CSE 105 Discussion – Week 4

PUMPING LEMMA AND PDA

Do non-regular languages exist?

Yes! Why?

Q1: What does it mean for a language to be regular? (might have multiple right answers)

A. Finite
B. Can be recognized by an NFA/DFA
C. Can be described by a Regex

Q2: What is the cardinality of the set of all Regex over some alphabet?

A. Finite
B. Countable
C. Uncountable

Do non-regular languages exist?

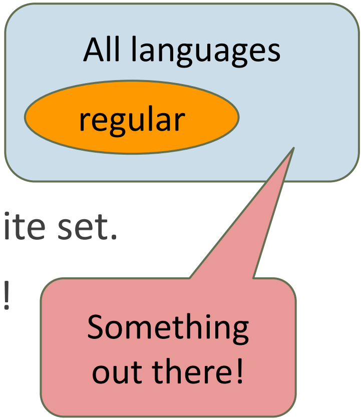
Yes! Why?

Q1: What does it mean for a language to be regular? (might have multiple right answers)

A. Finite
B. Can be recognized by an NFA/DFA
C. Can be described by a Regex

Q2: What is the cardinality of the set of all Regex over some alphabet?

A. Finite
B. Countable
C. Uncountable



Now, the set of all languages is uncountable since it is the powerset of an infinite set.

Also note that the set of languages NFA/DFA/Regex can describe are the same!

Intuitions about non-regular langs

- 0*1*
- {1^n 0^m | n, m > 1}
- {0, 00, 0000, 00000000, …}
- {0, 000, 00000, …}
- {0^n 1^m | n > m > 0}

Intuitions about non-regular langs

- 0*1* regular
- {1^n 0^m | n, m > 1} regular
- {0, 00, 0000, 00000000, …} non-regular
- {0, 000, 00000, …} regular
- {0^n 1^m | n > m > 0} non-regular

Pumping Lemma

If A is a regular language then there is a number p (the 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

- $|y| > 0$, and // The loop is nonempty
- For each $i \geq 0, xy^i z \in A$ // Pumping the loop any # of times creates other strings in A
- $|xy| \leq p$ // The loop appears in the first p characters

For regular languages we can set $p >$ # of states in DFA recognizing language

For finite languages we can set $p >$ length of longest string in language

Pumping Lemma T/F

If A is a regular language then there is a number p (the 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

- $|y| > 0$, and // The loop is nonempty
- For each $i \geq 0, xy^iz \in A$ // Pumping the loop any # of times creates other strings in A
- $|xy| \leq p$ // The loop appears in the first p characters

Are the following statements True or False?

- L is regular $\rightarrow$ L has a pumping length

Pumping Lemma T/F

If A is a regular language then there is a number p (the 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

- $|y| > 0$, and // The loop is nonempty
- For each $i \geq 0, xy^iz \in A$ // Pumping the loop any # of times creates other strings in A
- $|xy| \leq p$ // The loop appears in the first p characters

Are the following statements True or False?

- L is regular $\rightarrow$ L has a pumping length **True**, this is what the Pumping Lemma tells us
- L is not regular $\rightarrow$ L does not have a pumping length

Pumping Lemma T/F

If A is a regular language then there is a number p (the 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

- $|y| > 0$, and // The loop is nonempty
- For each $i \geq 0, xy^iz \in A$ // Pumping the loop any # of times creates other strings in A
- $|xy| \leq p$ // The loop appears in the first p characters

Are the following statements True or False?

- L is regular $\rightarrow$ L has a pumping length **True**, this is what the Pumping Lemma tells us
- L is not regular $\rightarrow$ L does not have a pumping length *False*, this is the converse of Pumping Lemma
- L has a pumping length $\rightarrow$ L is regular

Pumping Lemma T/F

If A is a regular language then there is a number p (the 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

- $|y| > 0$, and // The loop is nonempty
- For each $i \geq 0, xy^iz \in A$ // Pumping the loop any # of times creates other strings in A
- $|xy| \leq p$ // The loop appears in the first p characters

Are the following statements True or False?

- L is regular $\rightarrow$ L has a pumping length **True**, this is what the Pumping Lemma tells us
- L is not regular $\rightarrow$ L does not have a pumping length *False*, this is the converse of Pumping Lemma
- L has a pumping length $\rightarrow$ L is regular *False*, this is the inverse of Pumping Lemma
- L does not have a pumping length $\rightarrow$ L is not regular

Pumping Lemma T/F

If A is a regular language then there is a number p (the 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

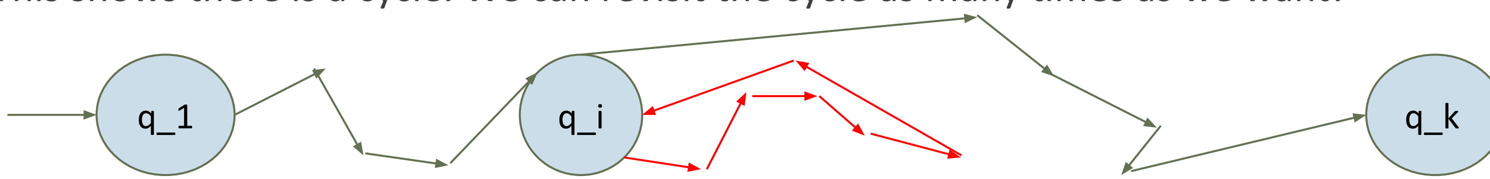
- $|y| > 0$, and // The loop is nonempty
- For each $i \geq 0, xy^iz \in A$ // Pumping the loop any # of times creates other strings in A
- $|xy| \leq p$ // The loop appears in the first p characters

Are the following statements True or False?

- L is regular $\rightarrow$ L has a pumping length **True**, this is what the Pumping Lemma tells us
- L is not regular $\rightarrow$ L does not have a pumping length *False*, this is the converse of Pumping Lemma
- L has a pumping length $\rightarrow$ L is regular *False*, this is the inverse of Pumping Lemma
- L does not have a pumping length $\rightarrow$ L is not regular **True**, this is the contrapositive of the Pumping Lemma
  - This is the statement we use to prove a language is not regular

Proof Sketch

- Suppose a language is regular, then it must have a DFA that recognizes it.
- DFA has finite amount of states, let's say k.
- Let s be a string of length $n \geq k$.
- Suppose s is accepted, that means after n transitions, we land in an accept state.
- Though the journey to accept state, we've visited n+1 states including the start.
- Now, $n+1 > k$, so at least one state has been visited twice.
- Let's say the we visited q_1, q_2 q_i, … q_i, … with q_i visited at least two times.
- This shows there is a cycle. We can revisit the cycle as many times as we want!



Pumping Lemma – Formal Logic

If A is a regular language then there is a number p (the 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

- $|y| > 0$, and // The loop is nonempty
- For each $i \geq 0, xy^iz \in A$ // Pumping the loop any # of times creates other strings in A
- $|xy| \leq p$ // The loop appears in the first p characters
- If A is a regular language then…
  - $\exists p\ (\forall s \in A\ |s| \geq p \rightarrow \exists x, y, z\ \left(s = xyz \wedge |y| > 0 \wedge |xy| \leq p \wedge \left(\forall i \in N\ xy^iz \in A\right)\right))$
- Contrapositive: Negate both sides and swap them
- If $\forall p(\exists s \in A\ |s| \geq p \wedge \forall x, y, z\ \left((s = xyz \wedge |y| > 0 \wedge |xy| \leq p) \rightarrow \left(\exists i \in N\ xy^iz \notin A\right)\right))$
  - …then A is a nonregular language
  - If we can show that all values of p are not pumping lengths for A then we have shown that A is nonregular

Pumping Lemma – Strategy

In proofs of nonregularity of language A using the pumping lemma our goal is to show

$$\forall p(\exists s \in A\ |s| \geq p \wedge \forall x, y, z\ \left((s = xyz \wedge |y| > 0 \wedge |xy| \leq p) \rightarrow \left(\exists i \in N\ xy^i z \notin A\right)\right))$$

- For any p

- There is a string s such that

- For any viable split of the string into x, y, and z

- We can choose some # of repetitions of y to get a string not in the language

Pumping Lemma – Strategy

In proofs of nonregularity of language A using the pumping lemma our goal is to show

$$\forall p(\exists s \in A \ |s| \geq p \wedge \forall x, y, z \left((s = xyz \wedge |y| > 0 \wedge |xy| \leq p) \rightarrow \left(\exists i \in N \ xy^i z \notin A\right)\right))$$

- For any p
  - Consider arbitrary p
- There is a string s such that
  - Choose a string s in terms of p (creative part)
- For any viable split of the string into x, y, and z
  - Define x, y, z according to PL conditions $|y| > 0 \wedge |xy| \leq p$
- We can choose some # of repetitions of y to get a string not in the language
  - Choose i such that $xy^i z$ is not in the language (other creative part)

Pumping Lemma – Example

- Consider the language $PAL = \{w \in \{0,1\}^* | w = w^R\}$, i.e. the set of all palindromes over $\{0,1\}$
- Show that PAL is nonregular using the pumping lemma
- WTS

$$\forall p(\exists s \in PAL\ |s| \geq p \wedge \forall x, y, z\left((s = xyz \wedge |y| > 0 \wedge |xy| \leq p) \rightarrow \left(\exists i \in N\ xy^i z \notin PAL\right)\right))$$

Consider arbitrary pumping length p. WTS there is a valid string in PAL that can't be pumped.

Which string should we choose?

A. 111000111

B. 10^p1

C. 0^p10^p

D. 0^p1^p

Pumping Lemma – Example

- Consider the language $PAL = \{w \in \{0,1\}^* | w = w^R\}$, i.e. the set of all palindromes over $\{0,1\}$
- WTS

$$\forall p(\exists s \in PAL\ |s| \geq p \wedge \forall x, y, z\ \left((s = xyz \wedge |y| > 0 \wedge |xy| \leq p) \rightarrow \left(\exists i \in N\ xy^i z \notin PAL\right)\right))$$

Consider arbitrary pumping length p. WTS there is a valid string in PAL that can't be pumped.

Consider string $s = 0^p 1 0^p \in PAL$, where $|s| > p$, as desired.

Let $s = xyz$ where $x = 0^k, y = 0^j, z = 0^l 1 0^p$ such that $j > 0,$ and $k + j + l = p$.

WTS there is a value i such that $xy^i z \notin PAL$

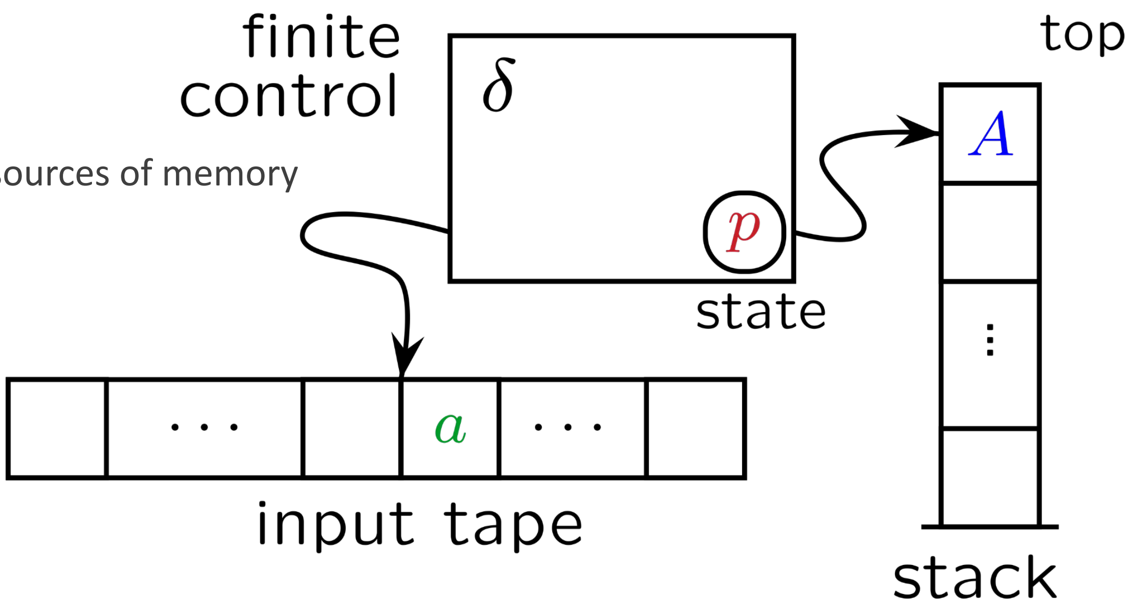
Consider $i = 0$. Then $xy^i z = xz = 0^k 0^l 1 0^p$. Since j > 0 then $k + l < k + j + l$.

Then $k + l < p$, so $0^k 0^l 1 0^p$ has an unequal number of leading and ending 0s, and therefore is not palindromic.

Therefore, $xy^0 z \notin PAL$ and p is not a pumping length for PAL. Thus PAL has no pumping length and is nonregular.

Pushdown Automata (PDA)

- What is the source of memory of an NFA?
  - The state it is in
  - That's it
- Now add stack
  - we now have two sources of memory



PDA Formal Description

DEFINITION 2.13

A ***pushdown automaton*** is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$, where Q, Σ, Γ, and F are all finite sets, and

1. Q is the set of states,
2. Σ is the input alphabet,
3. Γ is the stack alphabet,
4. $\delta \colon Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \longrightarrow \mathcal{P}(Q \times \Gamma_\varepsilon)$ is the transition function,
5. $q_0 \in Q$ is the start state, and
6. $F \subseteq Q$ is the set of accept states.

Compare And Contrast

DEFINITION 2.13

A ***pushdown automaton*** is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$, where Q, Σ, Γ, and F are all finite sets, and

1. Q is the set of states,
2. Σ is the input alphabet,
3. Γ is the stack alphabet,
4. $\delta\colon Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \longrightarrow \mathcal{P}(Q \times \Gamma_\varepsilon)$ is the transition function,
5. $q_0 \in Q$ is the start state, and
6. $F \subseteq Q$ is the set of accept states.

Non-deterministic!

$$\delta_{PDA}(state, char, pop) = \{(new\_state, push), \ldots\}$$

DEFINITION 1.37

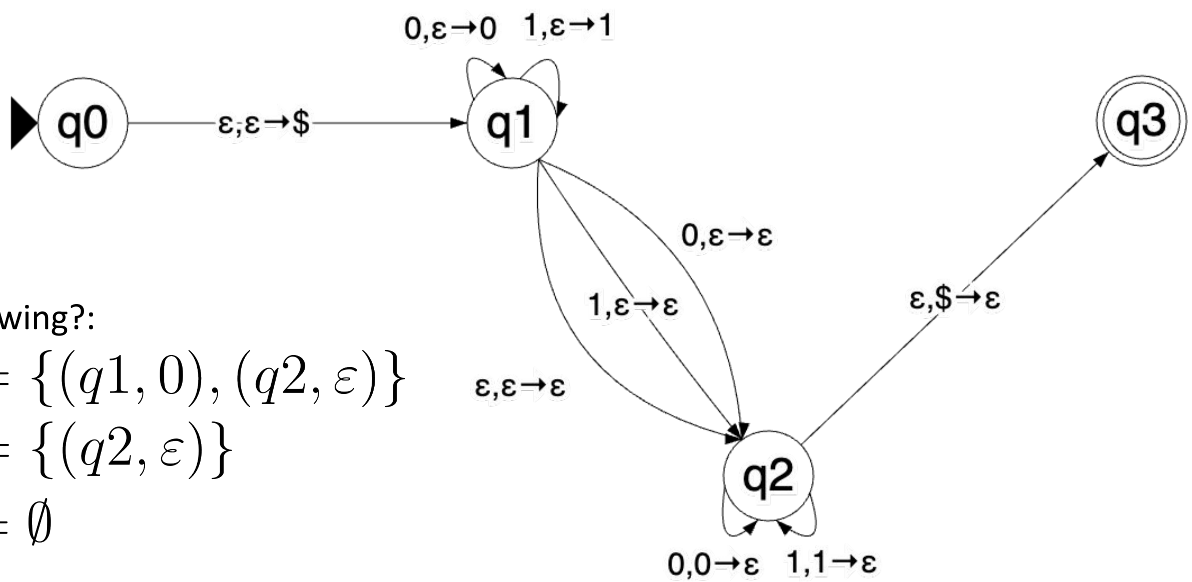
A ***nondeterministic finite automaton*** is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

1. Q is a finite set of states,
2. Σ is a finite alphabet,
3. $\delta\colon Q \times \Sigma_\varepsilon \longrightarrow \mathcal{P}(Q)$ is the transition function,
4. $q_0 \in Q$ is the start state, and
5. $F \subseteq Q$ is the set of accept states.

$$\delta_{NFA}(state, character) = \{new\_state, \ldots\}$$

PDA Transition Practice

flapjs link



What are the following?:

$$\delta(q1, 0, \varepsilon) = \{(q1, 0), (q2, \varepsilon)\}$$

$$\delta(q2, 1, 1) = \{(q2, \varepsilon)\}$$

$$\delta(q_2, 0, \epsilon) = \emptyset$$

Convert Languages to PDA

$$\{a_1b_1a_2b_2\ldots a_nb_n \mid count(a_1a_2\ldots a_n) = count(b_1b_2\ldots b_n) \;\wedge\; b_1b_2\ldots b_n \in L(0^*1^*)\}$$

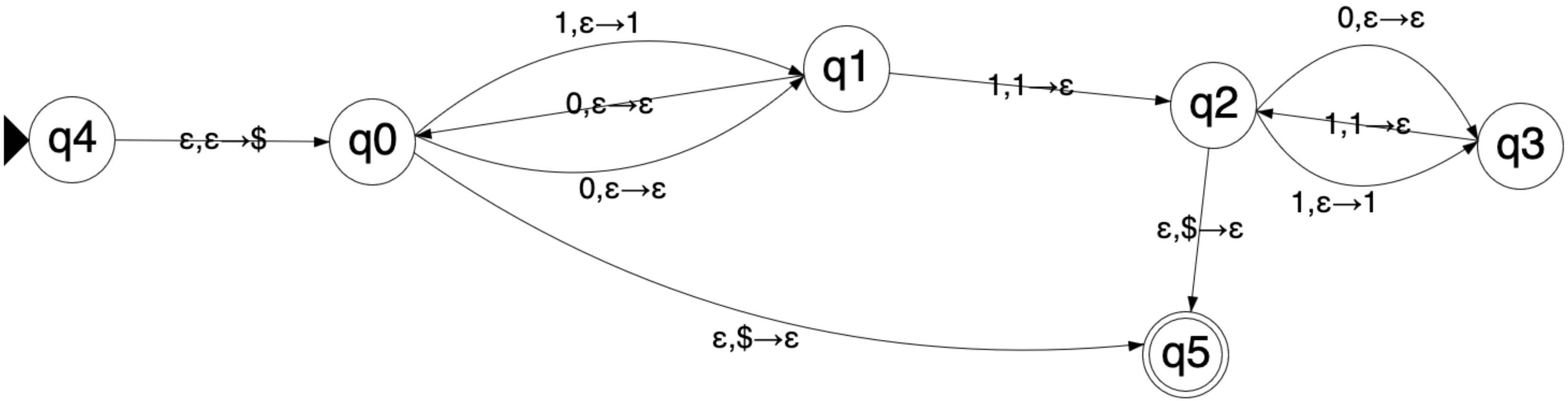
count(s) = number of 1s in s

- Need to keep track of 1s in even positions and make sure they match the number of 1s in odd positions
- All 1s in odd positions need to come after 0s
- Empty string is allowed

Sample machine

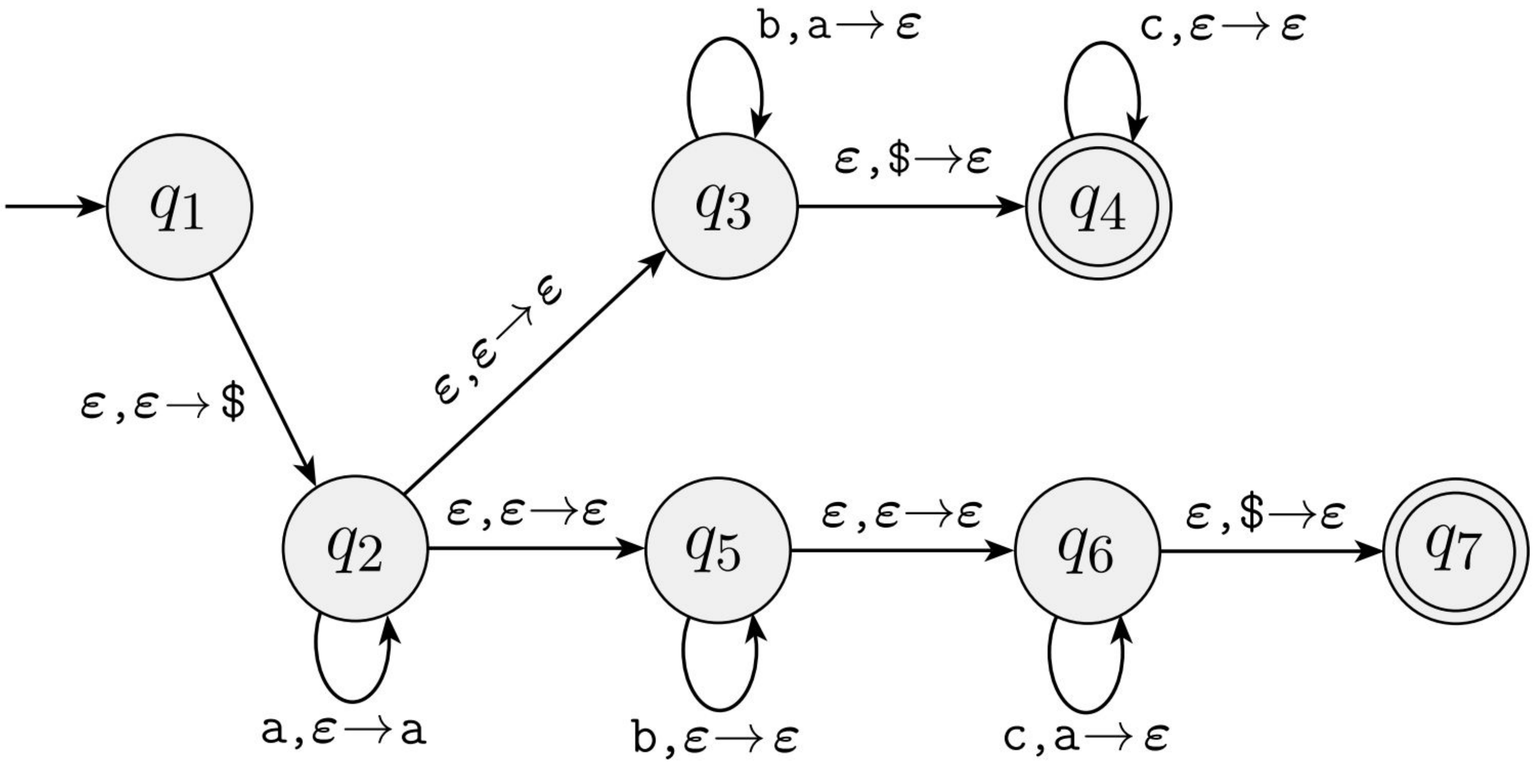
Convert Languages to PDA

$$\{a_1b_1a_2b_2\ldots a_nb_n \mid count(a_1a_2\ldots a_n) = count(b_1b_2\ldots b_n) \ \wedge \ b_1b_2\ldots b_n \in L(0^*1^*)\}$$

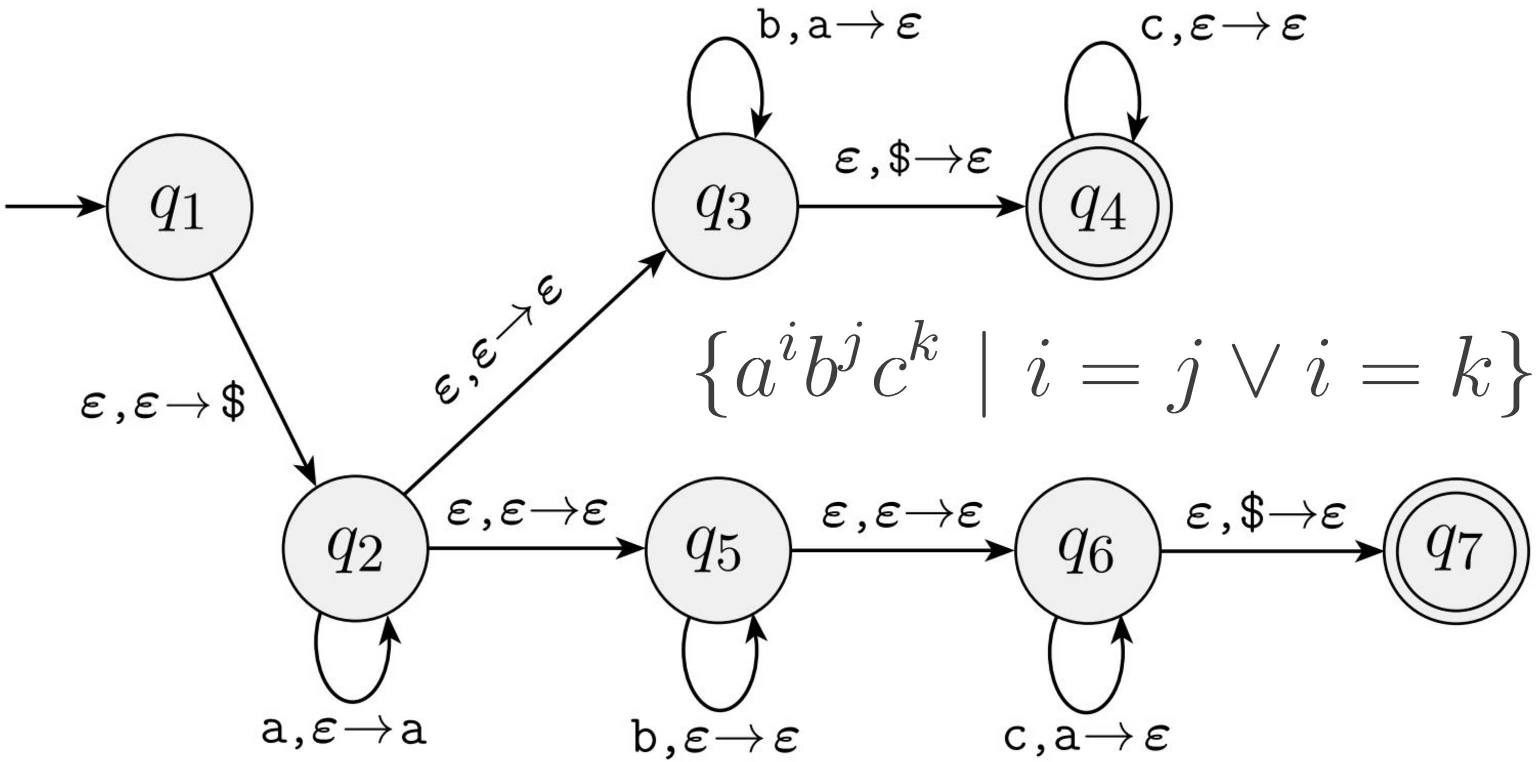


[flap.js link]

Decipher PDA Language



Decipher PDA Language



$$\{a^i b^j c^k \mid i = j \vee i = k\}$$