CSE211 – Formal Languages and Automata Theory

U2L7_Equivalence of PDA and CFL

Dr. P. Saravanan

School of Computing

SASTRA Deemed University

Outline

- Introduction
- Definition of PDA
- Instantaneous Descriptions
- The Language of a PDA
- Equivalence of PDA’s and CFG’s
  - PDA to CFG
  - CFG to PDA

Definition of PDA

- Formal Definition

  A PDA is a 7-tuple $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ where

  – Q: a finite set of states

  - Σ: a finite set of input symbols
  - Γ: a finite stack alphabet
  - δ: a transition function such that $\delta(q, a, X)$ **is a set of pairs** (p, γ) where
    - $q \in Q$ (the current state)
    - $a \in \Sigma$ or $a = \varepsilon$ (an input symbol or an empty string)
    - $X \in \Gamma$
    - $p \in Q$ (the next state)

Definition of PDA

SASTRA DEEMED TO BE UNIVERSITY

- Formal Definition
  - $\gamma \in \Gamma^*$ which replaces X on the top of the stack:
    when $\gamma = \varepsilon$, the top stack symbol is popped up
    when $\gamma = X$, the stack is unchanged
    when $\gamma = YZ$, X is replaced by Z, and Y is pushed to the top
    when $\gamma = \alpha Z$, X is replaced by Z and string α is pushed to the top
  - q_0: the start state
  - Z_0: the start symbol of the stack
  - F: the set of accepting or final states

Instantaneous Descriptions of PDA

- The *configuration* of a PDA is represented by a 3-tuple (q, w, γ) where
  - q is the state;
  - w is the remaining input; and
  - γ *is the* stack content.
- Such a 3-tuple is called an instantaneous description (ID) of the PDA.

Instantaneous Descriptions of a PDA

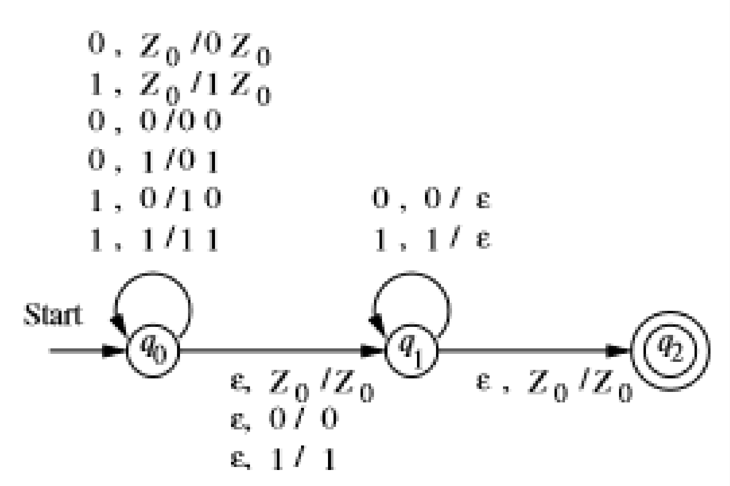
- Instantaneous Descriptions of a PDA
  - The change of an ID into another is called a *move*, denoted by the symbol $\vdash_P$, or $\vdash$ when *P* is understood.
  - So, if $\delta(q, a, X)$ contains (p, α), then the following is a corresponding move:

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

  - We use $\overset{*}{\vdash}_P$ or $\overset{*}{\vdash}$ to indicate zero or more moves.

Instantaneous Descriptions of a PDA

– Moves for the PDA to accept input $w = 1111$:



$(q_0, 1111, Z_0) \vdash (q_0, 111, 1Z_0)$

$\vdash (q_0, 11, 11Z_0) \vdash (q_1, 11, 11Z_0)$

$\vdash (q_1, 1, 1Z_0)$

$\vdash (q_1, \varepsilon, Z_0) \vdash (q_2, \varepsilon, Z_0)$

Instantaneous Descriptions of a PDA

- Instantaneous Descriptions of a PDA
  - **Theorem 6.5**

If $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ is a PDA, and

$$(q, x, \alpha) \overset{*}{\underset{P}{\vdash}} (p, y, \beta),$$

then for any string w in Σ^* and γ in Γ^*, it is also true that

$$(q, xw, \alpha\gamma) \overset{*}{\underset{P}{\vdash}} (p, yw, \beta\gamma).$$

The Language of a PDA

- Some important facts:
  - Two ways to define languages of PDA's:
    - by final state and
    - by empty stack, as mentioned before.
  - It can be proved that a language *L* has a PDA that accepts it by final state *if and only if* *L* has a PDA that accepts it by empty stack.
  - For a given PDA *P*, the language that *P* accepts by final state and by empty stack are usually *different*.

The Language of a PDA

- Acceptance by Final State
  - **Definition:**

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

$$\{w \mid (q_0, w, Z_0) \overset{*}{\underset{P}{\vdash}} (q, \varepsilon, \alpha), q \in F\}$$

for any α.

The Language of a PDA

- Acceptance by Empty Stack
  - **Definition:**

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

    $$\{w \mid (q_0, w, Z_0) \overset{*}{\underset{P}{\vdash}} (q, \varepsilon, \varepsilon)\}$$

    for any q.

The Language of a PDA

- Acceptance by Empty Stack
  - **Example 6.8**

    The PDA of Example 6.2 may be modified to accept L_{ww^R} by empty stack:

    simply change the original transition

$$\delta(q_1, \varepsilon, Z_0) = \{(q_2, Z_0)\}$$

to be

$$\delta(q_1, \varepsilon, Z_0) = \{(q_2, \varepsilon)\}.$$

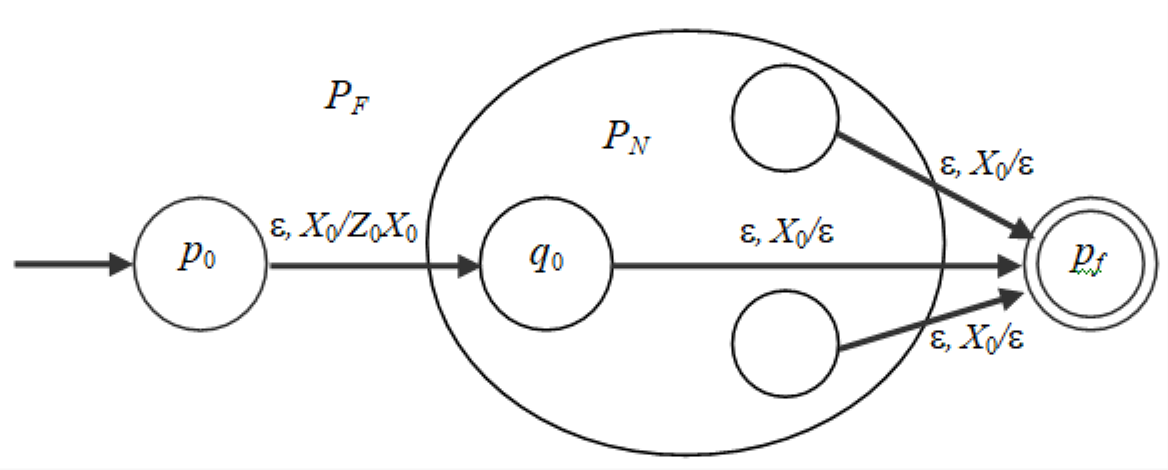
(just eliminate Z_0)

The Language of a PDA

- From Empty Stack to Final State
  - **Theorem 6.9**

If $L = N(P_N)$ for some PDA $P_N = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0)$, then there is a PDA P_F such that $L = L(P_F)$.

Proof. Just add new final state from all end

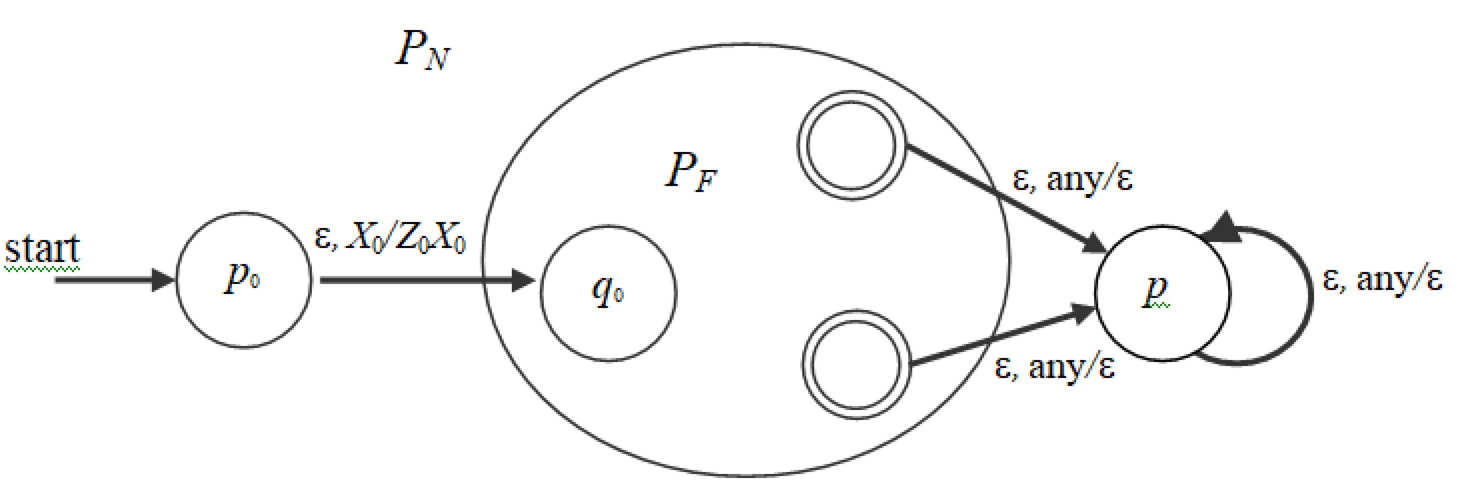


The Language of a PDA

- From Final State to Empty Stack
  - **Theorem 6.11**

    Let L be $L(P_F)$ for some PDA $P_F = (Q, \Sigma, \Gamma, \delta_F, q_0, Z_0, F)$. Then there is a PDA P_N such that $L = N(P_N)$.

    Proof. The idea is to use Fig. 6.7 below (in final states of P_F, pop up the remaining symbols in the stack).



Equivalence of PDA's and CFG's

- Equivalences to be proved:
  1) CFL's defined by CFG's
  2) Languages accepted by final state by some PDA
  3) Languages accepted by empty stack by some PDA

– Equivalence of 2) and 3) above have been proved.

Equivalence of PDA's and CFG's

- From Grammars to PDA's
  - Given a CFG $G = (V, T, Q, S)$, construct a PDA P that accepts $L(G)$ by empty stack in the following way:
  - $P = (\{q\}, T, V \cup T, \delta, q, S)$ where the transition function δ is defined by:
    - for each nonterminal A,
      - $\delta(q, \varepsilon, A) = \{(q, \beta) \mid A \to \beta$ is a production of $G\}$;
    - for each terminal a,
      - $\delta(q, a, a) = \{(q, \varepsilon)\}$.

Equivalence of PDA's and CFG's

- From Grammars to PDA's
  - **Theorem 6.13**

    If PDA P is constructed from CFG G by the construction above, then $N(P) = L(G)$.

Equivalence of PDA's and CFG's

- From Grammars to PDA's
  - **Example 6.12** - Construct a PDA from the expression grammar

$$I \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1;$$
$$E \to I \mid E*E \mid E+E \mid (E).$$

The transition function for the PDA is as follows:

a) $\delta(q, \varepsilon, I) = \{(q, a), (q, b), (q, Ia), (q, Ib), (q, I0), (q, I1)\}$

b) $\delta(q, \varepsilon, E) = \{(q, \mathrm{I}), (q, E+E), (q, E*E), (q, (E))\}$

c) $\delta(q, d, d) = \{(q, \varepsilon)\}$ where d may any of the terminals $a, b, 0, 1, (,), +, *$.

References

- John E. Hopcroft, Rajeev Motwani and Jeffrey D. Ullman, *Introduction to Automata Theory,* Languages, and Computation, Pearson, 3rd Edition, 2011.
- Peter Linz, An Introduction to Formal Languages and Automata, Jones and Bartle Learning International, United Kingdom, 6th Edition, 2016.

Next Class:

Problems in Equivalence of PDA and CFL

THANK YOU.