CS 172: Computability and Complexity

Pushdown Automata & Equivalence with CFGs

Sanjit A. Seshia
EECS, UC Berkeley

Acknowledgments: L.von Ahn, L. Blum, M. Blum

Chomsky Normal Form

A CFG is in Chomsky Normal Form (CNF) if every rule is in one of the following three forms:

$S \rightarrow \varepsilon$

$A \rightarrow B\ C$ B, C are variables $\neq$ S

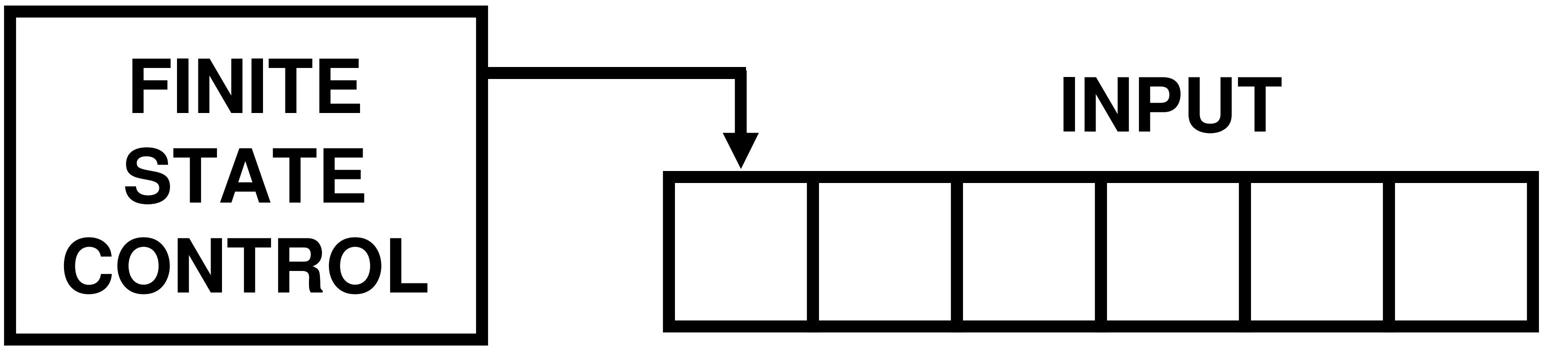
$A \rightarrow a$ a is a terminal

(S is the start variable; A is *any variable,* including S)

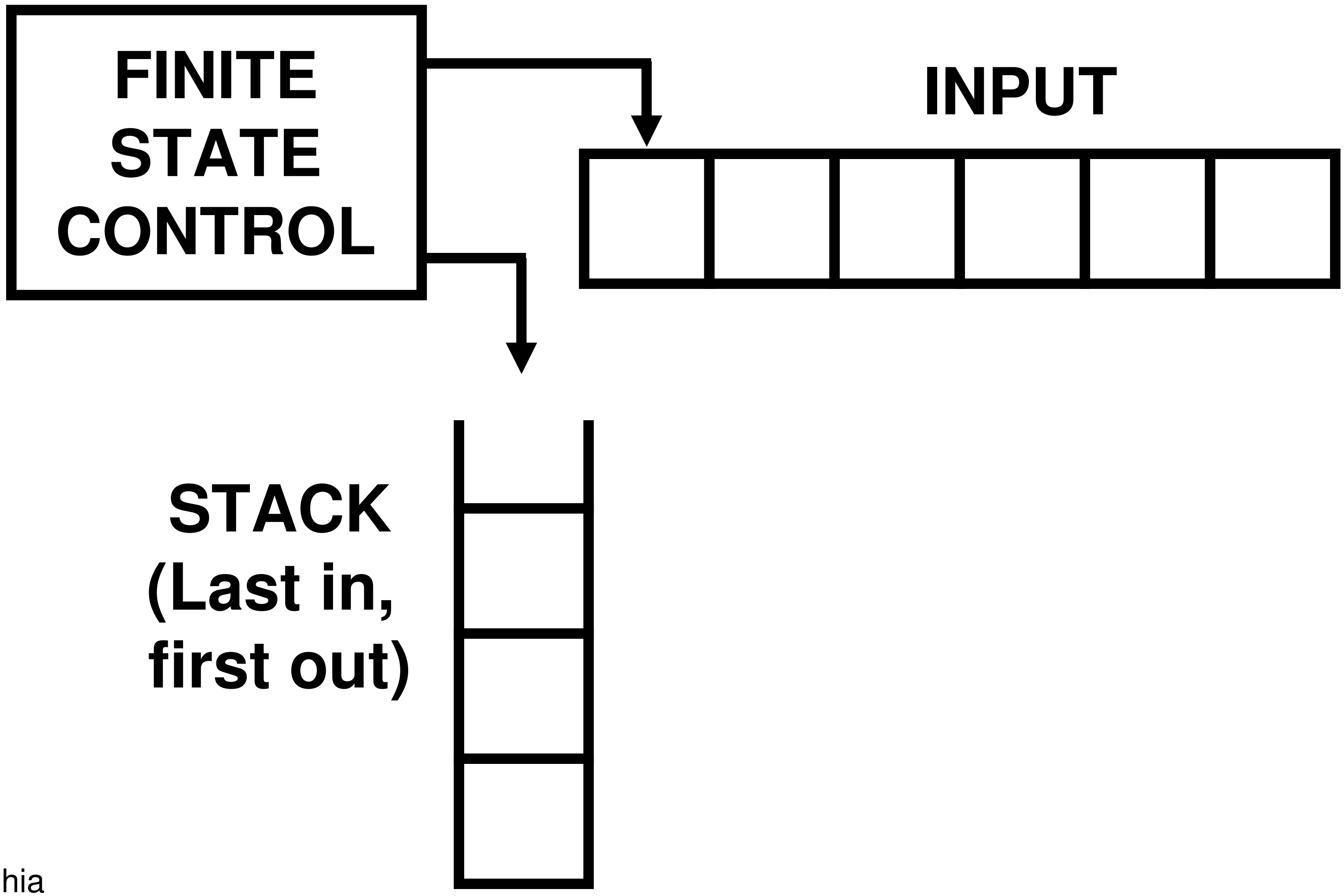
Theorem: Any CFG can be converted into an equivalent CFG (generating the same CFL) in Chomsky Normal Form

(proof done on the board – read Sipser Thm 2.9)

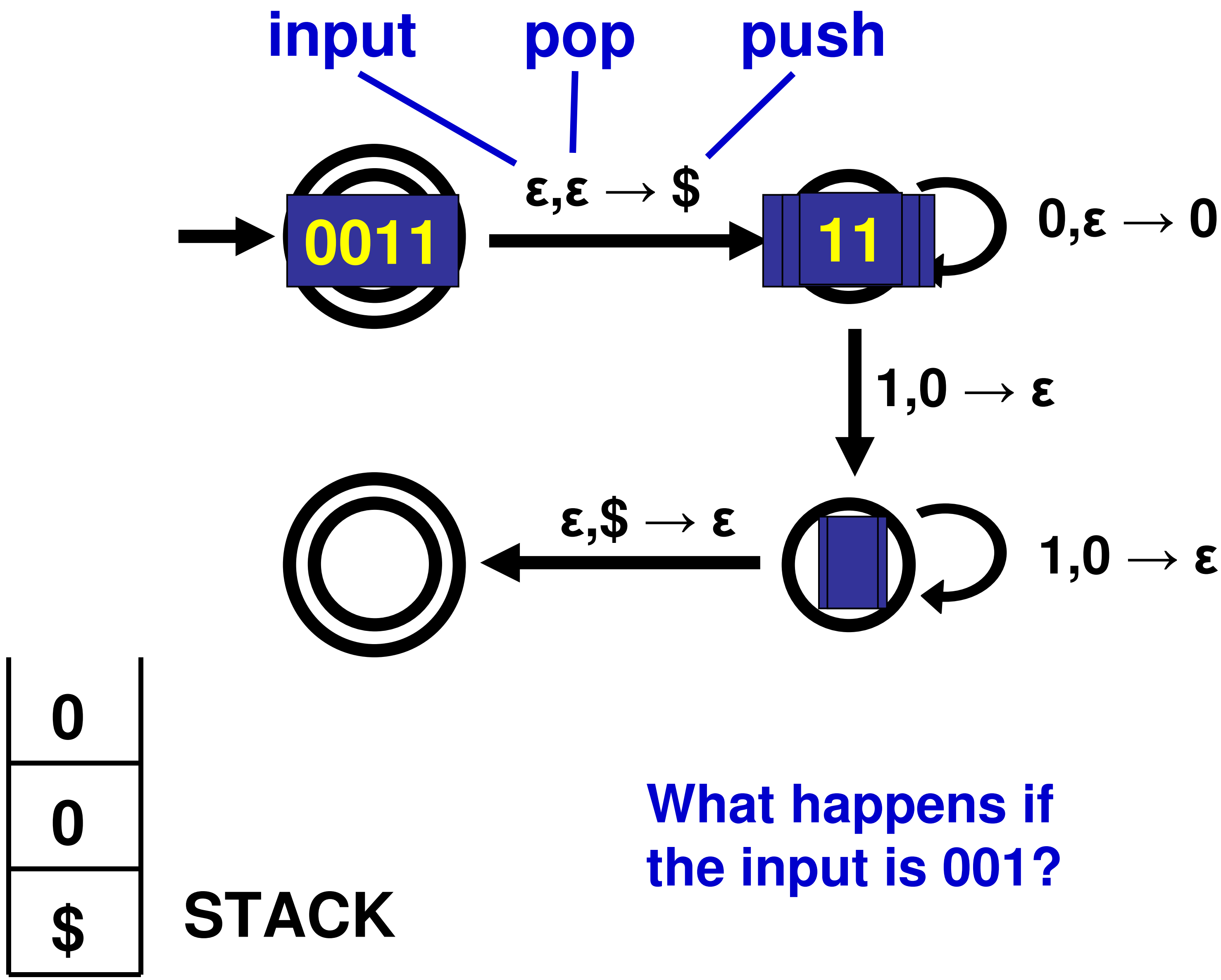
Finite Automaton



Pushdown Automaton



PUSHDOWN AUTOMATON EXAMPLE



What happens if the input is 001?

Informal Definition of **Acceptance**

- A pushdown automation accepts if, after reading the entire input, it ends in an accept state
  - Sometimes: (with an empty stack)

Definition: A (non-deterministic) PDA is a tuple $P = (Q, \Sigma, \Gamma, \delta, q_0, F)$, where:

Q is a finite set of states

Σ is the input alphabet

Γ is the stack alphabet

$\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \rightarrow 2^{Q \times \Gamma_\varepsilon}$ (non-determinism)

$q_0 \in Q$ is the start state

$F \subseteq Q$ is the set of accept states

2^S is the set of subsets of S

$\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}, \quad \Gamma_\varepsilon = \Gamma \cup \{\varepsilon\}$

Formal Definition of **Acceptance**

PDA P = $(Q, \Sigma, \Gamma, \delta, q_0, F)$ accepts a word $w \in \Sigma^*$

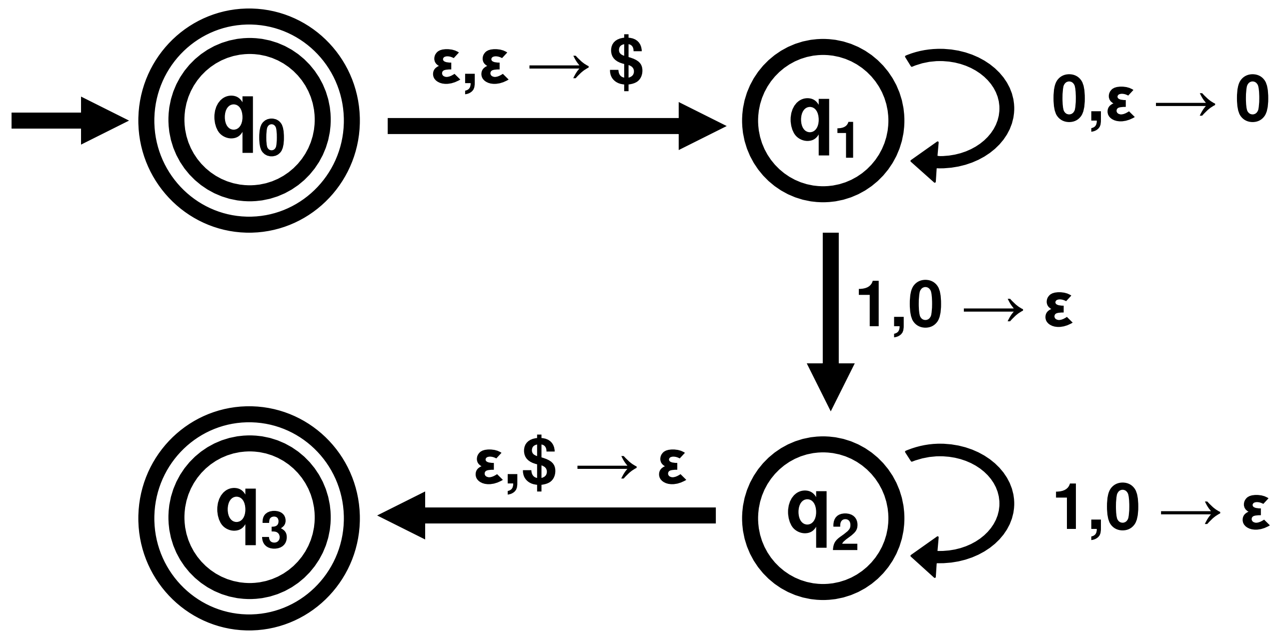
where $w = w_1 w_2 w_3 \ldots w_m$ with $w_i \in \Sigma_\varepsilon$

if *there exists* a sequence

$(q_0, s_0) \rightarrow (q_1, s_1) \rightarrow (q_2, s_2) \rightarrow \ldots \rightarrow (q_m, s_m)$

where

- $s_i \in \Gamma^*$ (represent the stack), with $s_0 = \varepsilon$,
- $q_m \in F$,
- $(q_{i+1}, b) \in \delta(q_i, w_{i+1}, a)$
  - where $s_i = at$ and $s_{i+1} = bt$, $a,b \in \Gamma_\varepsilon$, $t \in \Gamma^*$



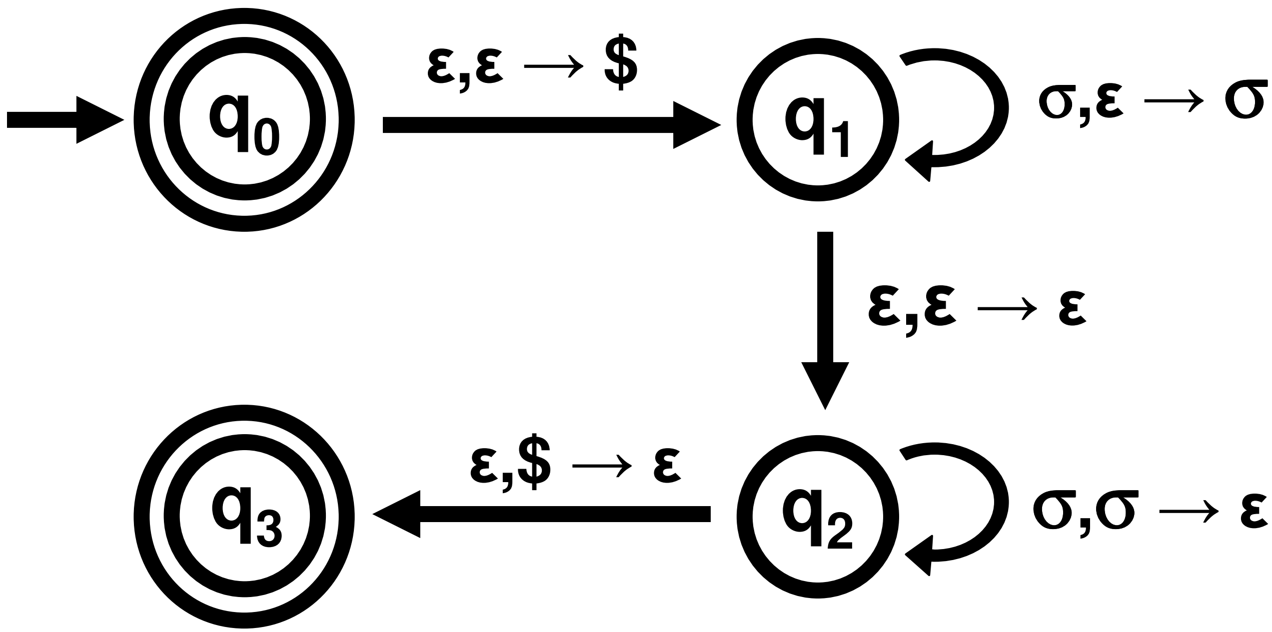
$Q = \{q_0, q_1, q_2, q_3\}$ $\Sigma = \{0,1\}$ $\Gamma = \{\$,0,1\}$

$$\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \rightarrow 2^{Q \times \Gamma_\varepsilon}$$

$\delta(q_1,1,0) = \{ (q_2,\varepsilon) \}$ $\delta(q_2,1,1) = \emptyset$

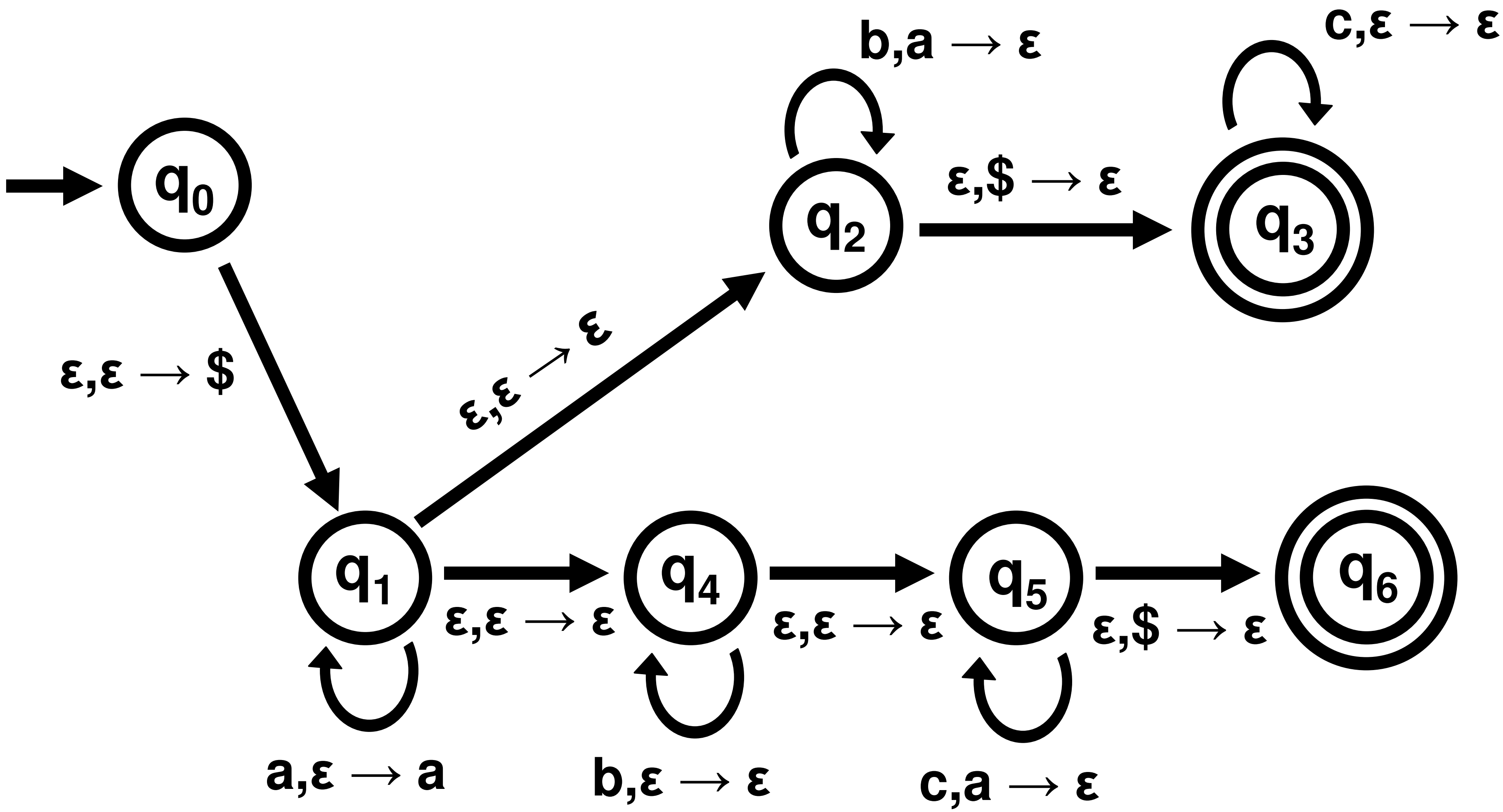
EVEN-LENGTH PALINDROMES

Σ = {a, b, c, …, z}, $\sigma \in \Sigma$



Build a PDA to recognize

$L = \{ a^i b^j c^k \mid i, j, k \geq 0 \text{ and } (i = j \text{ or } i = k) \}$



Theorem

A Language is generated by a CFG

$\Leftrightarrow$

It is recognized by a PDA

Theorem

A Language is generated by a CFG

$\Rightarrow$

It is recognized by a PDA

Suppose L is generated by a CFG G = (V, Σ, R, S)

Construct P = (Q, Σ, Γ, δ, q, F) that recognizes L

Intuition (warning: not a formal proof!)

Map a derivation in the CFG G to
an accepting sequence for the PDA P

Let $w \in L(G)$

There is a derivation in G:

$S \rightarrow \alpha_1 \rightarrow \alpha_2 \rightarrow \ldots \rightarrow \alpha_{m-1} \rightarrow w$ where $\alpha_i \in (\Sigma \cup V)^*$

We map it to an accepting sequence

$(q_0, s_0) \rightarrow (q_1, s_1) \rightarrow \ldots (q_2, s_2) \rightarrow \ldots \rightarrow (q_m, s_m) \rightarrow (q_f, s_f)$

where

$q_1 = q_2 = \ldots = q_m = q_{loop}$, $q_f \in F$,

$s_0 = S\$$, $s_i = \alpha_i\$$ $(1 \leq i \leq m)$

s_i is obtained from s_{i-1} $(1 \leq i \leq m)$ by using substitution at corresponding step of the derivation and matching terminals on the top of the stack with the input

(see picture on slide 16)

Suppose L is generated by a CFG G = (V, Σ, R, S)

Construct P = (Q, Σ, Γ, δ, q, F) that recognizes L

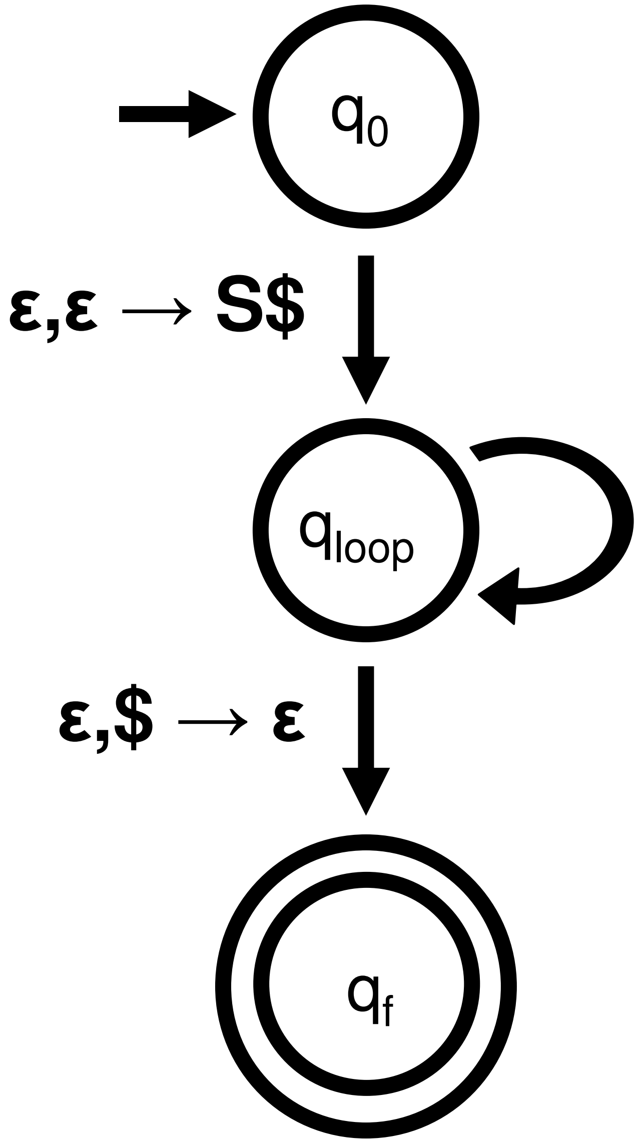
(1) Place the marker symbol $ and the start variable S on the stack

(2) Repeat the following steps forever:

(a) If top of stack is a variable, non-deterministically select rule that matches the variable and push result into the stack

(b) If top of stack is a terminal, read next symbol from input and compare it to terminal. If different, reject.

(c) If top of stack is $, then enter accept state. Accept if the input has all been read.



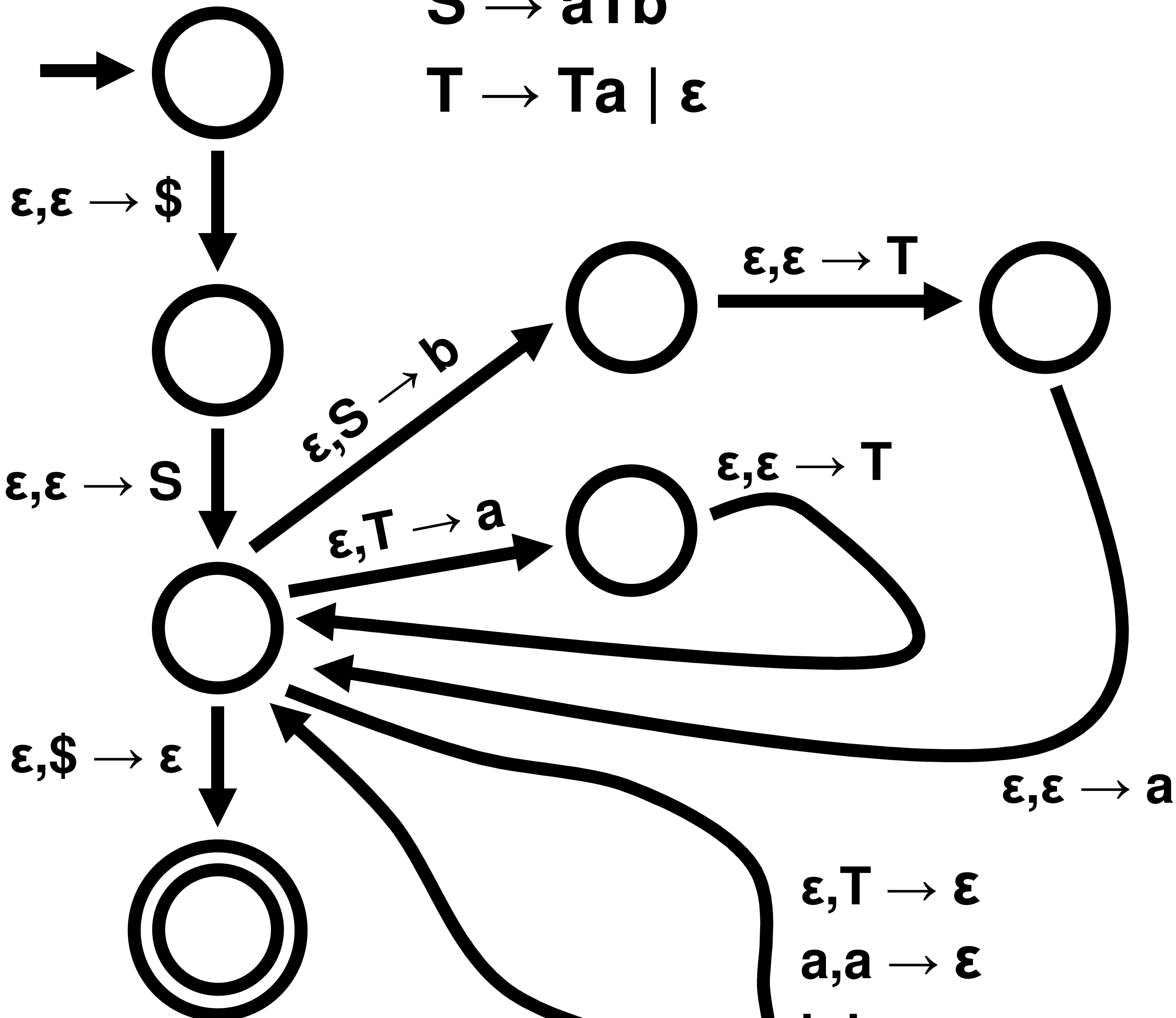
ε,A → w for rule A → w

a,a → ε for terminal a

Note: RHS is a string (non-std notation just for intuition)

$S \rightarrow aTb$

$T \rightarrow Ta \mid \varepsilon$



Next:

A Language is generated by a CFG

$\Rightarrow$

It is recognized by a PDA

Next Steps

- Read Sipser 2.3 in preparation for next lecture