# COSE215: Theory of Computation

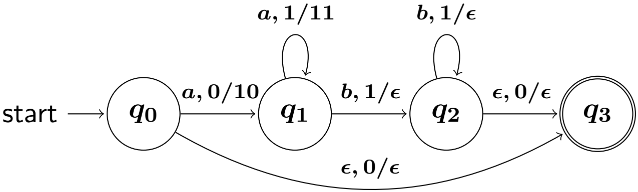
# Lecture 13 — Pushdown Automata (2)

Hakjoo Oh
2017 Spring

# Exercises

- $L = \{a^n b^n \mid n \geq 0\}$:

$$P = (\{q_0, q_1, q_2, q_3\}, \{a, b\}, \{0, 1\}, \delta, q_0, 0, \{q_3\})$$



- $L = \{w \in \{a,b\}^* \mid n_a(w) = n_b(w)\}$
  - Intuition: Whenever we read $a$, we insert a counter symbol $0$ onto the stack, and pop one counter symbol from the stack whenever $b$ is found. For example,

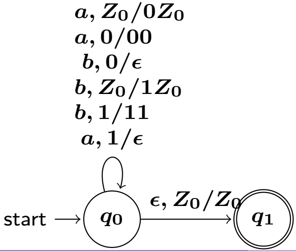
$$(abab, Z_0) \rightarrow (bab, 0Z_0) \rightarrow (ab, Z_0) \rightarrow (b, 0Z_0) \rightarrow (\epsilon, Z_0)$$

  - For the cases where a prefix of the input string contains more $b$'s than $a$'s, use a negative counter symbol, say 1, for counting the $b$'s that should be matched against $a$'s later. For example,

$$(bbaa, Z_0) \rightarrow (baa, 1Z_0) \rightarrow (aa, 11Z_0) \rightarrow (a, 1Z_0) \rightarrow (\epsilon, Z_0)$$

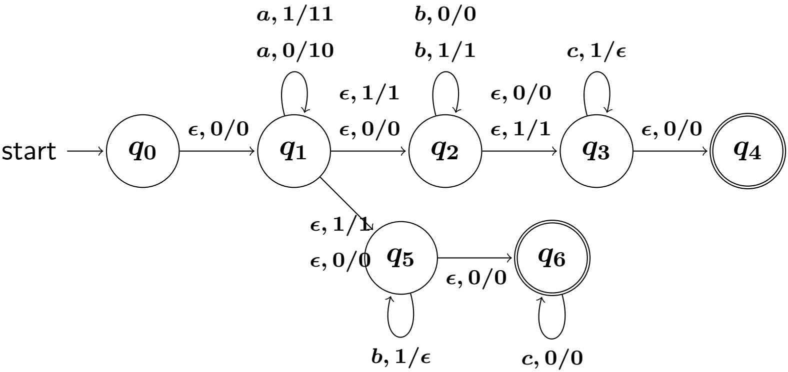
  - The pushdown automaton:

$$P = (\{q_0, q_1\}, \{a, b\}, \{0, 1, Z_0\}, \delta, q_0, Z_0, \{q_1\})$$



- $L = \{a^i b^j c^k \mid i,j,k \geq 0 \wedge (i = j \vee i = k)\}$
  - Think of the two cases separately:
    1. $L_1 = \{a^i b^j c^k \mid i,j,k \geq 0 \wedge i = j\}$.
    2. $L_2 = \{a^i b^j c^k \mid i,j,k \geq 0 \wedge j = k\}$.

$$P = (\{q_0,q_1,q_2,q_3,q_4,q_5,q_6\},\{a,b,c\},\{0,1\},\delta,q_0,0,\{q_4,q_6\})$$



# Configurations of PDA

- A configuration of a PDA consists of the automaton state and the stack contents.

- The configuration or instantaneous description (ID) is represented by $(q, w, \gamma)$, where
  - $q$ is the state,
  - $w$ is the remaining input, and
  - $\gamma$ is the stack contents.

- Suppose $(q, aw, X\beta)$ is a configuration and $(p, \alpha) \in \delta(q, a, X)$. Then, the configuration moves in one step to $(p, w, \alpha\beta)$:

$$(q, aw, X\beta) \vdash (p, w, \alpha\beta)$$

# The Language of Pushdown Automata

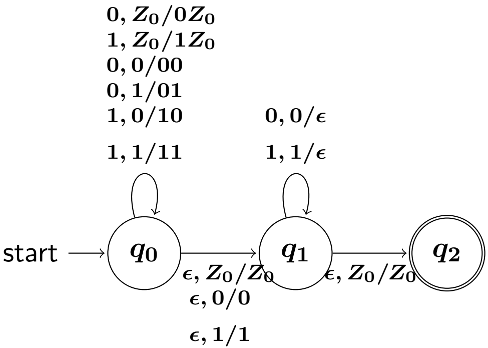
**Definition (Acceptance by Final State)**

Let $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a PDA. Then $L(P)$, the language of $P$ by final state, is

$$L(P) = \{w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q, \epsilon, \alpha)\}$$

for some state $q \in F$ and any stack string $\alpha$.

# Example




The PDA contains $1111$, because $(q_0, 1111, Z_0) \vdash^* (q_2, \epsilon, Z_0)$.

# Another Way of Defining The Language of a PDA

**Definition (Acceptance by Empty Stack)**

Let $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a PDA. Then $N(P)$, the language of $P$ accepted by empty stack, is

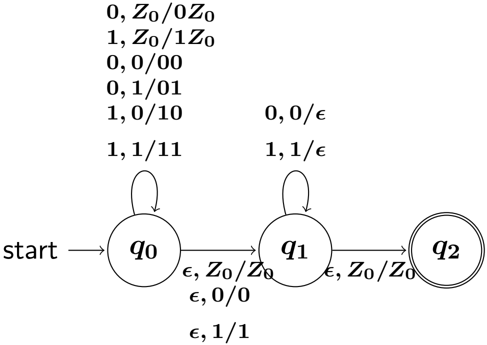
$$N(P) = \{w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q, \epsilon, \epsilon)\}$$

for any $q$.

When accepting by empty stack, we omit the $F$ component:

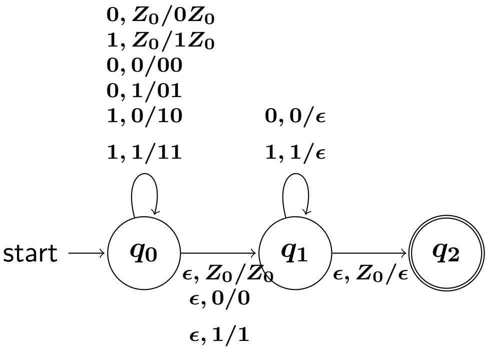
$$(Q, \Sigma, \Gamma, \delta, q_0, Z_0)$$

# Example




- $L(P) =$
- $N(P) =$

# Examples




- $L(P) =$
- $N(P) =$

# Equivalence

**Theorem (Equivalence of Final State and Empty Stack)**

*For any language $L$, there exists a PDA $P_F$ such that $L = L(P_F)$ iff there exists a PDA $P_N$ such that $L = N(P_N)$.*

**Lemma (From Empty Stack to Final State)**

*For any PDA $P_N = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0)$, there is a PDA $P_F$ such that $N(P_N) = L(P_F)$.*

**Lemma (From Final State to Empty Stack)**

*For any PDA $P_F = (Q, \Sigma, \Gamma, \delta_F, q_0, Z_0, F)$, there is a PDA $P_N$ such that $N(P_N) = L(P_F)$.*

# From Empty Stack to Final State

Given $P_N = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0)$, define

$$P_F = (Q \cup \{p_0, p_f\}, \Sigma, \Gamma \cup \{X_0\}, \delta_F, p_0, X_0, \{p_f\})$$

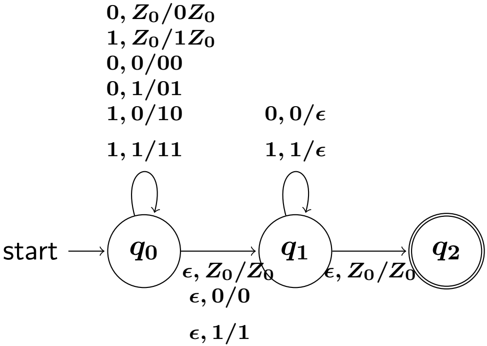
where

1. $\delta_F(p_0, \epsilon, X_0) = \{(q_0, Z_0X_0)\}$
2. For all $q \in Q$, $a \in \Sigma \cup \{\epsilon\}$, and $Y \in \Gamma$, $\delta_F(q, a, Y)$ contains $\delta_N(q, a, Y)$.
3. For all $q \in Q$, $\delta_F(q, \epsilon, X_0)$ contains $(p_f, \epsilon)$.

Then, $w$ is in $L(P_F)$ if and only if $w$ is in $N(P_N)$.

# Example

Convert the following PDA to a PDA that accepts that same language by empty stack:



# From Final State to Empty Stack

Given $P_F = (Q, \Sigma, \Gamma, \delta_F, q_0, Z_0, F)$, define

$$P_N = (Q \cup \{p_0, p\}, \Sigma, \Gamma \cup \{X_0\}, \delta_N, p_0, X_0)$$

where

1. $\delta_N(p_0, \epsilon, X_0) = \{(q_0, Z_0X_0)\}$
2. For all $q \in Q$, $a \in \Sigma \cup \{\epsilon\}$, and $Y \in \Gamma$, $\delta_N(q, a, Y)$ includes $\delta_F(q, a, Y)$.
3. For all accepting states $q \in F$ and $Y \in \Gamma \cup \{X_0\}$, $\delta_N(q, \epsilon, Y)$ includes $(p, \epsilon)$.
4. For all stack symbols $Y \in \Gamma \cup \{X_0\}$, $\delta_N(p, \epsilon, Y) = \{(p, \epsilon)\}$.

# Equivalence of PDA's and CFG's

The following three classes of languages:

1. The context-free languages, i.e., the languages defined by CFG's.
2. The languages that are accepted by final state by some PDA.
3. The languages that are accepted by empty stack by some PDA.

are all the same class.

# From CFG to PDA

Given a CFG $G = (V, T, P, S)$, define a PDA $P$ (by empty stack):

$$P = (\{q\}, T, V \cup T, \delta, q, S)$$

where

- For each variable $A \in V$,

$$\delta(q, \epsilon, A) = \{(q, \beta) \mid (A \to \beta) \text{ is in } G\}$$

- For each terminal $a \in T$,

$$\delta(q, a, a) = \{(q, \epsilon)\}$$

# Example

$$G = (\{B\}, \{(,)\}, P, B)$$

$$B \rightarrow BB \mid (B) \mid \epsilon$$

# Deterministic Pushdown Automata

**Definition**

A pushdown automata $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ is a *deterministic pushdown automata* (DPDA) if $P$ makes at most one move at a time, i.e.,

1. $|\delta(q, a, X)| \leq 1$ for any $q \in Q$, $a \in \Sigma \cup \{\epsilon\}$, and $X \in \Gamma$.
2. If $\delta(q, a, X) \neq \emptyset$ for some $a \in \Sigma$, then $\delta(q, \epsilon, X) = \emptyset$.

**Definition**

A language $L$ is said to be a deterministic context-free language iff there exists a DPDA $P$ such that $L = L(P)$.

# Example

The language

$$L = \{a^n b^n \mid n \geq 0\}$$

is a deterministic context-free language.

$\boxed{\text{Fact1: DCFLs includes some CFLs}}$

# Example

The language

$$\boldsymbol{L = \{ww^R \mid w \in \{a, b\}^*\}}$$

is *not* a deterministic context-free language.

$\boxed{\text{Fact2: DCFLs do not include some CFLs}}$

# Regular Languages and DCFLs

$\boxed{\text{Fact3: DCFLs include all RLs}}$

## Theorem

*If $L$ is a regular language, then $L = L(P)$ for some DPDA $P$.*

## Proof.

Let $A = (Q, \Sigma, \delta_A, q_0, F)$ be a DFA. Construct DPDA

$$P = (Q, \Sigma, \{Z_0\}, \delta_p, q_0, Z_0, F)$$

where define $\delta_p(q, a, Z_0) = \{(p, Z_0)\}$ for all $p$ and $q$ such that $\delta_A(q, a) = p$. Then, $(q_0, w, Z_0) \vdash^* (p, \epsilon, Z_0)$ iff $\delta_A^*(q_0, w) = p$. □

# DPDA's and Ambiguous Grammars

Fact4: All DCFLs have unambiguous grammars.

**Theorem**

*If $L = L(P)$ for some DPDA $P$, then $L$ has an unambiguous grammar.*

Fact5: DCFLs do not include all unambiguous CFLs.

The language

$$L = \{ww^R \mid w \in \{a, b\}^*\}$$

has an unambiguous grammar

$$S \to aSa \mid bSb \mid \epsilon$$

but not a DPDA language.

# Summary

- PDA $=$ FA with a stack
- PDA is more powerful than FA. Cover *all* CFLs.
  - Still limited, e.g., $\{ww \mid w \in \Sigma^*\}$.
- DPDA is between FA and PDA

In general,

- FA with an external storage
  - queue, two stacks, random access memory, ...?
  - increase the language-recognizing power?