

# CSCI 256: Theory of Computing

## CHAPTER 6 *Pushdown Automata*

Dr. Benjoe A. Juliano

Tel. 530 898-4619 (office)  
530 898-6442 (dept)  
Fax. 530 898-5995

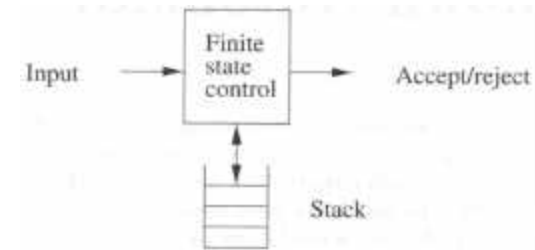
Juliano@ecst.csuchico.edu  
http://www.ecst.csuchico.edu/~juliano



## Definition

### Informal Introduction

- The *pushdown automaton* (PDA) is essentially an  $\epsilon$ -NFA with a stack.



## Definition

- A PDA,  $P = (Q, S, G, \delta, q_0, Z_0, F)$ , is a 7-tuple consisting of
  - a finite set of *states*, denoted  $Q$
  - a finite set of *input symbols*, denoted  $S$
  - a finite *stack alphabet*, denoted  $G$
  - a *transition function*,  $\delta: Q \times \Sigma \cup \{\epsilon\} \times \Gamma \rightarrow Q \times \Gamma^*$
  - the *start state*  $q_0 \in Q$
  - the *start symbol*  $Z_0 \in \Gamma$ ,
  - a set of *final states*  $F \subseteq Q$



## Definition

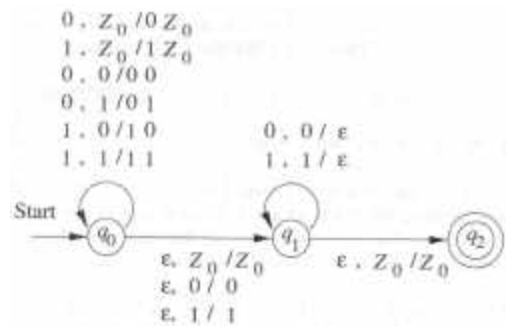
- For the *transition function*,  $\delta: Q \times \Sigma \cup \{\epsilon\} \times \Gamma \rightarrow Q \times \Gamma^*$ 
  - Informally, for  $q \in Q$ ,  $a \in \Sigma \cup \{\epsilon\}$ , and  $X \in \Gamma$ ,  $\delta(q, a, X) = (p, g)$  where
    - $p \in Q$  is the *next state*; and
    - $g \in \Gamma^*$  is the string of stack symbols that replaces  $X$  at the top of the stack.
      - If  $g = \epsilon$ , then the stack is *popped*.
      - If  $g = X$ , then the stack is *unchanged*.
      - If  $g = YZ$ , then  $X$  is replaced by  $Z$ , and  $Y$  is *pushed* onto the stack.



## Definition

### Example

- The CFL  $L = \{ww^R \mid w \in \{0,1\}^*\}$  is represented by the following PDA:



B. Juliano © 2002, based on notes by J. Ullman

5

## Instantaneous Descriptions

- The **configuration** of a PDA  $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$  is a triple  $(q, w, \mathbf{g})$ , where
  - $q \in Q$  is the (current) state;
  - $w \in \Sigma^*$  is the remaining output; and
  - $\mathbf{g} \in \Gamma^*$  is the stack contents. (By convention, the top of the stack is at the left end of  $\mathbf{g}$  and the bottom at the right end.)
- Such a triple is called an **instantaneous description**, or ID, of the PDA,  $P$ .

B. Juliano © 2002, based on notes by J. Ullman

6

## Moves of a PDA

- The “turnstile” notation for connecting pairs of IDs that represent one or many **moves** of a PDA  $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$  is denoted by  $\vdash_P$ , or just  $\vdash$  when  $P$  is understood.
- Suppose  $\delta(q, a, X) = (p, \alpha)$ . Then, for all strings  $w \in \Sigma^*$  and  $\beta \in \Gamma^*$ :
 
$$(q, aw, X\beta) \vdash (p, w, \alpha\beta)$$
- So, by consuming  $a$  (which may be  $\epsilon$ ) from the input and replacing  $X$  on the top of the stack with  $\alpha$ , we can go from state  $q$  to state  $p$ .

B. Juliano © 2002, based on notes by J. Ullman

7

## Moves of a PDA

- The symbol  $\vdash_P^*$ , or just  $\vdash^*$  when  $P$  is understood, is used to denote zero or more moves of the PDA  $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ .
- That is,
  - $I \vdash^* I$ , for any ID  $I$ .
  - $I \vdash^* J$ , if there exists some ID  $K$  such that  $I \vdash K$  and  $K \vdash^* J$ . So,  $I \vdash^* J$  if there is a sequence of  $n$  IDs  $K_1, K_2, \dots, K_n$  such that  $I = K_1, J = K_n$ , and for all  $1 \leq i \leq n-1$ , we have  $K_i \vdash K_{i+1}$ .

B. Juliano © 2002, based on notes by J. Ullman

8

## PDA IDs and Moves

- Three important principles about IDs and moves:
  - If a sequence of IDs (*computation*) is legal for a PDA  $P$ , then the computation formed by adding the same additional input string to the end of the input (second component) in each ID is also legal.
  - If a computation is legal for a PDA  $P$ , then the computation formed by adding the same additional stack symbols below the stack in each ID is also legal.

## PDA IDs and Moves

- Three important principles about IDs and moves:
  - If a computation is legal for a PDA  $P$ , and some tail of the input is not consumed, then the computation formed by removing this tail from the input in each ID is also legal.
- Theorem 6.5**
  - If  $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$  is a PDA, and  $(q, x, \alpha) \vdash_P^* (p, y, \beta)$ , then for any strings  $w \in \Sigma^*$  and  $\hat{\gamma} \in \Gamma^*$ , it is also true that

$$(q, xw, \alpha\hat{\gamma}) \vdash_P^* (p, yw, \beta\hat{\gamma})$$

## PDA IDs and Moves

- Theorem 6.6**
  - If  $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$  is a PDA, and
 
$$(q, xw, \alpha) \vdash_P^* (p, yw, \beta)$$
 then it is also true that  $(q, x, \alpha) \vdash_P^* (p, y, \beta)$ .

## PDA Languages

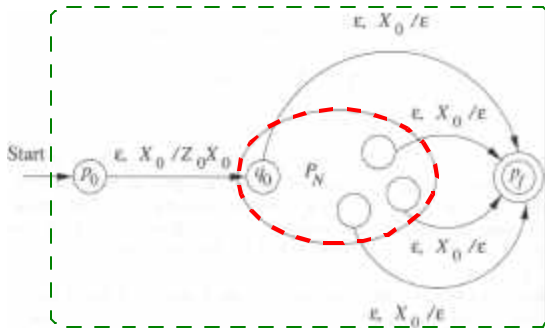
- Given a PDA  $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ .
  - $P$  **accepts** an input string  $w \in \Sigma^*$  if  $(q_0, w, Z_0) \vdash_P^* (p, \epsilon, \gamma)$  for any final state  $p \in F$  and any stack string  $\gamma \in \Gamma^*$ . This approach is known as “**acceptance by final state**” and the set of strings accepted this way is denoted  $L(P)$ .
  - $P$  **accepts** an input string  $w \in \Sigma^*$  if  $(q_0, w, Z_0) \vdash_P^* (q, \epsilon, \epsilon)$  for any state  $q \in Q$ . This approach is also known as “**acceptance by empty stack**” and the set of strings accepted this way is denoted  $N(P)$ .

## PDA Languages

From Figure 6.4 of IATLC, Hopcroft, Motwani, & Ullman, 2001.

### Theorem 6.9 (From *Empty Stack* to *Final State*)

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



B. Juliano © 2002, based on notes by J. Ullman

13

## PDA Languages

### Theorem 6.9 (From *Empty Stack* to *Final State*)

- (Constructive) Proof
  - Given  $L=N(P_N)$  for some PDA  $P_N=(Q,\Sigma,\Gamma,\delta_N,q_0,Z_0,F)$ .
  - Construct a PDA  $P_F$  as follows:
    - Introduce new start state  $p_0$  and new bottom-of-stack marker  $X_0$ .
    - First move of  $P_F$ :  $\delta_F(p_0,\epsilon,X_0) = (q_0,Z_0X_0)$
    - Then,  $P_F$  simulates  $P_N$ ; i.e. give  $P_F$  all the transitions of  $P_N$ .

B. Juliano © 2002, based on notes by J. Ullman

14

## PDA Languages

### Theorem 6.9 (From *Empty Stack* to *Final State*)

- (Constructive) Proof, *continued*:
  - Construction of PDA  $P_F$  ...
    - Introduce a new final state  $p_f$  for  $P_F$ .
    - For every state  $q \in Q$ ,  $\delta_F(q,\epsilon,X_0) = (p_f,\epsilon)$ .
  - Does this really work? For  $w \in Q^*$  where  $w \in N(P_N)$ :
    - $(p_0,w,X_0) \vdash_{P_F}^* (q_0,w,Z_0X_0) \vdash_{P_F}^* (q,\epsilon,X_0) \vdash_{P_F} (p_f,\epsilon,\epsilon)$
    - So,  $P_F$  accepts  $w$  by final state.

B. Juliano © 2002, based on notes by J. Ullman

15

## PDA Languages

### Theorem 6.9 (From *Empty Stack* to *Final State*)

- Example
  - Consider the following PDA,  $P_N = (Q=\{q\}, \Sigma=\{i,e\}, \Gamma=\{Z\}, \delta_N, q, Z, F=\emptyset)$ , which processes sequences of if's (denoted  $i$ ) and else's (denoted  $e$ ) in a C program:
    - Start
    - Transitions:  $e, Z/\epsilon$  and  $i, Z/Z$
  - Notice that accepts/recognizes if/else errors by the *empty stack*.

B. Juliano © 2002, based on notes by J. Ullman

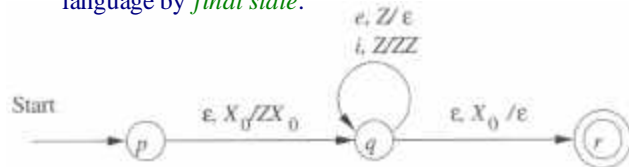
16

From Figure 6.6 of (ATLC, Hopcroft, Motwani, &amp; Ullman, 2001).

## PDA Languages

### Theorem 6.9 (From *Empty Stack* to *Final State*)

- Example, *continued*
  - Construct from  $P_N$  a PDA  $P_F$  that accepts the same language by *final state*.



- Hence,  $P_F = (\{q\} \cup N\{p_0=p, p_f=r\}, \Sigma=\{i, e\}, \{Z\} \cup N\{X_0\}, \delta_F, p_0=p, X_0, \{p_f=r\})$ .

B. Juliano © 2002, based on notes by J. Ullman

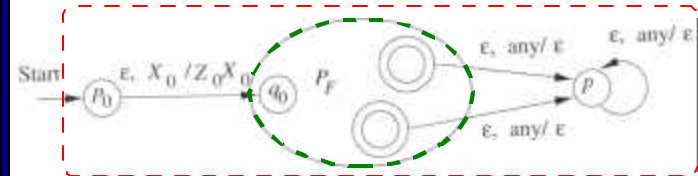
17

From Figure 6.7 of (ATLC, Hopcroft, Motwani, &amp; Ullman, 2001).

## PDA Languages

### Theorem 6.11 (From *Final State* to *Empty Stack*)

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



B. Juliano © 2002, based on notes by J. Ullman

18

## PDA Languages

### Theorem 6.11 (From *Final State* to *Empty Stack*)

- (Constructive) Proof:
  - Given  $L=L(P_F)$  for some PDA  $P_F=(Q, \Sigma, \Gamma, \delta_F, q_0, Z_0, F)$ .
  - Construct a PDA  $P_N$  as follows:
    - Introduce new start state  $p_0$ , new state  $p$ , and a new bottom-of-stack marker  $X_0$ .
    - First move of  $P_N$ :  $\delta_N(p_0, \epsilon, X_0) = (q_0, Z_0 X_0)$ .
    - Then,  $P_N$  simulates  $P_F$ ; i.e. give  $P_N$  all the transitions of  $P_F$ .

B. Juliano © 2002, based on notes by J. Ullman

19

## PDA Languages

### Theorem 6.11 (From *Final State* to *Empty Stack*)

- (Constructive) Proof, *continued*:
  - Construction of PDA  $P_N$  ...
    - For every state  $q \in F$ , and any stack symbol  $Y \in \Gamma$ ,  $\delta_F(q, \epsilon, Y) = (p, \epsilon)$ .
    - For any stack symbol  $Y \in \Gamma$ , add the transition  $\delta_F(p, \epsilon, Y) = (p, \epsilon)$  to use state  $p$  as an auxiliary to keep popping the stack of  $P_N$  until it is empty.
  - For  $w \in Q^*$  where  $w \in L(P_F)$  and  $q \in F$ :
    - $(p_0, w, X_0) \vdash_{P_N}^* (q_0, w, Z_0 X_0) \vdash_{P_N}^* (q, \epsilon, X_0) \vdash_{P_N}^* (p, \epsilon, \epsilon)$
    - So,  $P_N$  accepts  $w$  by empty stack.

B. Juliano © 2002, based on notes by J. Ullman

20

From Figure 6.8 of *ITLC*, Hopcroft, Motwani, & Ullman, 2001.

## Equivalence of PDAs & CFGs

- The following three classes of languages:
    - CFLs; *i.e.* languages defined by CFGs
    - Languages accepted by final state by some PDA
    - Languages accepted by empty stack by some PDA
- are all the same class!

We've shown these through Theorems 6.9 and 6.11



We need to show these next ...

B. Juliano © 2002, based on notes by J. Ullman

21

## Equivalence of PDAs & CFGs

### From CFGs to PDAs

- Let  $L = L(G)$  for some CFG  $G = (V, T, P, S)$ .
- Idea: Have PDA  $A$  simulate leftmost derivations in  $G$ , where a left-sentential form is represented by
  - the sequence of input symbols that  $A$  has consumed from its input, followed by
  - $A$ 's stack top
- Example: If  $(q,abcd,S) \vdash^* (q,cd,ABC)$ , then the left-sentential form represented is  $abABC$ .

B. Juliano © 2002, based on notes by J. Ullman

22

## Equivalence of PDAs & CFGs

### From CFGs to PDAs, *continued*:

- Moves of  $A$ 
  - If a terminal  $a \in T$  is on top of the stack, then there better be an  $a$  waiting on the input. PDA  $A$  consumes symbol  $a$  from the input and pops it from the stack, if so.
  - If a variable  $B \in V$  is on top of the stack, then PDA  $A$  has a choice of replacing  $B$  on the stack by the body of any production with head  $B$ .

B. Juliano © 2002, based on notes by J. Ullman

23

## Equivalence of PDAs & CFGs

### From CFGs to PDAs, *continued*:

- Formally, PDA  $A = (\{q\}, \Sigma, V \cup \Sigma, \delta, q, S, \emptyset)$  where  $\delta$  is defined by
  - if  $B \in V$ , then
 
$$\delta(q, \varepsilon, B) = \{(q, \alpha) \mid B \rightarrow \alpha \text{ is in } P\};$$
 or
  - if  $a \in \Sigma$ , then  $\delta(q, a, a) = \{(q, \varepsilon)\}$ .
- Example:
  - Given CFG  $G = (\{S, A\}, \{0, 1\}, P, S)$  where  $P$  consists of
 
$$S \rightarrow AS \mid \varepsilon$$

$$A \rightarrow 0A1 \mid A1 \mid 01$$

B. Juliano © 2002, based on notes by J. Ullman

24

## Equivalence of PDAs & CFGs

- From CFGs to PDAs, *continued*:
  - Example:
    - Then, PDA  $M = (\{q\}, \{0,1\}, \{0,1,A,S\}, \delta, q, S, \emptyset)$ , where  $\delta$  is defined by
      - "  $\delta(q,\varepsilon,S) = \{(q,AS),(q,\varepsilon)\}$
      - "  $\delta(q,\varepsilon,A) = \{(q,0A1),(q,A1),(q,01)\}$
      - "  $\delta(q,0,0) = \{(q,\varepsilon)\}$
      - "  $\delta(q,1,1) = \{(q,\varepsilon)\}$

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- Theorem 6.13**
  - If PDA  $P$  is constructed from CFG  $G$  by the construction above, then  $N(P) = L(G)$ .
- Proof
  - By induction on the number of steps in the derivation  $S \xrightarrow{lm}^* \alpha$  that for any  $x$ ,  $(q,wx,S) \vdash^* (q,x,\beta)$ , where
    - "  $w\beta = \alpha$
    - "  $\beta$  is the suffix of  $\alpha$  that begins at the leftmost variable ( $\beta = \varepsilon$  if there is no variable)
  - Proof detail in textbook ...*

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- From PDAs to CFGs
  - Let  $L = N(P)$  for some PDA  $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ .
  - Idea: Units of PDA action have the net effect of popping one symbol from the stack, consuming some input, and making a state change.
  - For  $q, p \in Q$  and  $X \in \Gamma$ , the **composite symbol**  $[qXp]$  is a single CFG variable that generates exactly those strings  $w$  such that  $P$  can read  $w$  from the input, pop  $X$  (net effect), and go from state  $q$  to state  $p$ .

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- Theorem 6.14**
  - Let  $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$  be a PDA. Then, there is a CFG  $G$  such that  $L(G) = N(P)$ .
- (Constructive) Proof
  - Construct CFG  $G = (V, \Sigma, P, S)$  where
    - "  $V$  consists of
      - A the start symbol,  $S$ , and
      - A all symbols of the form  $[qXp]$ , where  $q, p \in Q$  and  $X \in \Gamma$

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- (Constructive) Proof *continued*:
    - $P$  consists of
      - $S \rightarrow [q_0 Z_0 p]$ , for all states  $p \in Q$ .
      - Let  $(r, Y_1 Y_2 w Y_k) \in \delta(q, a, X)$ , where
        - $a \in \Sigma \cup \{\epsilon\}$
        - $k$  can be any number, including 0, in which case  $(r, Y_1 Y_2 w Y_k) = (r, \epsilon)$
- Then for all  $r_1, r_2, w, r_k \in Q$ ,  $G$  has the production
- $$[q X r_k] \rightarrow a [r Y_1 r_1] [r_1 Y_2 r_2] w [r_{k-1} Y_k r_k]$$

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- (Constructive) Proof *continued*:
  - Hence, productions in  $P$  could be of the form
    - **popping rule**,  $[q Z p] \rightarrow a$ , whenever  $(p, \epsilon) \in \delta(q, a, Z)$  ;
    - **one stack symbol, one state replacement rule**,  $[q Z r] \rightarrow a [p Y r]$ , for all  $r \in Q$ , whenever  $(p, Y) \in \delta(q, a, Z)$  ; or
    - **one stack symbol replaced by two rule**,  $[q Z s] \rightarrow a [p X r] [r Y s]$ , for all  $r, s \in Q$ , whenever  $(p, XY) \in \delta(q, a, Z)$ .

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- From PDAs to CFGs, *continued*:
  - So, the essence of the Theorem is
    - $[q X p] \xRightarrow{*} w$  if and only if  $(q, w, X) \vdash^* (p, \epsilon, \epsilon)$
  - Show the above holds by induction
    - (If) on number of moves made by PDA
    - (Only-if) on number of steps in the derivation
  - *Proof detail in textbook ...*

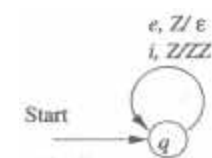
B. Juliano © 2002, based on notes by J. Ullman



From Figure 6.5 of ATLC, Hopcroft, Motwani, &amp; Ullman, 2001.

## Equivalence of PDAs & CFGs

- From PDAs to CFGs, *continued*:
  - Example: Recall the PDA,  $P_N = (Q = \{q\}, \Sigma = \{i, e\}, \Gamma = \{Z\}, \delta_N, q, Z, F = \emptyset)$ , which processes sequences of if's (denoted  $i$ ) and else's (denoted  $e$ ) in a C program:

and accepts/recognizes if/else errors by the *empty stack*.

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- **From PDAs to CFGs, *continued*:**
  - Construct CFG  $G = (V, \Sigma, P, S)$  where
    - $V$  consists of
      - the start symbol,  $S$ , and
      - $[qZq]$ , the only composite symbol
    - $P$  consists of
      - $S \rightarrow [qZq]$
      - $[qZq] \rightarrow i[qZq][qZq]$ , since  $(q,ZZ) \in \delta_N(q,i,Z)$
      - $[qZq] \rightarrow \epsilon$ , since  $(q,\epsilon) \in \delta_N(q,e,Z)$

B. Juliano © 2002, based on notes by J. Ullman

## Equivalence of PDAs & CFGs

- **From PDAs to CFGs, *continued*:**
  - Hence, replacing the composite symbol  $[qZq]$  by  $A$ , the CFG  $G$  has the productions
 
$$S \rightarrow A$$

$$A \rightarrow iAA \mid \epsilon$$
  - Furthermore, CFG  $G$  can be simply written as
 
$$G = (\{S\}, \{i,e\}, \{S \rightarrow iSS \mid \epsilon\}, S)$$

B. Juliano © 2002, based on notes by J. Ullman

## Deterministic PDAs

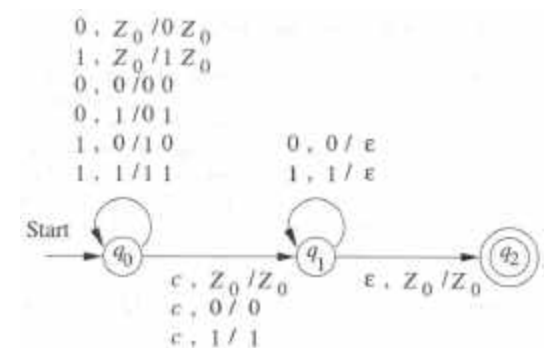
- A PDA,  $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ , is a **deterministic PDA** or **DPDA** if and only if the following conditions are met:
  - $|\delta(q,a,X)| \leq 1$  for any state  $q \in Q$ , input symbol  $a \in \Sigma \setminus \{\epsilon\}$ , and stack symbol  $X \in \Gamma$ .
  - If  $\delta(q,a,X) \neq \emptyset$  for some  $a \in \Sigma$ , then  $\delta(q,\epsilon,X) = \emptyset$ .
- Note: Parsers **are** DPDAs ...

B. Juliano © 2002, based on notes by J. Ullman

From Figure 6.11 of ATLC, Hopcroft, Motwani, &amp; Ullman, 2001.

## Deterministic PDAs

- **Example**
  - DPDA that accepts  $\{wcw^R \mid w \in (0+1)^*\}$



B. Juliano © 2002, based on notes by J. Ullman

## Deterministic PDAs

- **Regular Languages and DPDAs**
  - The DPDAs accept a class of languages that is between the RLs and the CFLs.
- **Theorem 6.17**
  - If  $L$  is RL, then  $L=L(P)$  for some DPDA  $P$ .
- **Proof:**
  - Let  $A = (Q, \Sigma, \delta_A, q_0, F)$  be a DFA.
  - Construct DPDA  $P = (Q, \Sigma, \{Z_0\}, \delta_P, q_0, Z_0, F)$  by defining  $\delta_P(q, a, Z_0) = \{(p, Z_0)\}$  for all states  $p, q \in Q$ , such that  $\delta_A(q, a) = p$ .

B. Juliano © 2002, based on notes by J. Ullman



## Deterministic PDAs

- 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  $y$ .
- **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  $P'$ .

B. Juliano © 2002, based on notes by J. Ullman



## Deterministic PDAs

- **DPDAs and CFLs**
  - The languages accepted by DPDAs by final state properly include the regular languages, but are properly included in the CFLs.
- **DPDAs and Ambiguous Grammars**
  - **Theorem 6.20**
    - If  $L=N(P)$  for some DPDA  $P$ , then  $L$  has an unambiguous CFG  $G$ .
  - **Theorem 6.21**
    - If  $L=L(P)$  for some DPDA  $P$ , then  $L$  has an unambiguous CFG  $G$ .

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- **Alternate construction, CFG<sup>®</sup> PDA**
  - **Idea:** Instead of constructing a PDA with only one state (as in our textbook), create extra temporary states to push additional symbols into the stack. (This models production rules that have a body with more than one symbol.)
  - Given CFG  $G = (V, \Sigma, P, S)$ .
  - Construct PDA  $P = (Q, \Sigma, V \cup \Sigma, \delta, q_s, Z_0, \{q_f\})$  such that  $Q$  contains start state  $q_s$ , final state  $q_f$  and other states to be defined next ...

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- **Alternate construction, *continued*:**
  - $P$ 's transition function,  $\delta$ , contains:
    - $\delta(q_s, \epsilon, \epsilon) = \{(q_f, S)\}$
    - $\delta(q_f, a, a) = \{(q_f, \epsilon)\}$ , for all  $a \in \Sigma$
    - For every rule  $r_i = (A \rightarrow w_1 w_2 \dots w_k)$  in  $P$ , where each  $w_j \in VN\Sigma$ ,  $1 \leq j \leq k$ , create  $k-1$  new states  $q_{i,1}, q_{i,2}, \dots, q_{i,k-1}$  ...

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- **Alternate construction, *continued*:**
  - Add into  $\delta$  as follows:
    - $\delta(q_f, \epsilon, A) \ni (q_{i,1}, w_k)$
    - $\delta(q_{i,j}, \epsilon, \epsilon) = \{(q_{i,j+1}, w_{k-j})\}$ , for  $1 \leq j \leq k-2$
    - $\delta(q_{i,k-1}, \epsilon, \epsilon) = \{(q_f, w_1)\}$
  - If  $k \leq 1$ , i.e.  $r_i = (A \rightarrow w_1)$  where  $w_1 \in VN\Sigma \setminus \{\epsilon\}$ , then define  $(q_f, w_1) \in \delta(q_f, \epsilon, A)$ .

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

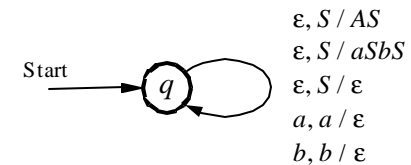
- **Alternate construction, *continued*:**
  - Example: Consider CFG  $G = (\{S\}, \{a, b\}, P, S)$ , where  $P$  contains  $S \rightarrow aS \mid aSbS \mid \epsilon$ .
  - By the construction in our textbook:
    - PDA  $P_1 = (\{q\}, \{a, b\}, \{S, a, b\}, \delta, q, S, \emptyset)$  where
      - $\delta(q, \epsilon, S) = \{(q, aS), (q, aSbS), (q, \epsilon)\}$
      - $\delta(q, a, a) = \{(q, \epsilon)\}$
      - $\delta(q, b, b) = \{(q, \epsilon)\}$

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- **Alternate construction, *continued*:**
  - By the construction in our textbook:
    - The transition diagram for PDA  $P_1$  is



- Note that  $L(G) = N(P_1)$ .

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- **Alternate construction, *continued*:**
  - By alternate construction:
    - Initially, PDA  $P_2 = (\{q_s, q_f\}, \{a, b\}, \{S, a, b\}, \delta, q_s, \varepsilon, \{q_f\})$

$$\delta(q_s, \varepsilon, \varepsilon) = \{(q_f, S)\}$$

$$\delta(q_f, a, a) = \{(q_f, \varepsilon)\}$$

$$\delta(q_f, b, b) = \{(q_f, \varepsilon)\}$$

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- **Alternate construction, *continued*:**
  - By alternate construction:
    - For rule  $r_1 = (S \rightarrow aS)$  in  $P$ :
      - Create  $k-1=1$  new state,  $q_{1,1}$ .
      - Add the following into  $\delta$ :
        - $\delta(q_f, \varepsilon, S) \ni (q_{1,1}, S)$
        - $\delta(q_{1,1}, \varepsilon, \varepsilon) = \{(q_f, a)\}$

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- **Alternate construction, *continued*:**
  - By alternate construction:
    - For rule  $r_2 = (S \rightarrow aSbS)$  in  $P$ :
      - Create  $k-1=3$  new states,  $q_{2,1}, q_{2,2}, q_{2,3}$ .
      - Add the following into  $\delta$ :
        - $\delta(q_f, \varepsilon, S) \ni (q_{2,1}, S)$
        - $\delta(q_{2,1}, \varepsilon, \varepsilon) = \{(q_{2,2}, b)\}$
        - $\delta(q_{2,2}, \varepsilon, \varepsilon) = \{(q_{2,3}, S)\}$
        - $\delta(q_{2,3}, \varepsilon, \varepsilon) = \{(q_f, a)\}$

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

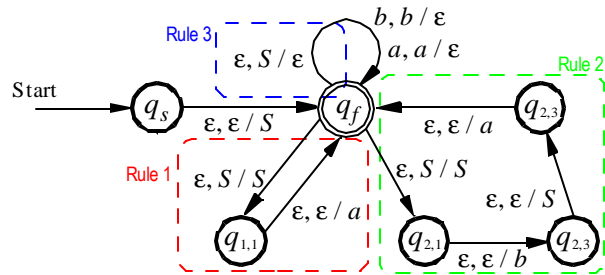
- **Alternate construction, *continued*:**
  - By alternate construction:
    - For rule  $r_3 = (S \rightarrow \varepsilon)$  in  $P$ :
      - *No new states to add!*
      - Add the following into  $\delta$ :
        - $\delta(q_f, \varepsilon, S) \ni (q_f, \varepsilon)$

B. Juliano © 2002, based on notes by J. Ullman



## Equivalence of PDAs & CFGs

- **Alternate construction**, *continued*:
  - By alternate construction:
    - The transition diagram for PDA  $P_2$  is



- Note that  $L(G) = L(P_2)$ .

B. Juliano © 2002, based on notes by J. Ullman

## Copyright and Intellectual Property Notice

This document and parts of its contents are the *Intellectual Property* (IP) of Dr. Benjoe A. Juliano of the Department of Computer Science at California State University, Chico (CSUC). Dr. Juliano claims exclusive moral rights of ownership under current *Copyright Laws* (Title 17 of the United States Code and 1998 Digital Millennium Copyright Act) and *IP Policies/Guidelines* (CSUC EM83-08, EM97-07, and Article 39 of the CFA/CSU Contract) including, but not limited to:

- the exclusive right to copy, reproduce, and/or distribute this document;
- the right to be identified as the creator of this work (the right of attribution);
- the right to take action against false attribution; and
- the right to object to derogatory treatment of this work (the right of integrity).

B. Juliano © 2002, based on notes by J. Ullman

