

Equivalence of PDA, CFG

Conversion of CFG to PDA

Conversion of PDA to CFG

Overview

- ◆ When we talked about closure properties of regular languages, it was useful to be able to jump between RE and DFA representations.
- ◆ Similarly, CFG's and PDA's are both useful to deal with properties of the CFL's.

Overview – (2)

- ◆ Also, PDA's, being "algorithmic," are often easier to use when arguing that a language is a CFL.
- ◆ **Example:** It is easy to see how a PDA can recognize balanced parentheses; not so easy as a grammar.

Converting a CFG to a PDA

- ◆ Let $L = L(G)$.
- ◆ Construct PDA P such that $N(P) = L$.
- ◆ P has:
 - ◆ One state q .
 - ◆ Input symbols = terminals of G .
 - ◆ Stack symbols = all symbols of G .
 - ◆ Start symbol = start symbol of G .

Intuition About P

- ◆ At each step, P represents some *left-sentential form* (step of a leftmost derivation).
- ◆ If the stack of P is α , and P has so far consumed x from its input, then P represents left-sentential form $x\alpha$.
- ◆ At empty stack, the input consumed is a string in $L(G)$.

Transition Function of P

1. $\delta(q, a, a) = (q, \epsilon)$. (*Type 1* rules)
 - ◆ This step does not change the LSF represented, but “moves” responsibility for a from the stack to the consumed input.
2. If $A \rightarrow \alpha$ is a production of G , then $\delta(q, \epsilon, A)$ contains (q, α) . (*Type 2* rules)
 - ◆ Guess a production for A , and represent the next LSF in the derivation.

Proof That $L(P) = L(G)$

- ◆ We need to show that $(q, wx, S) \vdash^* (q, x, \alpha)$ for any x if and only if $S \Rightarrow_{Im}^* W\alpha$.
- ◆ **Part 1:** “only if” is an induction on the number of steps made by P .
- ◆ **Basis:** 0 steps.
 - ◆ Then $\alpha = S$, $w = \epsilon$, and $S \Rightarrow_{Im}^* S$ is surely true.

Induction for Part 1

- ◆ Consider n moves of P : $(q, wx, S) \vdash^* (q, x, \alpha)$ and assume the IH for sequences of $n-1$ moves.
- ◆ There are two cases, depending on whether the last move uses a **Type 1** or **Type 2** rule.

Use of a Type 1 Rule

- ◆ The move sequence must be of the form $(q, yax, S) \vdash^* (q, ax, a\alpha) \vdash (q, x, \alpha)$, where $ya = w$.
- ◆ By the IH applied to the first $n-1$ steps, $S \Rightarrow_{\text{Im}}^* ya\alpha$.
- ◆ But $ya = w$, so $S \Rightarrow_{\text{Im}}^* w\alpha$.

Use of a Type 2 Rule

- ◆ The move sequence must be of the form $(q, wx, S) \vdash^* (q, x, A\beta) \vdash (q, x, \gamma\beta)$, where $A \rightarrow \gamma$ is a production and $\alpha = \gamma\beta$.
- ◆ By the IH applied to the first $n-1$ steps, $S \Rightarrow_{Im}^* wA\beta$.
- ◆ Thus, $S \Rightarrow_{Im}^* w\gamma\beta = w\alpha$.

Proof of Part 2 ("if")

- ◆ We also must prove that if $S \Rightarrow_{lm}^* W\alpha$, then $(q, wx, S) \vdash^* (q, x, \alpha)$ for any x .
- ◆ Induction on number of steps in the leftmost derivation.
- ◆ Ideas are similar; omitted.

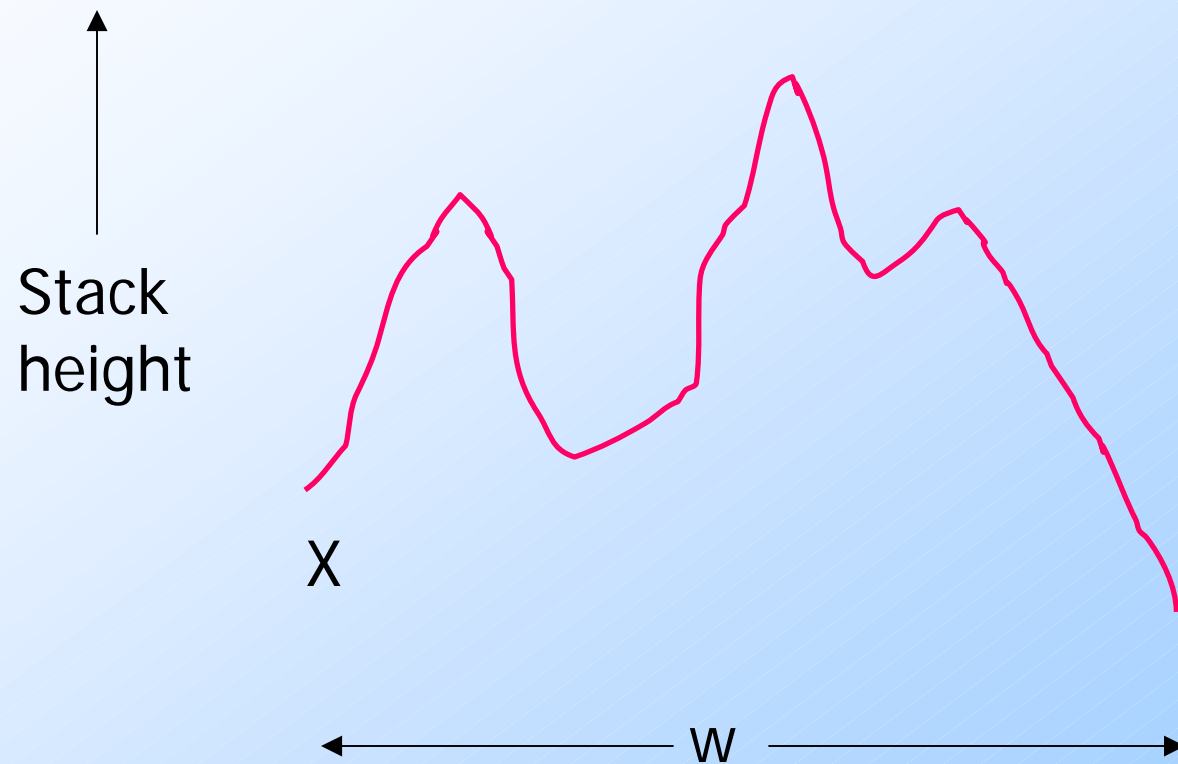
Proof – Completion

- ◆ We now have $(q, wx, S) \vdash^* (q, x, \alpha)$ for any x if and only if $S \Rightarrow_{lm}^* w\alpha$.
- ◆ In particular, let $x = \alpha = \epsilon$.
- ◆ Then $(q, w, S) \vdash^* (q, \epsilon, \epsilon)$ if and only if $S \Rightarrow_{lm}^* w$.
- ◆ That is, w is in $N(P)$ if and only if w is in $L(G)$.

From a PDA to a CFG

- ◆ Now, assume $L = N(P)$.
- ◆ We'll construct a CFG G such that $L = L(G)$.
- ◆ **Intuition**: G will have variables $[pXq]$ generating exactly the inputs that cause P to have the net effect of popping stack symbol X while going from state p to state q .
 - ◆ P never gets below this X while doing so.

Picture: Popping X



Variables of G

- ◆ G's variables are of the form $[pXq]$.
- ◆ This variable generates all and only the strings w such that
$$(p, w, X) \vdash^* (q, \epsilon, \epsilon).$$
- ◆ Also a start symbol S we'll talk about later.

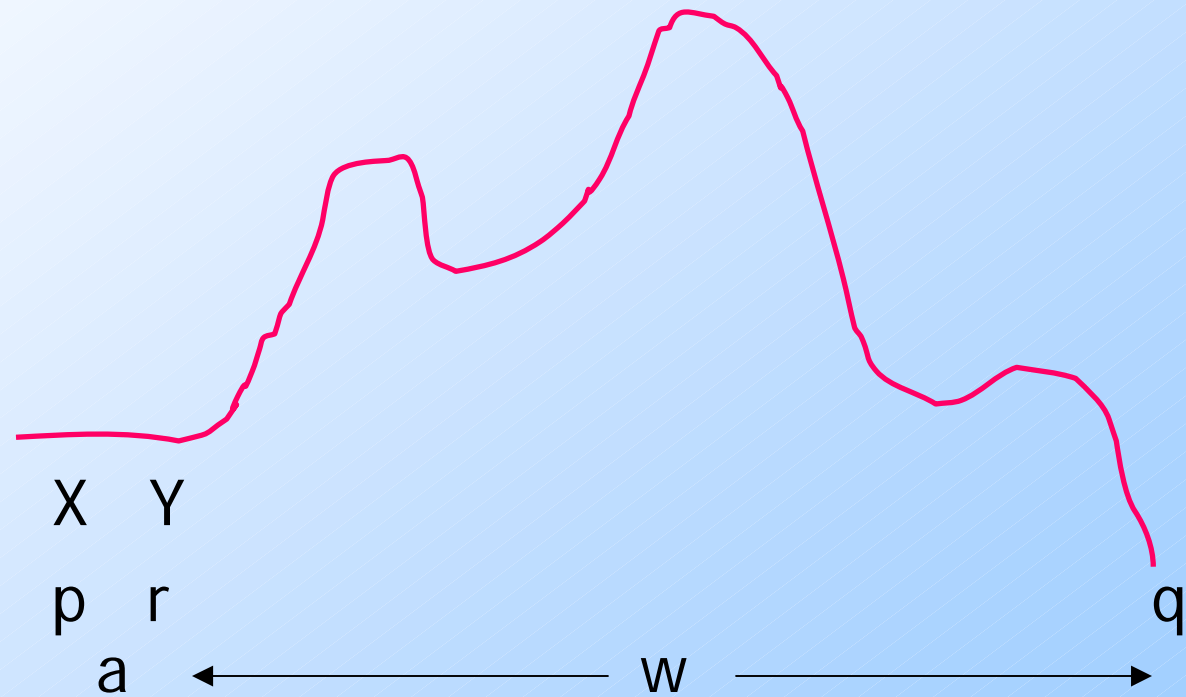
Productions of G

- ◆ Each production for $[pXq]$ comes from a move of P in state p with stack symbol X .
- ◆ **Simplest case:** $\delta(p, a, X)$ contains (q, ϵ) .
 - ◆ Note a can be an input symbol or ϵ .
- ◆ Then the production is $[pXq] \rightarrow a$.
- ◆ Here, $[pXq]$ generates a , because reading a is one way to pop X and go from p to q .

Productions of G – (2)

- ◆ **Next simplest case:** $\delta(p, a, X)$ contains (r, Y) for some state r and symbol Y .
- ◆ G has production $[pXq] \rightarrow a[rYq]$.
 - ◆ We can erase X and go from p to q by reading a (entering state r and replacing the X by Y) and then reading some w that gets P from r to q while erasing the Y .

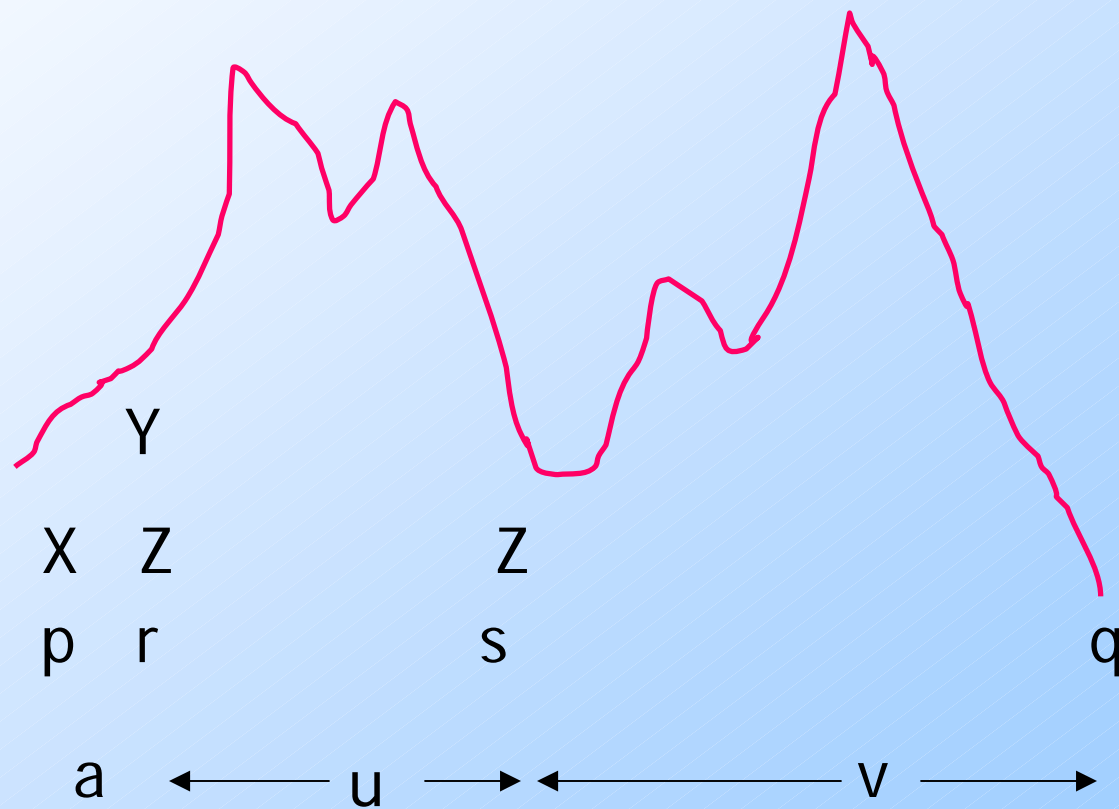
Picture of the Action



Productions of G – (3)

- ◆ **Third simplest case:** $\delta(p, a, X)$ contains (r, YZ) for some state r and symbols Y and Z .
- ◆ Now, P has replaced X by YZ .
- ◆ To have the net effect of erasing X , P must erase Y , going from state r to some state s , and then erase Z , going from s to q .

Picture of the Action



Third-Simplest Case – Concluded

- ◆ Since we do not know state s , we must generate a family of productions:

$$[pXq] \rightarrow a[rYs][sZq]$$

for all states s .

- ◆ $[pXq] \Rightarrow^* auv$ whenever $[rYs] \Rightarrow^* u$ and $[sZq] \Rightarrow^* v$.

Productions of G: General Case

◆ Suppose $\delta(p, a, X)$ contains (r, Y_1, \dots, Y_k) for some state r and $k \geq 3$.

◆ Generate family of productions

$[pXq] \rightarrow$

$a[rY_1s_1][s_1Y_2s_2]\dots[s_{k-2}Y_{k-1}s_{k-1}][s_{k-1}Y_kq]$

Completion of the Construction

- ◆ We can prove that $(q_0, w, Z_0) \vdash^* (p, \epsilon, \epsilon)$ if and only if $[q_0 Z_0 p] \Rightarrow^* w$.
 - ◆ Proof is two easy inductions.
- ◆ But state p can be anything.
- ◆ Thus, add to G another variable S , the start symbol, and add productions $S \rightarrow [q_0 Z_0 p]$ for each state p .