CSE211 – Formal Languages and Automata Theory

U2L9_Deterministic PDA

Dr. P. Saravanan

School of Computing

SASTRA Deemed University

Outline

- Deterministic PDA’s
- Definition and Example
- RL and DPDA
- Properties of CGL

Deterministic PDA's

- Definition of a Deterministic PDA
  - Intuitively, a PDA is deterministic if there is never a choice of moves (including *ε–moves*) in any situation.
  - Formally, a PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ is said to be deterministic (a DPDA) if and only if the following two conditions are met:
    - $\delta(q, a, X)$ has at most one element for any $q \in Q$, $a \in \Sigma$ or $a = \varepsilon$, and $X \in \Gamma$.
    - If $\delta(q, a, X)$ is nonempty for some $a \in \Sigma$, then $\delta(q, \varepsilon, X)$ must be empty.

Deterministic PDA's

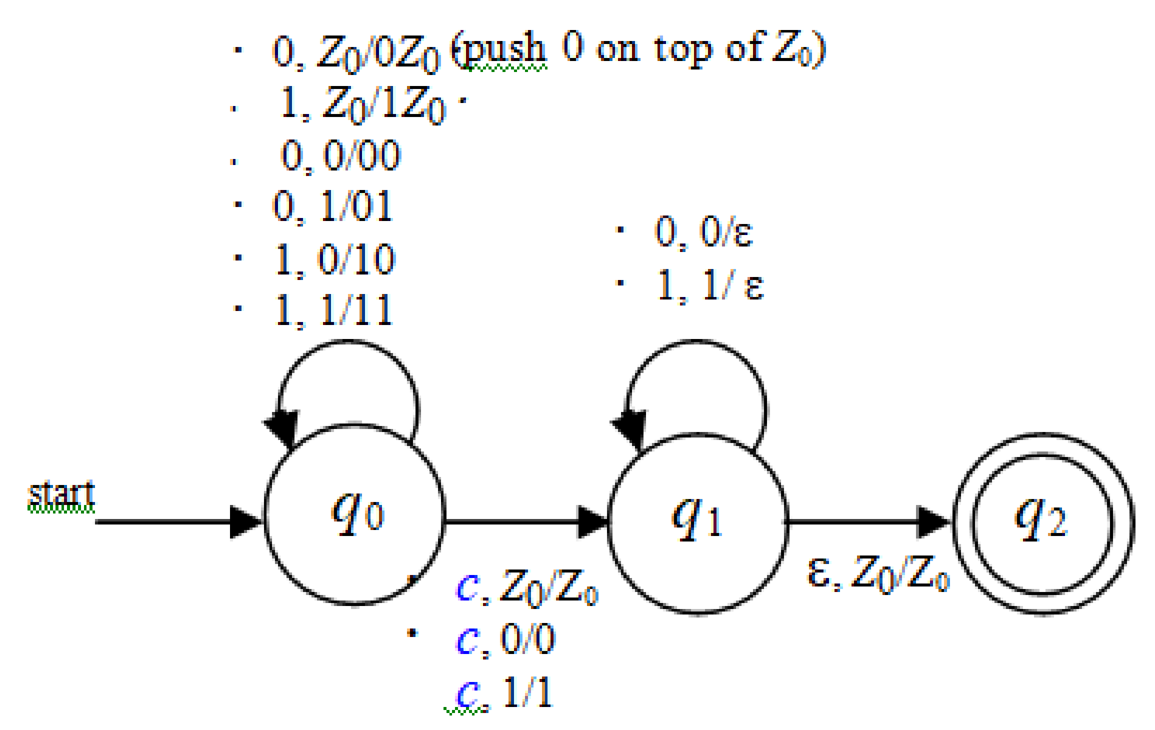
- Definition of a DPDA
  - **Example**
    - There is no DPDA for L_{ww^R} of Example 6.2.
    - But there is a DPDA for a modified version of L_{ww^R} as follows,

$$L_{wcw^R} = \{wcw^R \mid w \in L((\mathbf{0} + \mathbf{1})^*)\}.$$

    - To recognize wcw^R, just store 0's & 1's in stack until center marker c is seen. Then, match the remaining input w^R with the stack content (w).
    - The PDA can so be designed to be deterministic by searching the center marker *without trying matching all the time nondeterministically* .

Deterministic PDA's

- DPDA Example
  - **Example 6.16** (cont'd) A desired DPDA is as follows. (The difference is just the blue *c*.)



Deterministic PDA's

- Regular Languages and DPDA's
  - The DPDA's accepts a class of languages that is *between* the RL's and the CFL's, as proved in the following.
  - **Theorem 6.17**

    If L is an RL, then $L = L(P)$ for some DPDA P (accepting by final state).

    Proof. Easy. Just use a DPDA to simulate a DFA as follows.

    If DFA $A = (Q, \Sigma, \delta_A, q_0, F)$ accepts L, then construct DPDA $P = (Q, \Sigma, \{Z_0\}, \delta_P, q_0, Z_0, F)$ where δ_P is such that $\delta_P(q, a, Z_0) = \{(p, Z_0)\}$ for all states p and q in Q such that $\delta_A(q, a) = p$.

Deterministic PDA's

- Regular Languages and DPDA's
  - The language-recognizing capability of the *DPDA by empty stack* is *rather limited*.
  - **Theorem 6.19**

    A language L is $N(P)$ for some DPDA P if and only if L has the prefix property and L is $L(P')$ for some DPDA P' (for proof, do exercise 6.4.3).
  - A language L is said to have the prefix property if there are no two different strings x and y in L such that x is a prefix of y.

Deterministic PDA's

- DPDA's and CFL's
  - DPDA's can be used to accept non-RL's, for example, L_{wcw^R} mentioned before.
    - It can be proved by the pumping lemma that L_{wcw^R} *is not an RL* (see the textbook, pp. 254~255).
  - On the other hand, DPDA's *by final state* cannot accept certain CFL's, for example, L_{ww^R}.
    - It can be proved that L_{ww^R} cannot be accepted by a DPDA by final state (see an informal proof in the textbook, p. 255).

Deterministic PDA's

- DPDA's and Ambiguous Grammars
  - **Theorem 6.20**

    If $L = N(P)$ (*accepting by empty stack*) for some DPDA P, then L has an unambiguous CFG.
  - **Theorem 6.21**

    If $L = L(P)$ for some DPDA P (*accepting by final state*), then L has an unambiguous CFG.

PROPERTIES OF CONTEXT-FREE LANGUAGES

Properties of CFL

- CFG's may be simplified to fit certain special forms, like
  - *Chomsky Normal Form (CNF)* and
  - *Greiback Normal Form (GNF)*.
- Some, but not all, properties of RL's are also possessed by the CFL's.
- Unlike the RL, many computational problems about the CFL *cannot* be answered.
- That is, there are many undecidable problems about CFL's.

A Substitution Rule

$$S \to aB$$
$$A \to aaA$$
$$A \to abBc$$
$$B \to aA$$
$$B \to b$$

Substitute
$$B \to b$$

$$S \to aB \mid ab$$
$$A \to aaA$$
$$A \to abBc \mid abbc$$
$$B \to aA$$

A Substitution Rule

$$S \to aB \mid ab$$

$$A \to aaA$$

$$A \to abBc \mid abbc$$

$$B \to aA$$

Substitute

$$B \to aA$$

$$S \to \cancel{aB} \mid ab \mid aaA$$

$$A \to aaA$$

$$A \to \cancel{abBc} \mid abbc \mid abaAc$$

Equivalent grammar

In general

$$A \to xBz$$

$$B \to y_1$$

Substitute

$$B \to y_1$$

$$A \to xBz \mid xy_1z$$

Simplification of CFG

- *Every CFG can be transformed into an equivalent grammar in Chomsky Normal Form,* after simplifying the CFG in the following ways:
-
  - eliminating *useless symbols* (which do not appear in any derivation from the start symbol);
  - eliminating ε-*productions* (of the form $A \rightarrow \varepsilon$);
  - eliminating *unit productions* (of the form $A \rightarrow B$);

Summary

- Deterministic PDA’s
- Definition and Example
- RL and DPDA
- Properties of CGL
- Normal forms
- Eliminating production rules

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:

Simplification of CFG

THANK YOU