Lecture 3: Context-Free Languages and Pushdown Automata

Lecturer: Renjie Yang

3.1 Grammar

Sample English grammar rules:
sentence → subject   predicate
subject → article   adjective   noun
predicate → verb   object

Example The big dog chased the cat.

Definition 3.1 *A grammar is a 4-tuple $\{N, T, R, S\}$ consists of:*

- *A finite set N of grammar symbols called non-terminals;*

- *A finite set T of symbols called terminals, such that $N \cap T = \phi$;*

- *A finite set R of grammar rules of the form $\gamma \to \delta$ where γ and δ are strings over the symbol set $N \cup T$ with the following restrictions:*

  - *γ is not the empty string;*

  - *There is at least one production with S alone on the left hand side;*

  - *Each non-terminal must appear on the left hand side of some grammar rule;*

- *A non-terminal symbol S called start symbol.*

Example Let $\Sigma = \{a, b, c\}$, S is the start symbol. The following rules is a grammar for Σ^*:
$S \to \epsilon$
$S \to aS$
$S \to bS$
$S \to Sc$

Example Languages and their corresponding grammar:
$\{a^n \mid n \in \mathbb{N}\}$ $S \to aS | \epsilon$
$\{a^n b^n \mid n \in \mathbb{N}\}$ $S \to aSb | \epsilon$
$\{(ab)^n \mid n \in \mathbb{N}\}$ $S \to abS | \epsilon$

Note A grammar is called a regular grammar if each of its grammar rule takes one of the following forms where the uppercase letters are non-terminals and w is a non-empty string of terminals:

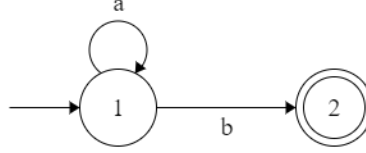- $S \to \epsilon$

- $S \to w$

- $S \to T$

- $S \rightarrow wT$

Note A regular language can be written by FSM, regular expression, or regular grammar. For example,

a^*b

$S \rightarrow aS; S \rightarrow b$

and the following FSM

all express the same language $\{b, ab, aab, aaab, \cdots\}$

3.2 Context-Free Grammar

Definition 3.2 *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;*

4. *$S \in V$ is the start variable.*

Definition 3.3 *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 $uAv \Rightarrow uwv$. Say that u derives v, written $u \overset{*}{\Rightarrow} v$, if $u = v$ or if a sequence $u_1, u_2, \ldots, u_k)$ exists for $k \geq 0$ and*

$$u \Rightarrow u_1 \Rightarrow u_2 \Rightarrow \cdots \Rightarrow u_k \Rightarrow u.$$

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

Definition 3.4 *A context-free language is a language generated by a context-free grammar.*

Example $S \rightarrow (S)|SS|\epsilon$

Example $\{0^n 1^n | n \geq 0\}$: $S \rightarrow \epsilon | 0S1$

Example $L = \{w|w \in \{0, 1\}^*\}$ and the number of 0's equals the number of 1's.

$S \rightarrow 0A|1B|\epsilon$
$A \rightarrow 1S|0AA$
$B \rightarrow 0S|1BB$

$S \rightarrow SAB|\epsilon$
$A \rightarrow 0S1|\epsilon$
$B \rightarrow 1S0|\epsilon$

Definition 3.5 *If a grammar generates the same string in several different ways, we say that the string is derived ambiguously in that grammar. If a grammar generates some string ambiguously, we say that the grammar is ambiguous.*

Example $S \to S + S | S \times S | a$

Theorem 3.6 *Every regular language is context-free.*

Proof: Given a DFA for the regular language, we can construct a context-free grammar that generates the same language through the following steps:

- make a variable for each state;
- make the variable for the starting state the starting variable;
- make a rule for each edge;
- Add an epsilon rule for each accept state.

∎

Definition 3.7 *A context-free grammar is in Chomsky normal form if every rule is of the form*

$$A \to BC$$
$$A \to a$$

where a is any terminal and A, B, and C are any variables, except that B and C may not be the start variable. In addition, we permit the rule $S \to \epsilon$, where S is the start variable.

Example
$S \to ASA | aB$
$A \to B | S$
$B \to b | \epsilon$

Theorem 3.8 *Context-free languages are closed under union operation.*

Proof: Let L_1 and L_2 be two context-free languages. We can construct a context-free grammar for the union of the two by adding a grammar rule: $S_0 \to S_1 | S_2$, where S_1 and S_2 are the start symbol for L_1 and L_2 respectively. ∎

Theorem 3.9 *Context-free languages are closed under concatenation operation.*

Proof: Let L_1 and L_2 be two context-free languages. We can construct a context-free grammar for the union of the two by adding a grammar rule: $S_0 \to S_1 S_2$, where S_1 and S_2 are the start symbol for L_1 and L_2 respectively. ∎

Example A context-sensitive grammar: $L = \{1^n 2^n 3^n | \, n \geq 1\}$
$S \to 1SBC$
$S \to \epsilon$
$CB \to HB$

$HB \rightarrow HC$
$HC \rightarrow BC$
$1B \rightarrow 12$
$2B \rightarrow 22$
$2C \rightarrow 23$
$3C \rightarrow 33$

Note There could be more than one different grammars for the same language.
$S \rightarrow aS|aaS|b$

3.3 Pushdown Automata

Definition 3.10 *A pushdown automaton is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$, where Q, Σ, Γ, F are all finite sets:*

1. *Q is the set of states,*

2. *Σ is the input alphabet,*

3. *Γ is the stack alphabet,*

4. *$\delta : Q \times \Sigma_\epsilon \times \Gamma_\epsilon \rightarrow \wp(Q \times \Gamma_\epsilon)$ is the transition function,*

5. *$q_0 \in Q$ is the start state,*

6. *$F \subset Q$ is the set of accept states.*

Definition 3.11 *A pushdown automaton $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ accepts input w if w can be written as $w = w_1 w_2 \cdots w_m$, where each $w_i \in \Sigma_\epsilon$ and sequences of states $r_0, r_1, \ldots, r_m \in Q$ and strings $s_0, s_1, \ldots, s_m \in \Gamma^*$ exist that satisfy the following three conditions:*

1. *$r_0 = q_0$ and $s_0 = \epsilon$.*

2. *For $i = 0, \ldots, m - 1$, we have $(r_{i+1}, b) \in \delta(r_i, w_{i+1}, a)$, where $s_i = at$ and $s_{i+1} = bt$ for some $a, b \in \Sigma_\epsilon$ and t in Γ^*.*

3. *$r_m \in F$.*

Example Let M be $(Q, \Sigma, \Gamma, \delta, q_0, F)$, where
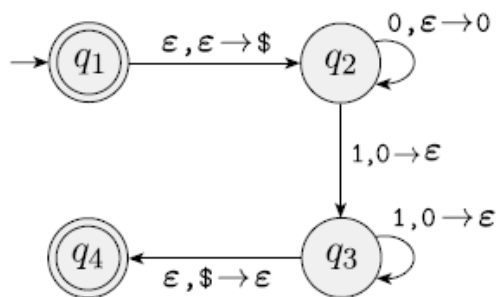$Q = q_1, q_2, q_3, q_4$
$\Sigma = \{0, 1\}$
$\Gamma = \{0, \$\}$
$F = \{q_1, q_4\}$
δ is given by the following table:

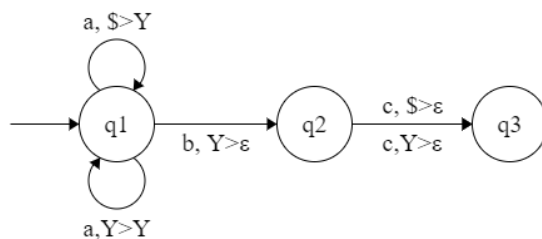| Input:	0			1			ε		
Stack:	0	$	ε	0	$	ε	0	$	ε
q_1									$\{(q_2,\$)\}$
q_2			$\{(q_2,0)\}$	$\{(q_3,\varepsilon)\}$					
q_3				$\{(q_3,\varepsilon)\}$				$\{(q_4,\varepsilon)\}$	
q_4									

4

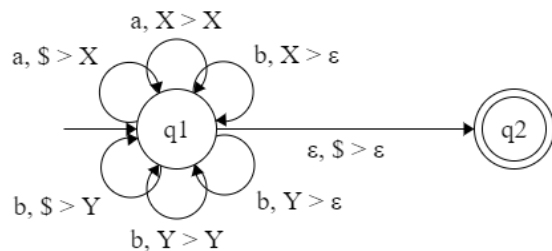The machine M can also be described by the following state diagram:

M is a pushdown automaton that recognizes language $\{0^n 1^n | \ n \geq 0\}$

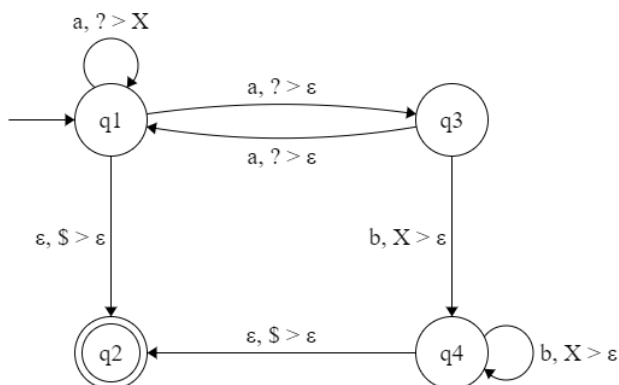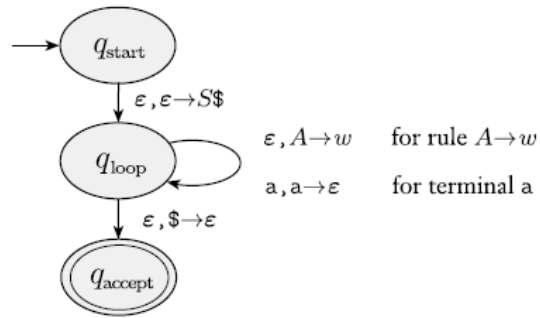**Note** The stack of a pushdown automata can be non-empty when the computation finishes.

Example aa^*bc

Example All strings over $\{a, b\}$ with the same number of a's and b's.

Example $S \rightarrow \epsilon | aSb | aaS$

Theorem 3.12 *If a language is context free, then some pushdown automaton recognizes it.*

q_{start}

$\varepsilon, \varepsilon \rightarrow S\$$

q_{loop}

$\varepsilon, \$ \rightarrow \varepsilon$

q_{accept}

$\varepsilon, A \rightarrow w$ for rule $A \rightarrow w$

$a, a \rightarrow \varepsilon$ for terminal a

Example Construct a pushdown automaton from the following grammar:

$$S \rightarrow aTb \mid b$$
$$T \rightarrow Ta \mid \epsilon$$

q_{start}

$\varepsilon, \varepsilon \rightarrow \$$

$\varepsilon, \varepsilon \rightarrow S$

$\varepsilon, S \rightarrow b$ $\varepsilon, \varepsilon \rightarrow T$ $\varepsilon, \varepsilon \rightarrow a$

$\varepsilon, T \rightarrow a$ $\varepsilon, \varepsilon \rightarrow T$

q_{loop}

$\varepsilon, \$ \rightarrow \varepsilon$

$\varepsilon, S \rightarrow b$
$\varepsilon, T \rightarrow \varepsilon$
$a, a \rightarrow \varepsilon$
$b, b \rightarrow \varepsilon$

q_{accept}