# Week4 monday

Recap so far: In DFA, the only memory available is in the states. Automata can only "remember" finitely far in the past and finitely much information, because they can have only finitely many states. If a computation path of a DFA visits the same state more than once, the machine can't tell the difference between the first time and future times it visits this state. Thus, if a DFA accepts one long string, then it must accept (infinitely) many similar strings.

**Definition** A positive integer $p$ is a **pumping length** of a language $L$ over $\Sigma$ means that, for each string $s \in \Sigma^*$, if $|s| \geq p$ and $s \in L$, then there are strings $x, y, z$ such that

$$s = xyz$$

and

$$|y| > 0, \qquad \text{for each } i \geq 0, \ xy^i z \in L, \qquad \text{and} \qquad |xy| \leq p.$$

**Negation**: A positive integer $p$ is **not a pumping length** of a language $L$ over $\Sigma$ iff

$$\exists s \left( \ |s| \geq p \wedge s \in L \wedge \forall x \forall y \forall z \left( \ (s = xyz \wedge |y| > 0 \wedge |xy| \leq p \ ) \to \exists i (i \geq 0 \wedge xy^i z \notin L)) \ \right) \right.$$

*Informally:*

Restating **Pumping Lemma**: If $L$ is a regular language, then it has a pumping length.

**Contrapositive**: If $L$ has no pumping length, then it is nonregular.

The Pumping Lemma *cannot* be used to prove that a language *is* regular.

The Pumping Lemma **can** be used to prove that a language *is not* regular.

*Extra practice*: Exercise 1.49 in the book.

**Proof strategy**: To prove that a language $L$ is **not** regular,

- Consider an arbitrary positive integer $p$

- Prove that $p$ is not a pumping length for $L$

- Conclude that $L$ does not have *any* pumping length, and therefore it is not regular.

**Example**: $\Sigma = \{0, 1\}$, $L = \{0^n 1^n \mid n \geq 0\}$.

Fix $p$ an arbitrary positive integer. List strings that are in $L$ and have length greater than or equal to $p$:


Pick $s =$

Suppose $s = xyz$ with $|xy| \leq p$ and $|y| > 0$.

<br>

Then when $i =$                  , $xy^i z =$

**Example**: $\Sigma = \{0, 1\}$, $L = \{ww^{\mathcal{R}} \mid w \in \{0, 1\}^*\}$.

Fix $p$ an arbitrary positive integer. List strings that are in $L$ and have length greater than or equal to $p$:

Pick $s =$

Suppose $s = xyz$ with $|xy| \leq p$ and $|y| > 0$.

$$\boxed{\phantom{xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx}}$$

Then when $i =$            , $xy^i z =$

**Example**: $\Sigma = \{0, 1\}$, $L = \{0^j 1^k \mid j \geq k \geq 0\}$.

Fix $p$ an arbitrary positive integer. List strings that are in $L$ and have length greater than or equal to $p$:

Pick $s =$

Suppose $s = xyz$ with $|xy| \leq p$ and $|y| > 0$.

$$\boxed{\phantom{xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx}}$$

Then when $i =$            , $xy^i z =$

**Example**: $\Sigma = \{0, 1\}$, $L = \{0^n 1^m 0^n \mid m, n \geq 0\}$.

Fix $p$ an arbitrary positive integer. List strings that are in $L$ and have length greater than or equal to $p$:

Pick $s =$

Suppose $s = xyz$ with $|xy| \leq p$ and $|y| > 0$.

$$\boxed{\phantom{xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx}}$$

Then when $i =$            , $xy^i z =$

*Extra practice:*

| Language | $s \in L$ | $s \notin L$ | Is the language regular or nonregular? |
|---|---|---|---|
| $\{a^n b^n \mid 0 \le n \le 5\}$ | | | |
| $\{b^n a^n \mid n \ge 2\}$ | | | |
| $\{a^m b^n \mid 0 \le m \le n\}$ | | | |
| $\{a^m b^n \mid m \ge n+3, n \ge 0\}$ | | | |
| $\{b^m a^n \mid m \ge 1, n \ge 3\}$ | | | |
| $\{w \in \{a,b\}^* \mid w = w^{\mathcal{R}}\}$ | | | |
| $\{ww^{\mathcal{R}} \mid w \in \{a,b\}^*\}$ | | | |

# Week4 friday

**Definition** A **pushdown automaton** (PDA) is specified by a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$ where $Q$ is the finite set of states, $\Sigma$ is the input alphabet, $\Gamma$ is the stack alphabet,

$$\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \to \mathcal{P}(Q \times \Gamma_\varepsilon)$$

is the transition function, $q_0 \in Q$ is the start state, $F \subseteq Q$ is the set of accept states.

*Formal definition*
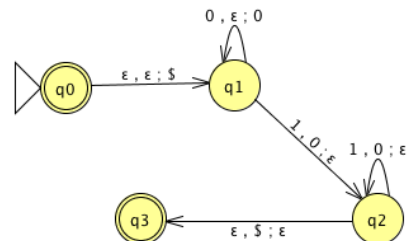


Draw the state diagram of a PDA with $\Sigma = \Gamma$.

Draw the state diagram of a PDA with $\Sigma \cap \Gamma = \emptyset$.

A PDA recognizing the set {                              } can be informally described as:

Read symbols from the input. As each 0 is read, push it onto the stack. As soon as 1s are seen, pop a 0 off the stack for each 1 read. If the stack becomes empty and there is exactly one 1 left to read, read that 1 and accept the input. If the stack becomes empty and there are either zero or more than one 1s left to read, or if the 1s are finished while the stack still contains 0s, or if any 0s appear in the input following 1s, reject the input.

State diagram for this PDA:

*Extra practice*: Consider the state diagram of a PDA with input alphabet $\Sigma$ and stack alphabet $\Gamma$.

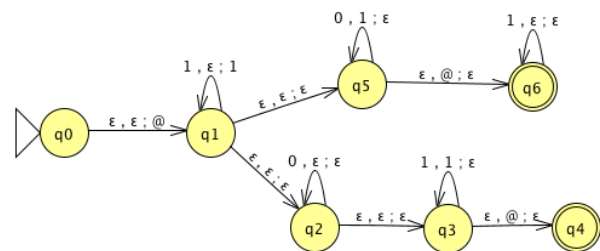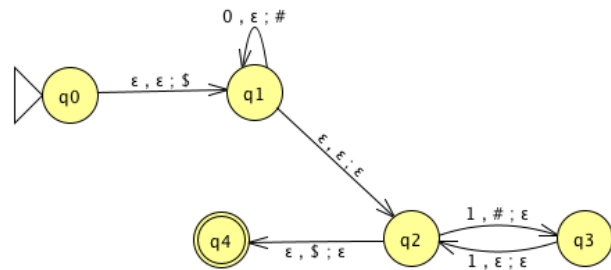| Label | means |
|---|---|
| $a, b; c$ when $a \in \Sigma$, $b \in \Gamma$, $c \in \Gamma$ | |
| $a, \varepsilon; c$ when $a \in \Sigma$, $c \in \Gamma$ | |
| $a, b; \varepsilon$ when $a \in \Sigma$, $b \in \Gamma$ | |
| $a, \varepsilon; \varepsilon$ when $a \in \Sigma$ | |

How does the meaning change if $a$ is replaced by $\varepsilon$?

*Note: alternate notation is to replace ; with $\rightarrow$*

For the PDA state diagrams below, $\Sigma = \{0, 1\}$.

| Mathematical description of language | State diagram of PDA recognizing language |
| --- | --- |





$\{0^i 1^j 0^k \mid i, j, k \geq 0\}$

# Week3 friday

**Theorem**: For an alphabet $\Sigma$, For each language $L$ over $\Sigma$,

$$L \text{ is recognized by some DFA}$$
$$\text{iff}$$
$$L \text{ is recognized by some NFA}$$
$$\text{iff}$$
$$L \text{ is described by some regular expression}$$

If (any, hence all) these conditions apply, $L$ is called **regular**.

**Prove or Disprove**: There is some alphabet $\Sigma$ for which there is some language recognized by an NFA but not by any DFA.

**Prove or Disprove**: There is some alphabet $\Sigma$ for which there is some finite language not described by any regular expression over $\Sigma$.

**Prove or Disprove**: If a language is recognized by an NFA then the complement of this language is not recognized by any DFA.

| Set | Cardinality |
|---|---|
| $\{0, 1\}$ | |
| $\{0, 1\}^*$ | |
| $\mathcal{P}(\{0, 1\})$ | |
| The set of all languages over $\{0, 1\}$ | |
| The set of all regular expressions over $\{0, 1\}$ | |
| The set of all regular languages over $\{0, 1\}$ | |

**Pumping Lemma** (Sipser Theorem 1.70): If $A$ is a regular language, then there is a number $p$ (a *pumping length*) where, if $s$ is any string in $A$ of length at least $p$, then $s$ may be divided into three pieces, $s = xyz$ such that

- $|y| > 0$

- for each $i \geq 0$, $xy^i z \in A$

- $|xy| \leq p$.

**True or False**: A pumping length for $A = \{0, 1\}^*$ is $p = 5$.

**True or False**: A pumping length for $A = \{1, 01, 001, 0001, 00001\}$ is $p = 4$.

**True or False**: A pumping length for $A = \{0^j 1 \mid j \geq 0\}$ is $p = 3$.

**True or False**: For any language $A$, if $p$ is a pumping length for $A$ and $p' > p$, then $p'$ is also a pumping length for $A$.