows

CFL $\to$ PDA V


q_s
$\epsilon, \epsilon \to \$$
$\epsilon, \epsilon \to S$
$\epsilon, S \to b$
$\epsilon, T \to \epsilon$
$a, a \to \epsilon$
$b, b \to \epsilon$
q_{loop}
$\epsilon, \$ \to \epsilon$
q_a
$\epsilon, \epsilon \to a$
q_2
$\epsilon, \epsilon \to T$
q_1
$\epsilon, S \to b$
$\epsilon, T \to a$
q_3
$\epsilon, \epsilon \to T$

CFL $\rightarrow$ PDA VI

- We use \$ to ensure that before accepting any string, stack is empty
- Then the start variable S is pushed
- The state q_{loop} is the main place to handle rules and process input characters
- Besides rules from CFG, we need

$$a, a \rightarrow \epsilon$$
$$b, b \rightarrow \epsilon.$$

Otherwise, input characters are never processed

CFL $\rightarrow$ PDA VII

- Consider an example sequence $aaaab$

CFL $\rightarrow$ PDA VIII

$$q_{\text{start}} \xrightarrow{\epsilon} q_{\text{loop}}, \{S, \$\} \xrightarrow{\epsilon} q_1, \{b, \$\} \xrightarrow{\epsilon} q_2, \{T, b, \$\}$$
$$\xrightarrow{\epsilon} q_{\text{loop}}, \{a, T, b, \$\} \xrightarrow{a} q_{\text{loop}}, \{T, b, \$\}$$
$$\xrightarrow{\epsilon} q_3, \{a, b, \$\} \xrightarrow{\epsilon} q_{\text{loop}}, \{T, a, b, \$\}$$
$$\xrightarrow{\epsilon} q_3, \{a, a, b, \$\} \xrightarrow{\epsilon} q_{\text{loop}}, \{T, a, a, b, \$\}$$
$$\xrightarrow{\epsilon} q_3, \{a, a, a, b, \$\} \xrightarrow{\epsilon} q_{\text{loop}}, \{T, a, a, a, b, \$\}$$
$$\xrightarrow{\epsilon} q_{\text{loop}}, \{a, a, a, b, \$\} \xrightarrow{a} q_{\text{loop}}, \{a, a, b, \$\}$$
$$\xrightarrow{a} q_{\text{loop}}, \{a, b, \$\} \xrightarrow{a} q_{\text{loop}}, \{b, \$\}$$
$$\xrightarrow{b} q_{\text{loop}}, \{\$\} \xrightarrow{\epsilon} q_{accept}$$

CFL $\rightarrow$ PDA IX

- Even with a non-deterministic setting, we ensure that only strings generated by this CFG can be accepted by the PDA
    1. A string is accepted only if all characters are processed (this is part of the PDA definition!)
    2. We have \$ to ensure that the stack is empty in the end