**Pushdown Automata**

NFA with single, last-in-first-out push down stack 
unlimited memory

input from alphabet $\Sigma$

from stack alphabet $\Gamma$

Stack operations:  
- Read top  
- Remove top (Pop)  
- Write top (Push)

$a, x \to y$  
Reading input $a$ with $x$ on top, replace $x$ with $y$  
$(a, x = \varepsilon)$

PDA $\leftarrow\rightarrow$ CFG

---

**Example PDA**

PDA to recognize $\{0^n1^n | n \geq 0\}$  
(non-regular)

$\Sigma = \{0, 1\}$

$\Gamma = \{0, \$\}$

Push $\$ on stack.

Start reading input. As read 0's, push them on stack.  
When reach 1, for each 1, pop a 0 off stack. If no 0 to pop, reject.  
If reach another 0, after you've read a 1, reject.  
Accept if finish input, and stack has $\$ on top.  
Finish by popping $\$ off stack.  
(Deterministic)
**Formal Definition of PDA**

A PDA $M$ is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$ where,

1. $Q$ is a finite set of states
2. $\Sigma$ is the input alphabet
3. $\Gamma$ is the stack alphabet
4. $\delta: Q \times \Sigma \times \Gamma \rightarrow P(Q \times \Gamma)$
5. $q_0$ in $Q$, the start state
6. $F \subseteq Q$, set of accept (final) states

Ex. $Q = \{q_1, q_2, q_3, q_4\}$ $\Sigma = \{0, 1\}$ $\Gamma = \{\varepsilon\}$ $F = \{q_1, q_4\}$

<table>
<thead>
<tr>
<th>input</th>
<th>0</th>
<th>1</th>
<th>$\varepsilon$</th>
</tr>
</thead>
<tbody>
<tr>
<td>stack</td>
<td>$\varepsilon$</td>
<td>$\varepsilon$</td>
<td>$\varepsilon$</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$q_2, 0$</td>
<td>$q_3, \varepsilon$</td>
<td>$q_4, \varepsilon$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$q_2, \varepsilon$</td>
<td>$q_3, \varepsilon$</td>
<td>$q_4, \varepsilon$</td>
</tr>
<tr>
<td>$q_3$</td>
<td>$q_3, \varepsilon$</td>
<td>$q_4, \varepsilon$</td>
<td>$q_4, \varepsilon$</td>
</tr>
</tbody>
</table>

**State Diagram of PDA**

to recognize $\{0^n1^n | n \geq 0\}$

- $q_1$: input $a$, top stack $b$, replace top stack $c$
  - $a = \varepsilon$ means don’t read input symbol
  - $b = \varepsilon$ means don’t read or pop top of stack
  - $c = \varepsilon$ means don’t write on top
Formal Definition of PDA Acceptance

A PDA M accepts $w = w_1 \ldots w_m$ in $\Sigma^*$ if there is a sequence of states $r_0, r_1, \ldots, r_m$ in $Q$ and strings $s_0 s_1 s_2 \ldots s_m$ in $\Gamma^*$ with:

1. $r_0 = q_0$, $s_0 = \varepsilon$ (starts properly)
2. $(r_{i+1}, b)$ is an element of $\delta(r_i, w_{i+1}, \alpha)$ with $s_i = \alpha t$ and $s_{i+1} = bt$, with $\alpha, b$ in $\Gamma \setminus \{\varepsilon\}$, $t$ in $\Gamma^*$ (moves properly)
3. $r_m$ is in $F$ (ends in final state)

Note: Don’t require stack to be empty to accept.

Example PDA

$L = \{ w w^R | w \in \{a,b\}^* \}$

$L = \{ w | w \text{ has the same number of 0’s and 1’s} \}$
Example PDA

\[ L = \{ a^i b^j c^k \mid i, j, k \geq 0 \text{ and } i = j \text{ or } i = k \} \]

Non-deterministically check that \( \# a's = \# b's \) OR
\( \# a's = \# c's \)

---

Pumping Lemma for CFL's

If \( A \) is a context-free language, then there is a no. \( p \) (pumping length) where, if \( s \) is any string in \( A \) of length at least \( p \), \( s \) may be divided into 5 pieces \( u, v, x, y, z \), \( s = uvxyz \), such that all of the following hold:

1. for each \( i \geq 0 \), \( uv^ixyz \) is in \( A \)
2. \( |vy| > 0 \)
3. \( |vxy| \leq p \)

Condition 1 lets us "pump out" elements in \( A \)

Note that either \( v \) or \( y \) can be \( \epsilon \), but both cannot be (by 2)
(Without 2, the lemma is trivially true, with \( v = y = \epsilon \))

Condition 3 assures us we can make \( vxy \) small, if needed.
Pumping Lemma Example

L = \{ a^n b^n c^n \mid n \geq 0 \} is not context-free.

Suppose L were CF. Then let p be the pumping length given by the pumping lemma.
Let s = a^p b^p c^p in L.

Note that |s| > p, so s = uvxyz as in the lemma.

Case 1: v and y contain only 1 type of symbol.
(E.g., v contains only a’s, and y only b’s)
Since there are 3 symbols in the alphabet of L,
there is a third type of symbol not in v or y.
Consider uvvxyyz. It contains more of the first
two symbols than the third; contradiction.

Pumping Lemma Example, Continued

Case 2: Either v or y contains at least 2 symbols.
(E.g., v contains a’s and b’s)
Consider uvvxyyz. It contains symbols out of
order in the part that repeats and contains at
least 2 symbols. Contradiction.
These are the only cases. End of proof.

Ex. v = ab. Then uababxyyz has substring abab, which
is not allowed.
**Pumping Lemma Example**

L = \{ww \mid w \in \{0,1\}^*\} is not context-free.

Suppose L were CF. Then let p be the pumping length given by the pumping lemma.
Let s = 0^p10^p1^p in L.

Note that |s| > p, so s = uvxyz as in the lemma.

*Case 1:* vxy is in the first half of the string
Then uvvxyz will have the second half
of the string start with a 1; but the first half starts
with 0. So it cannot be of form ww. Contradiction.

*Case 2:* vxy is in the second half.
Follows similarly to case 1.

---

**Pumping Lemma Example, continued**

*Case 3:* vxy is in the middle of the string.
Then uxz is of the form \(0^i10^j1\), where
i and j are not both p. This string is not
of form ww. Contradiction.

*Qu:* What about uvvxyz?

Could we have used \(0^p10^p1\) for s in the proof?