Formal Definition of PDA

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

- $Q$ is a finite set of states;
- $\Sigma$ is the input alphabet;
- $\Gamma$ is the stack alphabet;
- $\delta : Q \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \rightarrow \mathcal{P}(Q \times (\Gamma \cup \{\lambda\}))$ is the transition function;
- $q_0 \in Q$ is the start state; and
- $F \subseteq Q$ is the set of final or accepting states.
The Model

A typical mental model for a PDA looks like this:
Characteristics of PDAs

- A PDA is inherently nondeterministic. So really an extension of the NFA model.
- The push/pop head has read/write access to the top of the stack.
- \((q_j, B) \in \delta(q_i, a, A)\) means the PDA, when in state \(q_i\), reading symbol \(a\), and having symbol \(A\) on top of stack, is allowed to do the following in one step:
  - Consume \(a\);
  - Pop \(A\);
  - Push \(B\); and
  - Change to state \(q_j\).
- Also have the possibility of consuming \(\lambda\), popping \(\lambda\), or pushing \(\lambda\).
Example Using The Model

Let $M_1$ be the PDA given by:

$$M_1 = (Q_1, \Sigma, \Gamma, \delta_1, q_0, F_1)$$

$Q_1 = \{q_0\}$

$\Sigma = \{a\}$

$\Gamma = \{A\}$

$F_1 = \{q_0\}$

The transition function

$$\delta : Q \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \rightarrow \mathcal{P}(Q \times (\Gamma \cup \{\lambda\}))$$

is given by this table:

<table>
<thead>
<tr>
<th>$\delta_1(q_0, \lambda, \lambda)$</th>
<th>$\delta_1(q_0, a, \lambda)$</th>
<th>$\delta_1(q_0, \lambda, A)$</th>
<th>$\delta_1(q_0, a, A)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\emptyset$</td>
<td>${(q_0, A)}$</td>
<td>$\emptyset$</td>
<td>${(q_0, \lambda)}$</td>
</tr>
</tbody>
</table>

Try the model on input $aaaaaa$. 
Acceptance

Informally, a string \( w \in \Sigma^* \) is accepted by \( M \) if (1) \( M \) is in a final state after consuming all of \( w \) and (2) the stack is empty.

To define acceptance formally, we again need the notions of configuration and of the yields relation.
Configurations

A configuration of $M$ is an element of $Q \times \Sigma^* \times \Gamma^*$.

For an input $w \in \Sigma^*$, the initial or start configuration is

$$(q_0, w, \lambda).$$

The yields (in one step) relation $\vdash_M$ is a binary relation on $Q \times \Sigma^* \times \Gamma^*$. Assuming $q_i, q_j \in Q$, $u \in \Sigma^*$, $v \in \Gamma^*$, $\sigma \in \Sigma \cup \{\lambda\}$, and $\gamma \in \Gamma \cup \{\lambda\}$, we have

$$(q_i, \sigma u, \gamma v) \vdash_M (q_j, u, \rho v)$$

if and only if

$$(q_j, \rho) \in \delta(q_i, \sigma, \gamma).$$
Example

For the previous example $M_1$ and the input $w = aaaaa$, the initial configuration is

$$(q_0, aaaaa, \lambda).$$

The initial configuration yields what configurations in one step:

$$(q_0, aaaaa, \lambda) \vdash_{M_1} ?$$

What is the computation tree whose nodes are configurations and whose children are
gotten through the yields relation $?$
Yields in $t$ Steps

As with DFAs and NFAs, we have the notion of **yields in $t$ steps**:

$$(q_i, w, x) \xrightarrow{t \ M} (q_j, y, z).$$

Yields in Zero or More Steps

Taking the union of all these relations, we get **yields (in zero or more steps)**:

$$(q_i, w, x) \xrightarrow{* \ M} (q_j, y, z)$$

holds if and only if there exists a $t \geq 0$ such that

$$(q_i, w, x) \xrightarrow{t \ M} (q_j, y, z).$$
Acceptance Again

A string $w \in \Sigma^*$ is accepted by the PDA $M$ if

$$(q_0, w, \lambda) \xrightarrow{\ast} (q_i, \lambda, \lambda),$$

where $q_i \in F$.

The language $L(M)$ accepted by the PDA $M$ is the set of all strings accepted by $M$.

Said another way,

$$L(M) = \left\{ w \in \Sigma^* \mid (q_0, w, \lambda) \xrightarrow{\ast}_M (q_i, \lambda, \lambda) \right\}.$$

for some $q_i \in F$. 
Example

Let $M_2 = (\{q_0, q_1\}, \{a, b\}, \{A\}, \delta_2, q_0, \{q_1\})$, where

$$\delta_2(q_0, a, \lambda) = \{(q_0, A)\}$$

$$\delta_2(q_0, b, A) = \{(q_1, \lambda)\}$$

$$\delta_2(q_1, b, A) = \{(q_1, \lambda)\}$$

(all other transitions are $\emptyset$).

Argue that

$$L(M_2) = \{a^i b^i \mid 1 \leq i \}.$$
Another Example

Let $M_3 = (\{q_0, q_1\}, \{0, 1\}, \{A, B\}, \delta_3, q_0, \{q_1\})$, where

\[
\delta_3(q_0, 0, \lambda) = \{(q_0, A)\}
\]
\[
\delta_3(q_0, 1, \lambda) = \{(q_0, B)\}
\]
\[
\delta_3(q_0, \lambda, \lambda) = \{(q_1, \lambda)\}
\]
\[
\delta_3(q_1, 0, A) = \{(q_1, \lambda)\}
\]
\[
\delta_3(q_1, 1, B) = \{(q_1, \lambda)\}
\]

(all other transitions are $\emptyset$).

Argue that

\[
L(M_3) = \{ww^R \mid w \in \{0, 1\}^*\}.
\]
State Diagram

The state diagram for $M_3$ is

How would you modify the state diagram to accept

$$\{w \in \{0, 1\}^* \mid w = w^R\}.$$
Exercises

Design a PDA $M_4$ that accepts the following language:

$$L_4 = \{w \in \{0, 1\}^* \mid n_0(w) = n_1(w)\}$$

Design a PDA $M_5$ that accepts the following language:

$$L_5 = \{w \in \{a, b\}^* \mid n_a(w) = 2n_b(w)\}$$
Equivalence of PDAs and CFGs

We want to show that the languages accepted by PDAs are exactly the context-free languages.

There are two implications to show:

1. Any language generated by a CFG can be accepted by a PDA.

2. Any language accepted by a PDA can be generated by a CFG.
PDAs Accept CFLs

Start with an arbitrary context-free language $L$.

Let $G = (V, \Sigma, P, S)$ be any CFG that generates $L$.

We need a PDA $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ that accepts exactly the strings that $G$ generates.

This is easily accomplished using top-down parsing. We need two kinds of transitions:

1. For each rule $A \rightarrow \alpha$ in $P$, we need transitions to pop $A$ from the top of the stack and push $\alpha$ (in reverse order).

2. For each element $\sigma \in \Sigma$, we need a transition to match $\sigma$ in the input and on top-of-stack.
Example Construction

Start with the language

\[ L_6 = \{0^n1^m0^m1^n \mid m, n \geq 0 \}. \]

A grammar \( G_6 \) that generates \( L_6 \) is given by:

\[
\begin{align*}
S & \rightarrow 0S1 \mid A \\
A & \rightarrow 1A0 \mid \lambda.
\end{align*}
\]

Construct a PDA.

\[ M_6 = (Q_6, \{0, 1\}, \{0, 1, S, A\}, \delta_6, q_0, F_6), \]

where transitions accomplish the following:

1. For each rule \( A \rightarrow \alpha \) in \( P \), construct transitions to pop \( A \) from the top of the stack and push \( \alpha \) (in reverse order).

2. For each element \( \sigma \in \Sigma \), construct a transition to match \( \sigma \) in the input and on top-of-stack.

Show the operation of the PDA on these inputs: (1) 01; (2) 100.
**CFGs Generate PDA Languages**

Let $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ be any PDA. We want a CFG $G = (V, \Sigma, P, S)$ that generates $L(M)$.

**Key idea:** A leftmost derivation in $G$ should mimic a computation of $M$.

In general, a leftmost derivation in $G$ should result in a sentential form:

$$ S \xrightarrow{L} a_1 a_2 \cdots a_k B_1 B_2 \cdots B_j, $$

where $B_1 B_2 \cdots B_j$ is a string of nonterminals that represents the contents of the stack and the state of $M$. 
Example

Let $M_7 = (Q_7, \Sigma_7, \Gamma_7, \delta_7, q_0, F_7)$ where

$$Q_7 = \{q_0, q_1, q_2\}$$

$$\Sigma_7 = \{a\}$$

$$\Gamma_7 = \{A\}$$

$$F_7 = \{q_2\}$$

and the nontrivial transitions are:

$$\delta_7(q_0, a, \lambda) = \{(q_0, A)\}$$

$$\delta_7(q_0, \lambda, \lambda) = \{(q_1, \lambda)\}$$

$$\delta_7(q_1, a, A) = \{(q_2, \lambda)\}$$

$$\delta_7(q_2, a, A) = \{(q_2, \lambda)\}.$$

Draw the state diagram for $M_7$.

What is $L(M_7)$?
Example Computation

One accepting computation of $M_7$ is

$(q_0, aaaa, \lambda) \vdash (q_0, aaa, A) \vdash (q_0, aa, AA) \vdash (q_1, aa, AA) \vdash (q_2, a, A) \vdash (q_2, \lambda, \lambda)$.

How can we represent the configuration $(q_0, aa, AA)$, for example?

Try this picture:

```
  aa    A    A
  ↑    ↑    ↑
 q0  q2  q2
```

Extracting the configuration $(q_0, aa, AA)$ from this picture is easy.

In addition, the intermediate states are “guessed” by nondeterminism to be the future states of $M_7$. 
Example Representation

Here is the picture again:

\[
\begin{array}{ccc}
    aa & A & A \\
    \uparrow & \uparrow & \uparrow \\
    q_0 & q_2 & q_2 \\
\end{array}
\]

Here is a representation using a string, instead of a picture:

\[ aa \langle q_0, A, q_2 \rangle \langle q_2, A, q_2 \rangle. \]

The triples \( \langle q_0, A, q_2 \rangle \) and \( \langle q_2, A, q_2 \rangle \) are thought of as single characters, in fact, as nonterminals of a CFG.
Constructing $G$

We now construct a CFG $G = (V, \Sigma, P, S)$ that generates $L(M)$.

The set of non-terminals (variables) is

$$V = \{ S \} \cup \{ \langle q_i, \tau, q_j \rangle \mid q_i, q_j \in Q \text{ and } \tau \in \Gamma \cup \{ \lambda \} \}.$$

Still need to get the rules in $P$. 
Constructing $G$ (Continued)

The rules in $P$ are gotten as follows:

1. For each $q_j \in F$,
   \[ S \rightarrow \langle q_0, \lambda, q_j \rangle. \]
   (Guess accepting configuration $(q_j, \lambda, \lambda)$.)

2. For each $(q_j, B) \in \delta(q_i, x, A)$, where $A \in \Gamma \cup \{\lambda\}$, and for every $q_k \in Q$,
   \[ \langle q_i, A, q_k \rangle \rightarrow x \langle q_j, B, q_k \rangle. \]
   (Normal computation step.)

3. For each $A \in \Gamma \cup \{\lambda\}$, and for every $q_i, q_n, q_k \in Q$,
   \[ \langle q_i, A, q_k \rangle \rightarrow \langle q_i, \lambda, q_n \rangle \langle q_n, A, q_k \rangle. \]
   (Put $\lambda$ on stack; guess intermediate $q_n$.)

4. For each $q_k \in Q$,
   \[ \langle q_k, \lambda, q_k \rangle \rightarrow \lambda. \]
   (Eliminate $\lambda$’s on stack.)
Example

As before, let

\[ M_7 = (\{q_0, q_1, q_2\}, \{a\}, \{A\}, \delta_7, q_0, \{q_2\}), \]

where the non-trivial transitions are

\[ \delta_7(q_0, a, \lambda) = \{(q_0, A)\} \]
\[ \delta_7(q_0, \lambda, \lambda) = \{(q_1, \lambda)\} \]
\[ \delta_7(q_1, a, A) = \{(q_2, \lambda)\} \]
\[ \delta_7(q_2, a, A) = \{(q_2, \lambda)\}. \]

Construct an equivalent CFG \( G_7 \).
Constructing $G_7$

The set of non-terminals is

$$V = \{S\} \cup \{ \langle q_i, \tau, q_j \rangle \mid q_i, q_j \in Q \text{ and } \tau \in \Gamma \cup \{\lambda\} \}$$

$$= \{S, \langle q_0, A, q_0 \rangle, \langle q_0, A, q_1 \rangle, \langle q_0, A, q_2 \rangle, \langle q_1, A, q_0 \rangle, \langle q_1, A, q_1 \rangle, \langle q_1, A, q_2 \rangle, \langle q_2, A, q_0 \rangle, \langle q_2, A, q_1 \rangle, \langle q_2, A, q_2 \rangle, \langle q_0, \lambda, q_0 \rangle, \langle q_0, \lambda, q_1 \rangle, \langle q_0, \lambda, q_2 \rangle, \langle q_1, \lambda, q_0 \rangle, \langle q_1, \lambda, q_1 \rangle, \langle q_1, \lambda, q_2 \rangle, \langle q_2, \lambda, q_0 \rangle, \langle q_2, \lambda, q_1 \rangle, \langle q_2, \lambda, q_2 \rangle \}.$$
What are the rules of $G_7$?

1. For each $q_j \in F$,

   $S \rightarrow \langle q_0, \lambda, q_j \rangle$.

   (Guess accepting configuration $(q_j, \lambda, \lambda)$.)

   $S' \rightarrow \langle q_0, \lambda, q_2 \rangle$
2. For each \((q_j, B) \in \delta(q_i, x, A)\), where \(A \in \Gamma \cup \{\lambda\}\), and for every \(q_k \in Q\),

\[
\langle q_i, A, q_k \rangle \rightarrow x \langle q_j, B, q_k \rangle.
\]

(Normal computation step.)

\[
\begin{align*}
\langle q_0, \lambda, q_0 \rangle & \rightarrow a \langle q_0, A, q_0 \rangle \\
\langle q_0, \lambda, q_1 \rangle & \rightarrow a \langle q_0, A, q_1 \rangle \\
\langle q_0, \lambda, q_2 \rangle & \rightarrow a \langle q_0, A, q_2 \rangle \\
\langle q_0, \lambda, q_0 \rangle & \rightarrow \langle q_1, \lambda, q_0 \rangle \\
\langle q_0, \lambda, q_1 \rangle & \rightarrow \langle q_1, \lambda, q_1 \rangle \\
\langle q_0, \lambda, q_2 \rangle & \rightarrow \langle q_1, \lambda, q_2 \rangle \\
\langle q_1, A, q_0 \rangle & \rightarrow a \langle q_2, \lambda, q_0 \rangle \\
\langle q_1, A, q_1 \rangle & \rightarrow a \langle q_2, \lambda, q_1 \rangle \\
\langle q_1, A, q_2 \rangle & \rightarrow a \langle q_2, \lambda, q_2 \rangle \\
\langle q_2, A, q_0 \rangle & \rightarrow a \langle q_2, \lambda, q_0 \rangle \\
\langle q_2, A, q_1 \rangle & \rightarrow a \langle q_2, \lambda, q_1 \rangle \\
\langle q_2, A, q_2 \rangle & \rightarrow a \langle q_2, \lambda, q_2 \rangle
\end{align*}
\]
3. For each \( A \in \Gamma \cup \{\lambda\} \), and for every
\( q_i, q_n, q_k \in Q \),

\[
\langle q_i, A, q_k \rangle \rightarrow \langle q_i, \lambda, q_n \rangle \langle q_n, A, q_k \rangle.
\]

(Put \( \lambda \) on stack; guess intermediate \( q_n \).)

\[
\begin{align*}
\langle q_0, A, q_0 \rangle & \rightarrow \langle q_0, \lambda, q_0 \rangle \langle q_0, A, q_0 \rangle \\
\langle q_0, A, q_1 \rangle & \rightarrow \langle q_0, \lambda, q_0 \rangle \langle q_0, A, q_1 \rangle \\
\langle q_0, A, q_0 \rangle & \rightarrow \langle q_0, \lambda, q_1 \rangle \langle q_1, A, q_0 \rangle \\
\langle q_0, A, q_1 \rangle & \rightarrow \langle q_0, \lambda, q_1 \rangle \langle q_1, A, q_1 \rangle \\
\langle q_0, A, q_0 \rangle & \rightarrow \langle q_0, \lambda, q_2 \rangle \langle q_2, A, q_0 \rangle \\
\langle q_0, A, q_1 \rangle & \rightarrow \langle q_0, \lambda, q_2 \rangle \langle q_2, A, q_1 \rangle \\
\langle q_0, \lambda, q_0 \rangle & \rightarrow \langle q_0, \lambda, q_0 \rangle \langle q_0, \lambda, q_0 \rangle \\
\langle q_0, \lambda, q_1 \rangle & \rightarrow \langle q_0, \lambda, q_0 \rangle \langle q_0, \lambda, q_1 \rangle \\
\langle q_0, \lambda, q_0 \rangle & \rightarrow \langle q_0, \lambda, q_1 \rangle \langle q_1, \lambda, q_0 \rangle \\
\langle q_0, \lambda, q_1 \rangle & \rightarrow \langle q_0, \lambda, q_1 \rangle \langle q_1, \lambda, q_1 \rangle \\
\langle q_0, \lambda, q_0 \rangle & \rightarrow \langle q_0, \lambda, q_2 \rangle \langle q_2, \lambda, q_0 \rangle \\
\langle q_0, \lambda, q_1 \rangle & \rightarrow \langle q_0, \lambda, q_2 \rangle \langle q_2, \lambda, q_1 \rangle \\
\end{align*}
\]

and how many more ?
4. For each $q_k \in Q$, 

$$\langle q_k, \lambda, q_k \rangle \rightarrow \lambda.$$

(Eliminate $\lambda$’s on stack.)

$$\langle q_0, \lambda, q_0 \rangle \rightarrow \lambda$$

$$\langle q_1, \lambda, q_1 \rangle \rightarrow \lambda$$

$$\langle q_2, \lambda, q_2 \rangle \rightarrow \lambda$$
Computations in $M_7$

Now consider this computation of $M_7$

$$(q_0, aaaa, \lambda) \vdash (q_0, aaa, A) \vdash (q_0, aa, AA)$$
$$(q_1, aa, AA) \vdash (q_2, a, A) \vdash (q_2, \lambda).$$

What is the corresponding derivation in $G_7$?

See next slide
Example Continued

\[
S \xrightarrow{L} \langle q_0, \lambda, q_2 \rangle \\
\xrightarrow{L} a \langle q_0, A, q_2 \rangle \\
\xrightarrow{L} a \langle q_0, \lambda, q_2 \rangle \langle q_2, A, q_2 \rangle \\
\xrightarrow{L} aa \langle q_0, A, q_2 \rangle \langle q_2, A, q_2 \rangle \\
\xrightarrow{L} aa \langle q_0, \lambda, q_1 \rangle \langle q_1, A, q_2 \rangle \langle q_2, A, q_2 \rangle \\
\xrightarrow{L} aa \langle q_1, \lambda, q_1 \rangle \langle q_1, A, q_2 \rangle \langle q_2, A, q_2 \rangle \\
\xrightarrow{L} aa \langle q_1, A, q_2 \rangle \langle q_2, A, q_2 \rangle \\
\xrightarrow{L} aaa \langle q_2, \lambda, q_2 \rangle \langle q_2, A, q_2 \rangle \\
\xrightarrow{L} aaa \langle q_2, A, q_2 \rangle \\
\xrightarrow{L} aaaa \langle q_2, \lambda, q_2 \rangle \\
\xrightarrow{L} aaaaa
\]
Exercise

Let

\[ M_8 = (\{q_0\}, \{a, b\}, \{A\}, \delta_8, q_0, \{q_0\}) , \]

where the non-trivial transitions are

\[ \delta_8(q_0, a, \lambda) = \{(q_0, A)\} , \]
\[ \delta_8(q_0, \lambda, \lambda) = \{(q_0, A)\} , \]
\[ \delta_8(q_0, b, A) = \{(q_0, \lambda)\} . \]

Construct an equivalent CFG \( G_8 \) ?
What is Pumping in a CFL?

Consider this CFG

\[ S \rightarrow aAb \mid ba \]
\[ A \rightarrow aAb \mid bbS_{ab} \]

and this derivation

\[ S \Rightarrow aAb \]
\[ \Rightarrow aaAb \]
\[ \Rightarrow aabbS_{abbb} \]
\[ \Rightarrow aabbbbaabbb. \]

Generalizing, for all \( i \geq 0 \),

\[ S \Rightarrow^* (aabb)^i ba(abbb)^i. \]

Pumping in CFLs involves two substrings pumping simultaneously.
Pumping Lemma (Theorem 8.4.3).
Suppose that $L$ is a context-free language. Then there exists an integer $k$ such that for all strings $z \in L$ satisfying $|z| \geq k$, we can write $z = uvwx$ where

1. $|vx| \geq 1$;

2. $|vwx| \leq k$; and

3. $uv^iwx^i \in L$, for all $i \geq 0$. 

Recall:

Chomsky Normal Form

A CFG $G$ is in **Chomsky normal form** if all rules have one of these forms:

1. $A \rightarrow BC$, where $B \neq S$ and $C \neq S$;
2. $A \rightarrow a$;
3. $S \rightarrow \lambda$.

**Observation.** If $w \in L(G)$ and a derivation tree for $w$ has depth $n$, then $|w| \leq 2^n$.

**Theorem 5.4.2.** For every CFL $L$, there is a Chomsky normal form grammar that generates $L$. 

---

**Pushdown Automata**

**Pumping Lemma**
Proof of the Pumping Lemma

Let $G = (V, \Sigma, P, S)$ be a Chomsky normal form grammar that generates $L$. Choose $k = 2|V| + 1$. Let $z$ be any string in $L$ satisfying $|z| \geq k$.

Let $T$ be a derivation tree having yield $z$. Since $|z| > 2|V|$, the depth of $T$ is $>|V|$. Let $R$ be a path in $T$ from the root to a leaf that has length $>|V|$. Because of its length, $R$ must contain some nonterminal twice.

Let $A$ be the nonterminal on $R$ that is the root of the lowest subtree, call it $T_1$, containing a repetition of $A$ and no other repetitions on any path in the subtree. Let $T_2$ be the subtree within $T_1$ that is rooted at the repeated $A$.

See next figure.
Proof Continued

Then

1. $|vx| \geq 1$;
2. $|vw| \leq k$; and
3. $uv^iwx^iy \in L$, for all $i \geq 0$. 
Proving a Language is Not Context Free

Proof Strategy

• Start with a language $L$ that you want to show is not context-free.

• Let $k$ be an arbitrary positive integer.

• Choose $z \in L$ such that $|z| \geq k$ and $z$ does not pump. (This is the creative part.) Note that $z$ is a function of $k$.

• Show that for any decomposition $z = uvwxy$ with $|vx| \geq 1$ and $|vwx| \leq k$, there exists an $i \geq 0$ such that $uv^iwx^iy \notin L$. 
Example

The language

$$L_1 = \{a^i b^i c^i \mid i \geq 0\}$$

is not context-free.

Proof: Let $k > 0$ be arbitrary. Let $z = a^k b^k c^k$. Then $z \in L_1$ and $|z| \geq k$.

Let $z = uvwxy$ be any decomposition of $z$ satisfying $|vx| \geq 1$ and $|vwx| \leq k$. Using the restriction $|vwx| \leq k$, we obtain five cases.
Five Cases

1. All $a$’s in $vwx$. Then $v = a^r$ and $x = a^s$, where $r + s \geq 1$. Therefore, 
   
   \[ uv^2wx^2y = a^{k+r+s}b^kc^k \notin L_1. \]

2. All $b$’s in $vwx$. Similar to 1.

3. All $c$’s in $vwx$. Similar to 1.

4. Some $a$’s and $b$’s in $vwx$. Then $v = a^pb^q$ and $x = a^r b^s$, where $p + q + r + s \geq 1$.

   Let $z' = uv^0wx^0y = uwy$. Then
   
   \[ n_a(z') = k - p - r, \quad n_b(z') = k - q - s, \quad \text{and} \]
   
   \[ n_c(z') = k. \]

   Hence either $n_a(z') < n_c(z')$ or $n_b(z') < n_c(z')$. In any case, $z' \notin L_1$.

5. Some $b$’s and $c$’s in $vwx$. Similar to 4.

As $L_1$ does not satisfy the conclusion of the Pumping Lemma, it cannot be context free.
Example

The language

\[ L_2 = \{ww \mid w \in \{0,1\}^*\} \]

is not context-free.

**Proof:** Let \( k > 0 \) be arbitrary. Let \( z = 0^k10^k1 \). Then \( z \in L_2 \) and \( |z| \geq k \).

Does \( z \) pump?
Example — Try Again

The language

\[ L_2 = \{ww \mid w \in \{0,1\}^*\} \]

is not context-free.

**Proof:** Let \( k > 0 \) be arbitrary. Let \( z = 0^k1^k0^k1^k \). Then \( z \in L_2 \) and \( |z| \geq k \).

Let \( z = uvwxy \) be any decomposition of \( z \) satisfying \( |vx| \geq 1 \) and \( |vwx| \leq k \). Using the restriction \( |vwx| \leq k \), we obtain seven cases.
Seven Cases

1. \( vwx \) occurs in first substring of \( 0's \).
   Then \( v = 0^r \) and \( x = 0^s \), where \( r + s \geq 1 \).
   Therefore, \( uv^0wx^0y = 0^{k-r-s}1^k0^k1^k \not\in L_2 \).

2. \( vwx \) occurs in first substring of \( 1's \).
   Then \( v = 1^r \) and \( x = 1^s \), where \( r + s \geq 1 \).
   Therefore, \( uv^0wx^0y = 0^k1^k1^{k-r-s}0^k1^k \not\in L_2 \).

3. \( vwx \) occurs in second substring of \( 0's \).
   Then \( v = 0^r \) and \( x = 0^s \), where \( r + s \geq 1 \).
   Therefore, \( uv^0wx^0y = 0^k1^k0^k1^{k-r-s}1^k \not\in L_2 \).

4. \( vwx \) occurs in second substring of \( 1's \).
   Then \( v = 1^r \) and \( x = 1^s \), where \( r + s \geq 1 \).
   Therefore, \( uv^0wx^0y = 0^k1^k0^k1^{k-r-s} \not\in L_2 \).
Seven Cases (Concluded)

5. \(vwx\) begins in first substring of 0’s and ends in first substring of 1’s. Then \(v = 0^p1^q\) and \(x = 0^r1^s\), where \(p + q + r + s \geq 1\). Therefore, 
\[
uv^0wx^0y = uwy = 0^{k-p-r}1^{k-q-s}0^k1^k \notin L_2.
\]

6. \(vwx\) begins in first substring of 1’s and ends in second substring of 0’s. Similar to 5.

7. \(vwx\) begins in second substring of 0’s and ends in second substring of 1’s. Similar to 5.

As \(L_2\) does not satisfy the conclusion of the Pumping Lemma, it cannot be context free.
Closure Properties for CFLs

Theorem 8.5.1. Let $L_1$ and $L_2$ be CFLs. Then the following are also CFLs: $L_1 \cup L_2$, $L_1L_2$, and $L_1^*$. 

Theorem 8.5.2. The set of CFLs is not closed under intersection or complementation.

Theorem 8.5.3. Let $L_1$ be a regular language and let $L_2$ be a CFL. Then $L_1 \cap L_2$ is a context-free language.
Exercise

Show that the language
\[ L_3 = \{ w \in \{a, b, c\}^* \mid n_a(w) = n_b(w) = n_c(w) \} \]
is not context-free.

Recall that a string \( w \in \{0, 1\} \) can be interpreted as a binary representation of an integer \( N(w) \). Consider the language
\[ L_4 = \{ w \in \{0, 1\}^* \mid N(w) \equiv 0 \mod 7 \} \]
of integers (or at least their representations) that are multiples of 7. Some elements of \( L_4 \) are
\[ 0, 0000, 111, 1110, 10101, \ldots \]
Is \( L_4 \) context free?