Push Down Automata

Sipser pages 111 - 117

Push Down Automata

Push Down Automata (PDAs) are ε-NFAs with stack memory.

Transitions are labeled by an input symbol together with a pair of the form X/α

.

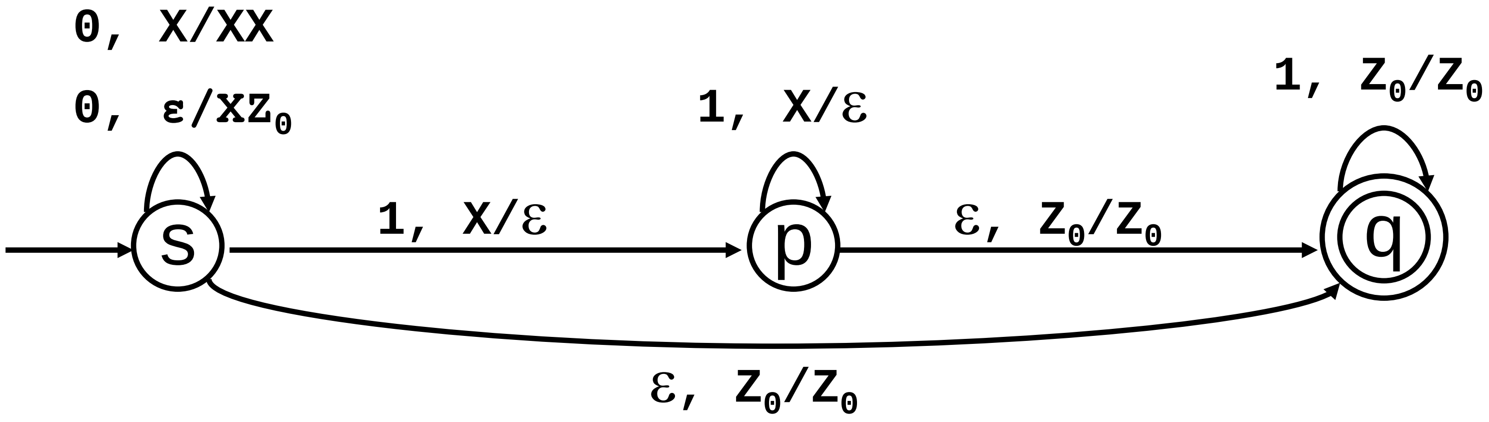
The transition is possible only if the top of the stack contains the symbol X

After the transition, the stack is changed by replacing the top symbol X with the string of symbols α. (Pop X, then push symbols of α.)

Example

PDAs can accept languages that are not regular. The following one accepts:

$$L = \{0^i 1^j \mid 0 \le i \le j\}$$



Definition

A PDA is a 6-tuple $P=(Q,\Sigma,\Gamma,\delta,q_0,F)$ where Q, Σ, q_0, F are as in NFAs, and

- Γ is the *stack alphabet*. It is assumed that initially the stack is empty.

- δ: $Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \longrightarrow P(Q \times \Gamma_\varepsilon{}^*)$ is the *transition function*: given a state, an input symbol (or ε), and a stack symbol, Γ_ε, it gives us a finite number of pairs (q,α), where q is the next state and α is the string of stack symbols that will replace X on top of the stack.

- Recall $\Sigma_\varepsilon = (\Sigma \cup \{\varepsilon\})$ $\quad\Gamma_\varepsilon = (\Gamma \cup \{\varepsilon\})$

In our example, the transition from s to s labeled $(0,\varepsilon/XZ_0)$ corresponds to the fact $(s,XZ_0)\in\delta(s,0,\varepsilon)$. A complete description of the transition function in this example is given by

$\delta(s,0,\varepsilon) = \{(s,XZ_0)\}$

$\delta(s,0,X) = \{(s,XX)\}$

$\delta(s,\varepsilon,Z_0) = \{(q,Z_0)\}$

$\delta(s,1,X) = \{(p,\varepsilon)\}$

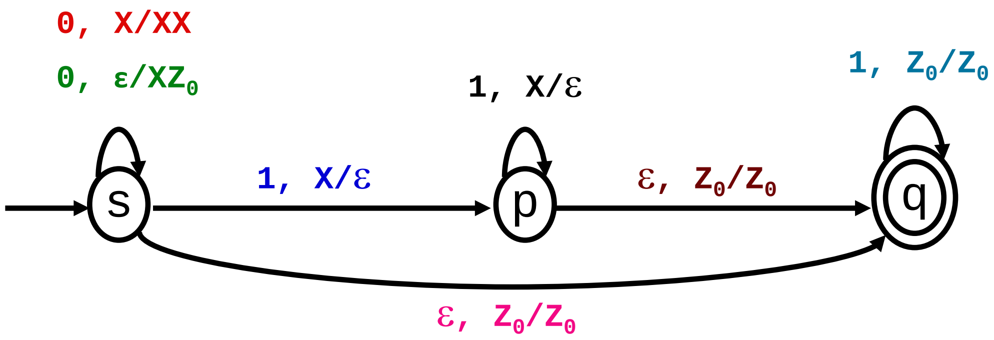
$\delta(p,1,X) = \{(p,\varepsilon)\}$

$\delta(p,\varepsilon,Z_0) = \{(q,Z_0)\}$

$\delta(q,1,Z_0) = \{(q,Z_0)\}$

and

$\delta(q,a,Y)=\emptyset$ for all other possibilities.



Sipser style acceptance

- Suppose a string w can be written: $w_1\ w_2\ \dots\ w_m$
  - $W_i \in \Sigma_\varepsilon$ Some of the w_i are allowed to be ε
  - I.e. One may write "abc" as $a\ \varepsilon\ b\ c\ \varepsilon$
- If there exist two sequences
  - $r_0\ r_1\ \dots\ r_m \in Q$
  - $s_0\ s_1\ \dots\ s_m \in \Gamma^*$ (The s_i represent the stack contents at step i)

1. $r_0=q_0$ and $s_0 = \varepsilon$
2. $(r_{i+1},\alpha) \in \delta(r_i,w_{i+1},A)$
   $s_i = A\beta \quad s_{i+1} = \alpha\beta$
3. $r_m \in F$

The initial state and stack

Corresponding elements in the sequences are related to the next via the transition function.

The last state in the sequence is in the Final states.

Instantaneous Descriptions and Moves of PDAs

IDs (also called *configurations*) describe the execution of a PDA at each instant. An ID is a triple (q, w, α), with this intended meaning:

- q is the current state
- w is the remaining part of the input
- α is the current content of the stack, with top of the stack on the left.

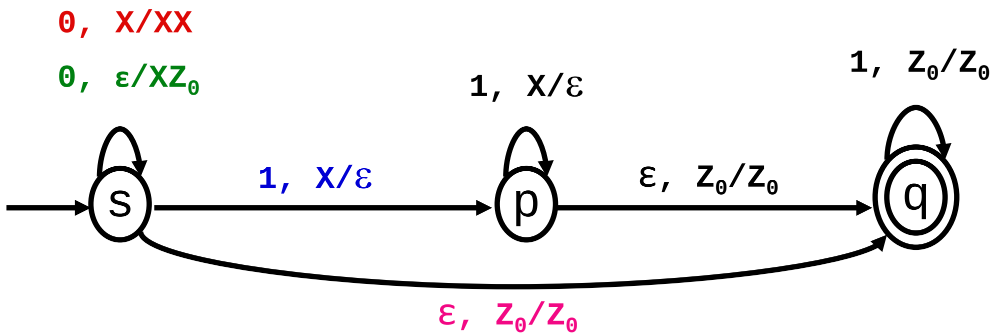
The relation |- describes possible moves from one ID to another during execution of a PDA. If $\delta(q,a,X)$ contains (p,α), then

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

is true for every w and β.

The relation $\vdash^*$ is the reflexive-transitive closure of |-

We have $(q,w,a) \vdash^* (q',w',a')$ when (q,w,a) leads through a sequence (possibly empty) of moves to (q',w',a')



$(s,011,\varepsilon) \mid\!- (s,11,XZ_0) \mid\!- (P,1,Z_0) \mid\!- (q,1,Z_0) \mid\!- (q,\varepsilon,Z_0)$

$(s,011,Z_0) \mid\!- (q,011,Z_0)$

Properties of |-

Property 1.

If $(q,x,\alpha) \mid\!-^* (p,y,\beta)$

Then $(q,xw,\alpha\gamma) \mid\!-^* (p,yw,\beta\gamma)$

If you only need some prefix of the input (x) and stack (α) to make a series of transitions, you can make the same transitions for any longer input and stack.

Property 2.

If $(q,xw,\alpha) \mid\!-^* (p,yw,\beta)$

Then $(q,x,\alpha) \mid\!-^* (p,y,\beta)$

It is ok to remove unused input, since a PDA cannot add input back on once consumed.

Another notion of acceptance

A PDA as above *accepts* the string w iff

$(q_0, w, \varepsilon) \vdash^* (p, \varepsilon, \alpha)$ is true for some final state p and some α. (We don't care what's on the stack at the end of input.)

The *language* L(P) of the PDA P is the set of all strings accepted by P.

Here is the chain of IDs showing that the string `001111` is accepted by our example PDA:

$(s, 001111, \varepsilon)$

$\vdash (s, 01111, XZ_0)$

$\vdash (s, 1111, XXZ_0)$

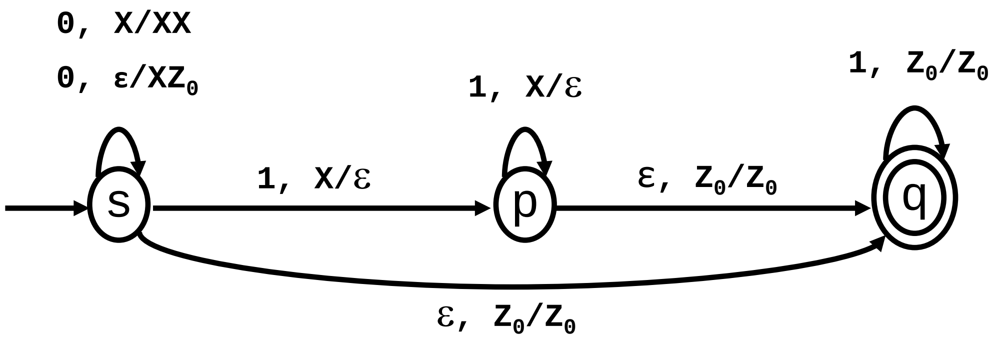
$\vdash (p, 111, XZ_0)$

$\vdash (p, 11, Z_0)$

$\vdash (q, 11, Z_0)$

$\vdash (q, 1, Z_0)$

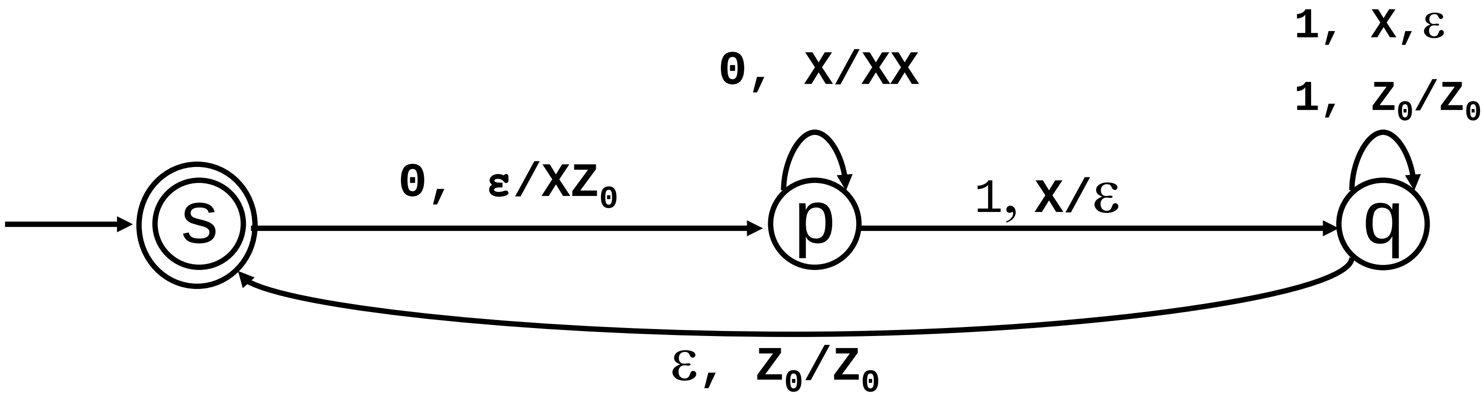
$\vdash (q, \varepsilon, Z_0)$



The language of the following PDA is

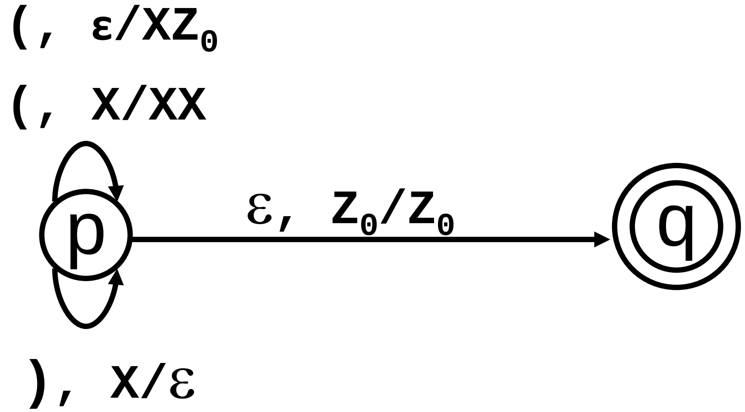
$\{0^i 1^j \mid 0 < i \leq j\}^*$.

How can we prove this?



Example

A PDA for the language of balanced parentheses:



Acceptance by Empty Stack

Define N(P) to be the set of all strings *w* such that

$(q_0, w, \varepsilon) \mid\!-^* (q, \varepsilon, \varepsilon)$

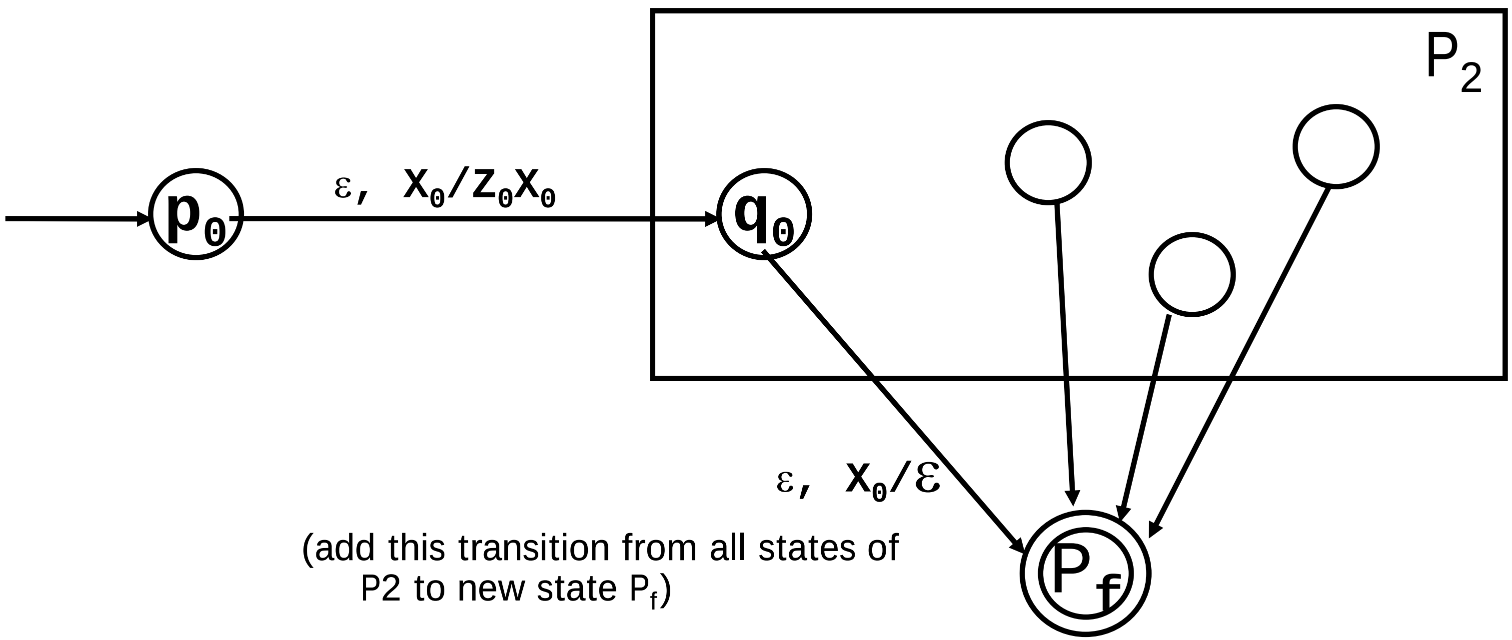
for some state q. These are the strings P *accepts by empty stack*. Note that the set of final states plays no role in this definition.

Theorem. A language is $L(P_1)$ for some PDA P_1 if and only if it is $N(P_2)$ for some PDA P_2.

Proof 1

1. From empty stack to final state.

Given P_2 that accepts by empty stack, get P_1 by adding a new start state and a new final state as in the picture below. We also add a new stack symbol X_0 and make it the start symbol for P_1's stack.



(add this transition from all states of P2 to new state P_f)

Proof 2

2. From final state to empty stack.

Given P_1, we get P_2 again by adding a new start state, final state and start stack symbol. New transitions are seen in the picture.