Chapter 7 Pushdown Automata

Nondeterministic Pushdown Automata

PDA and CFL

DPDA and DCFL

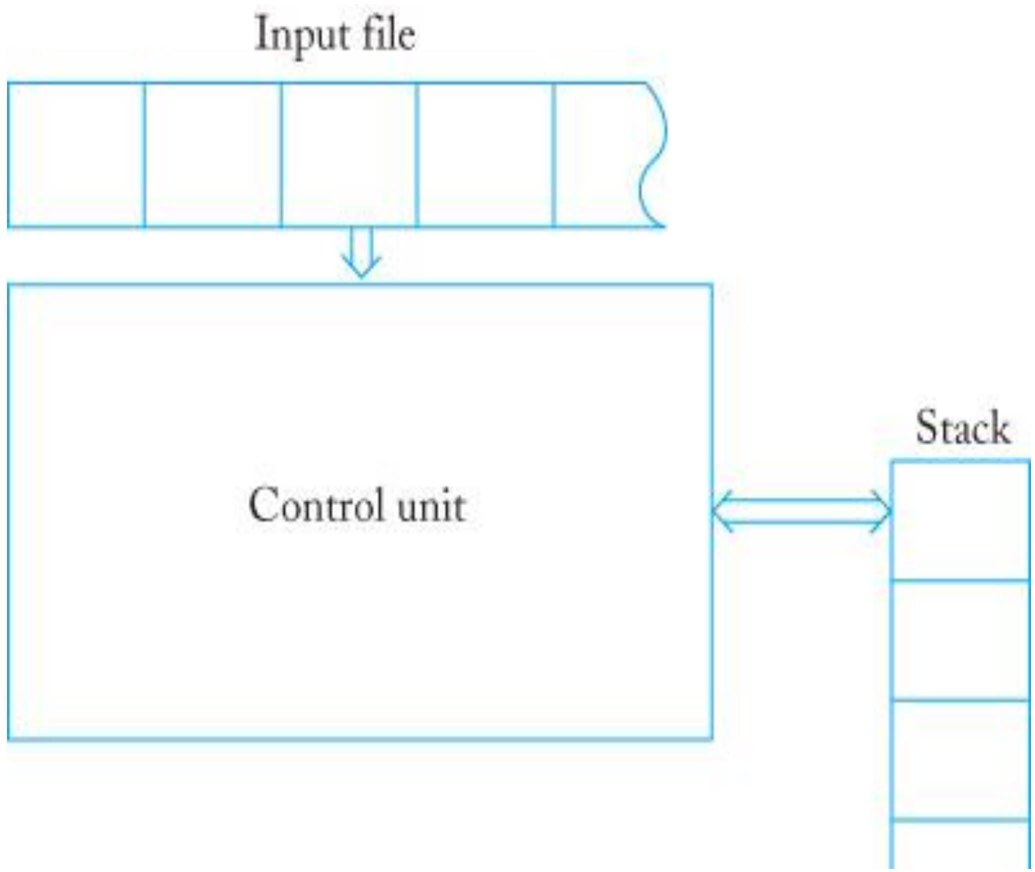
Pushdown Automata (PDA)

- Automata: FA, PDA, TM…
- Grammar: RG, CFG, CSG …
- Language: RL, CFL, CSL …
  - PDA is the model of nondeterministic top-down parser
  - For CFL
    - CFG – generator
    - PDA - recognizer

Difference between FA and PDA

- An informal definition of PDA
  - An input string
  - A read head to exam input once cell at a time
  - A finite state machine to control moves
  - LIFO push down stack
- DPDA is not equivalent to NPDA

Memory makes PDA a more powerful recognizer/accepter



Definition 7.1
A nondeterministic pushdown accepter (npda) is defined by the septuple

- $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$
- Q, Σ, δ, q_0, and F are the same as they are in FA
- Γ is a finite set of symbols called the stack alphabet
- $z \in \Gamma$ is the stack start symbol
- δ: $Q \times (\Sigma \cup \{\lambda\}) \times \Gamma \rightarrow$ finite subset of $Q \times \Gamma^*$

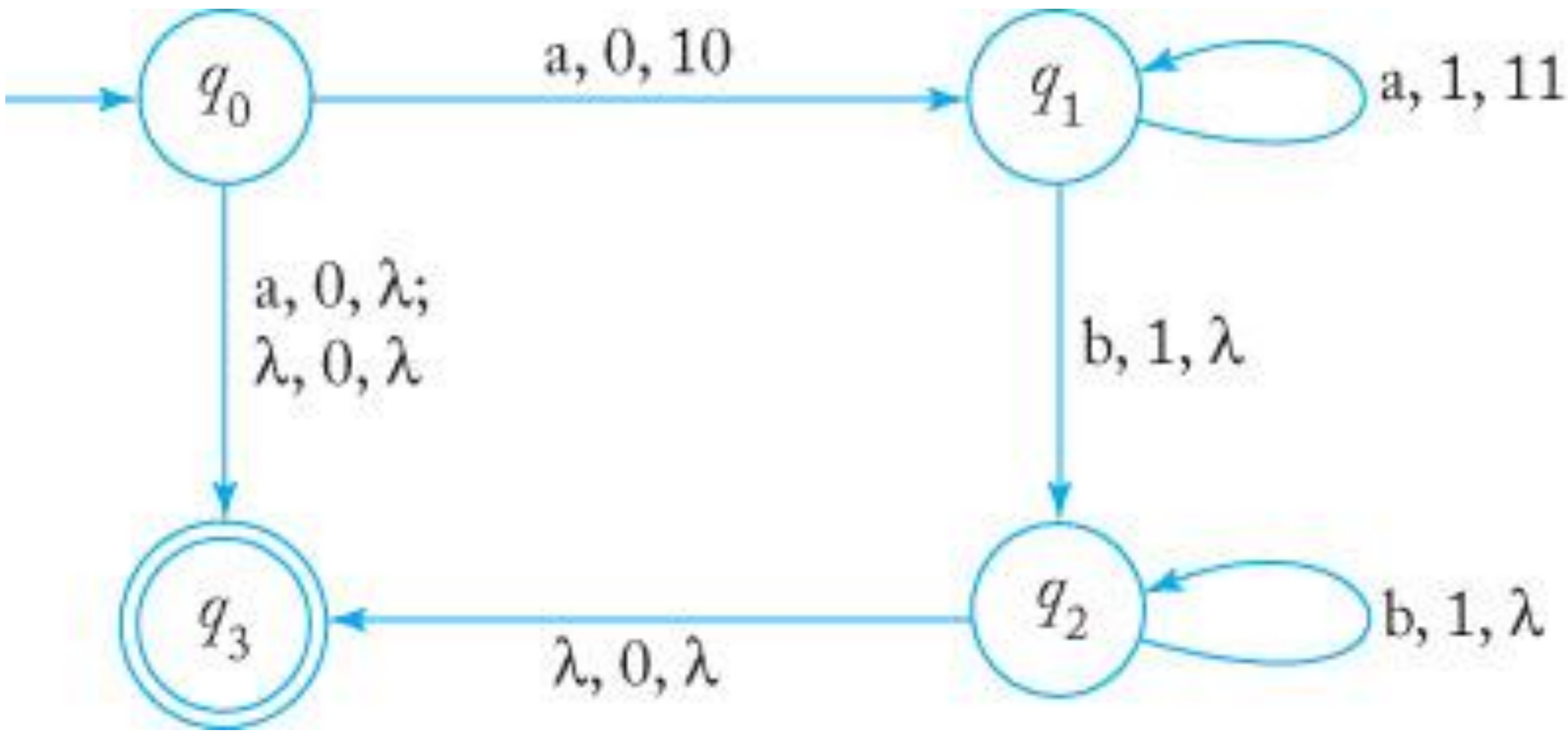
Example 7.1 sample NPDA transition rule

- $\delta\ (q_1, a, b) = \{(q_2, cd), (q_3, \lambda)\}$
- What does this transition rule mean?
  - Two things can happen when in q_1, read a, and top of the stack is b
    1. Move to q_2, and the string cd replaces b
    2. Move to q_3, and pop b

NPDA Examples

- FA $\rightarrow$ NPDA
  - Every FA can be transformed to a PDA
- $\{a^n b^n, n \geq 0\}$
  - Can you design an algorithm for this PDA?
- **Example 7.2**. $L = \{a^n b^n, n \geq 0\} \cup \{a\}$
  - PDA design and implementation
    - Design an algorithm first
    - Present your design in transition function, TG, or in picture notation (in Cohen book)

Example 7.3
TG for npda in example 7.2



The Language Accepted by a PDA

- $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$

- $L(M) = \{w \in \Sigma^*: (q_0, w, z) \rightarrow^*_M (p, \lambda, u), p \in F, u \in \Gamma^*\}$

- Language accepted by M is the set of all strings that can put M into a final state at the end of the string.

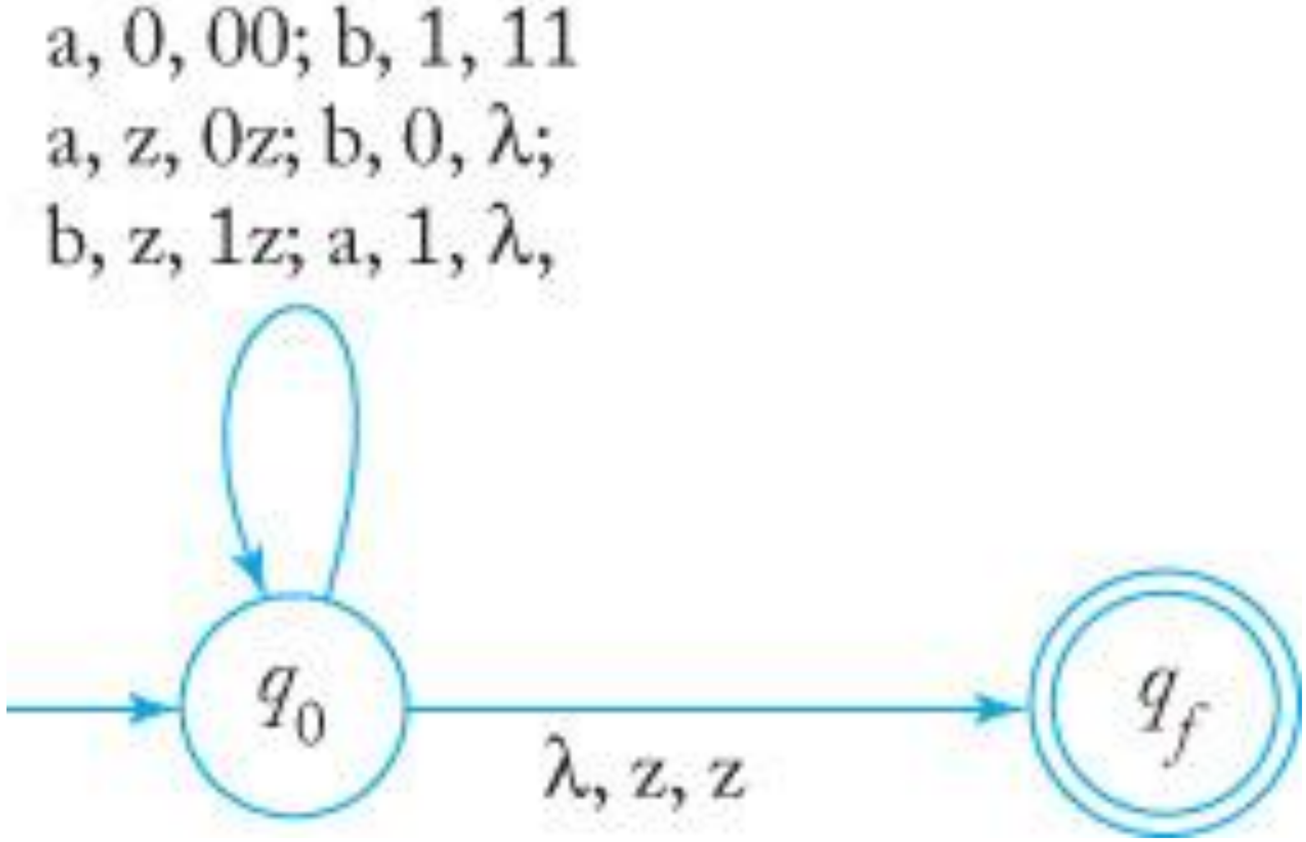
More examples of NPDA -1

- DPDA for $L = \{wXw^R\} \quad w \in (a+b)^*$
  - Algorithm design in English
- NPDA for ODDPALINDROME
  - Algorithm design…
- NPDA for EVENPALINDROME
  - Example 7.5 Linz
  - Focus on its algorithm design first!
  - Try to draw the TG

More examples of NPDA -2

- Example 7.4
  - Construct a npda for the language
  - $L = \{w \in \{a, b\}^*: n_a(w) = n_b(w)\}$
    - Algorithm design for this npda
      - Counter 0 is pushed into stack for each a read
      - Negative counter 1 for counting b's that are to be matched against a's later
    - TG in next slide

TG for $n_a(w) = n_b(w)$



Exercises from Linz 7.1

- Construct npda's that accept the following languages on $\Sigma = \{a, b, c\}$
- Exercise #4
  - (c) $L = \{a^n b^m c^{n+m}: n \geq 0, m \geq 0\}$
  - Try it in class
    - Algorithm first
    - Implementation next

PDA and CFL

- Theorem 7.1
  - For every CFL, there is an npda M such that $L = L(M)$
  - Proof (outline)
    - CFL $\rightarrow$ CFG in Greibach normal form, G
    - $G = (V, T, S, P)$
    - $M = (\{q_0, q_1, q_f\}, T, V \cup \{z\}, \delta, q_0, z, \{q_f\})$
    - See detail steps in the constructive proof on page 186-187

Example 7.6 - 1

Construct a pda that accepts the language generated by grammar $S \to aSbb \mid a$

- Transform to Greibach normal form
  - $S \to aSA \mid a$
  - $A \to bB$
  - $B \to b$
- The start symbol S is put on the stack by
  - $\delta(q_0, \lambda, z) = \{(q_1, Sz)\}$
- $S \to aSA \mid a$ are represented in the PDA by
  - $\delta(q_1, a, S) = \{(q_1, SA), (q_0, \lambda)\}$

Example 7.6 - 2

- In an analogous manner,
  - $A \rightarrow bB$ gives
    - $\delta(q_1, b, A) = \{(q_1, B)\}$
  - $B \rightarrow b$ gives
    - $\delta(q_1, b, B) = \{(q_1, \lambda)\}$
  - Appearance of z signals the completion and pda is put into final state
    - $\delta(q_1, \lambda, z) = \{(q_2, \lambda)\}$

CFG for PDA

- Theorem 7.2
  - If L = L(M) for some npda M, the L is a CFL
  - Proof.
    - A constructive proof to show that it is always possible to construct a grammar from a npda.
  - This theorem tells us that there always a CFG for a language represented by a npda.
  - In practice, one may design a CFG directly from the give CFL

DPDA and DCFL

- A PDA M is said to be deterministic if it is an automaton as defined in Definition 7.1 for NPDA, subject to
  - For any given input symbol and stack top, at most one move can be made
  - When a λ-move is possible for some configuration, no input-consuming alternative is available
- DCFL L – if and only if there exists a dpda M such that L = L(M)