Pumping Lemma and PDA

CSE 105 Week 4 Discussion

Deadlines and Logistics

- Schedule your tests asap on PrairieTest !
- Do review quizzes on PrairieLearn
- HW3 due Thur 2/6/25 at 5pm (late submission open until 8am next morning)

Non Regular Languages & Pumping Lemma

Regular Languages Recap

A language is regular...

GNFA

- If and only if it is recognized by some DFA macro-states
- If and only if it is recognized by some NFA structural induction
- If and only if it is described by some regular expression

The three models are equally expressive, and we have algorithmic ways to translate from one model to another

Regular Languages Recap

We've seen in class that the class of regular languages is closed under

- Complementation ← flip accept / non-accept state status in DFA
- Union
- Intersection ← make 2 DFA compute "together" (HW2 Q4.2)
- Set-wise concatenation
- Kleene star

Non Regular Languages

We’ve seen in class that there exists non regular languages. How to prove that a language is non regular?

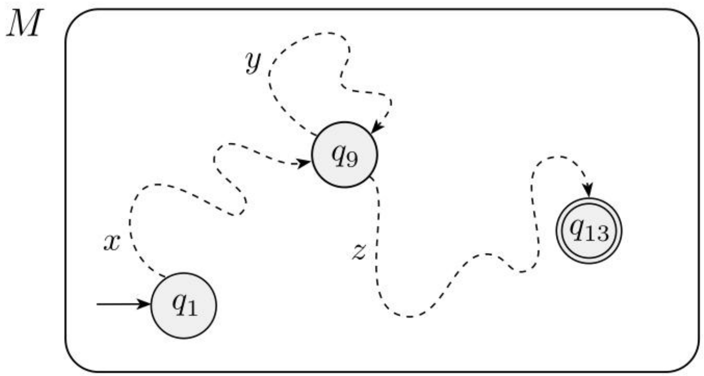
- No DFA / NFA can recognize it, no regular expression can describe it – universal statements
- Intead, we can look at an invariant property of all regular languages...

An Invariant Property of Regular Languages

- Observation: A DFA can only see so far in the past. How far ?
- Automata can only "remember"...
  - ...finitely far in the past
  - ...finitely much information
- If a computation path visits the same state more than once, the machine can't tell the difference between the first time and future times it visited that state.

$s = xyz$

"Long"!



xz
xyyz
xyyyz
⋮

Sipser Figure 1.72

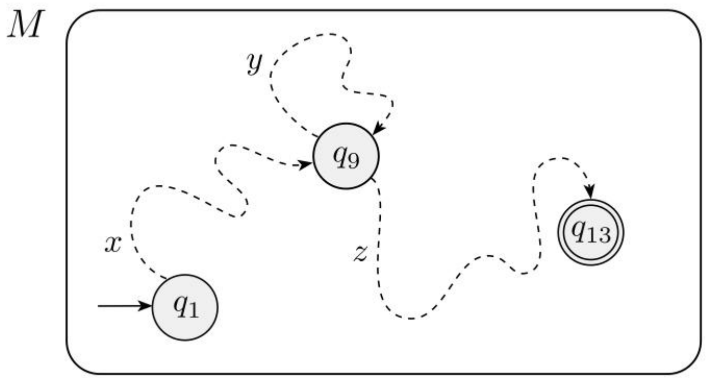
The Pumping Lemma

"long string"

THEOREM 1.70

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, $s = xyz$, satisfying the following conditions:

1. for each $i \geq 0$, $xy^iz \in A$,
2. $|y| > 0$, and → $y \neq \varepsilon$
3. $|xy| \leq p$.



Sipser Theorem 1.70, FIgure 1.72

The Pumping Lemma

THEOREM 1.70

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, $s = xyz$, satisfying the following conditions:

1. for each $i \geq 0$, $xy^iz \in A$,
2. $|y| > 0$, and
3. $|xy| \leq p$.

"vacuously true"

→ no such string!

$p = 7 >$ max len $= 6$

What happens when A is a finite language?

$A = \{00, 0101, 000110\}$

- ✓ Is A regular? yes (because there's a regex that describes it)
- ✓ Does A have a pumping length? If so, what can it be?

The Pumping Lemma

THEOREM 1.70 ....

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, $s = xyz$, satisfying the following conditions:

1. for each $i \geq 0$, $xy^iz \in A$,
2. $|y| > 0$, and
3. $|xy| \leq p$.

A positive integer P is the pumping length of a language L if :

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

Key points to note

Statement A : language *L* is regular
Statement B : language *L* has a pumping length P

Pumping lemma states : A→B, i.e. every regular language has a pumping length

Note that you cannot conclude that B→A ! i.e, just because a language has a pumping length P, it doesn't mean that it is regular !

In other words, pumping lemma cannot be used to prove that a language is regular.
What are the necessary and sufficient conditions for a language to be regular ?

DFA / NFA / regular expression

How to use pumping lemma ?

Recollect that $A \rightarrow B \equiv \neg B \rightarrow \neg A$ (CSE 20?) !*

What does this tell us ?

"If a language does NOT have a pumping length, then it is definitely not regular" !

*Contrapositive of an implication is equivalent to the implication itself

Strategy for proving non-regularity

To prove that a language L is not regular:

1. Consider arbitrary positive integer p
2. Prove that p isn't a pumping length for L (adhering to all conditions)
3. Conclude that L does not have any pumping length and is therefore not regular.

Strategy cont.

A positive integer P is the pumping length of a language L if :

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

The negation, “ A positive integer P is NOT the pumping length of a language L if : ”

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

Strategy cont.

A positive integer P is the pumping length of a language L if :

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

The negation, “ A positive integer P is NOT the pumping length of a language L if : ”

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

Although negating the 1st implication to get the 2nd is not part of CSE 105, I urge you to practice the negation !
Understanding first order predicate logic is a very useful skill to have !

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

For some P, ← w.t.s. P is not a pumping length of L

There is some “long” string *s* in the language *L* such that

For all “valid” splits of *s* into *x, y, z*

Repeating *y i* times, for some integer value *i* throws the resulting string out of the language.

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

For some P,

1 -Set P to be an arbitrary positive integer

There is some "long" string *s* in the language *L* such that

2 **-Choose s creatively** (critical step) such that $|s| \geq P \wedge s \in L$

For all "valid" splits of *s* into *x, y, z*

3 -Define x, y, z according to conditions $|y|>0 \wedge |xy| \leq P$ consider all

Repeating *y* *i* times, for some integer value *i* throws the resulting string out of the language.

4 -Choose i such that xy^iz is ejected from L

Prove that $L = \{a^m b^n \mid 0 \le m \le n\}$ is non-regular

① consider arbitrary postive integer P
w.t.s. P is not a pumping length for L

② pick $s = a^P b^P$ $\qquad |s| \ge P$ ✓ $\qquad s \in L$ ✓

$$\overbrace{aa\cdots a}^{P}\overbrace{bb\cdots b}^{P}$$

③ consider any $s = xyz$ where $|y| > 0$, $|xy| \le p$
i.e. $x = a^m$, $y = a^n \ (n > 0)$, $z = a^{p-m-n} b^p$

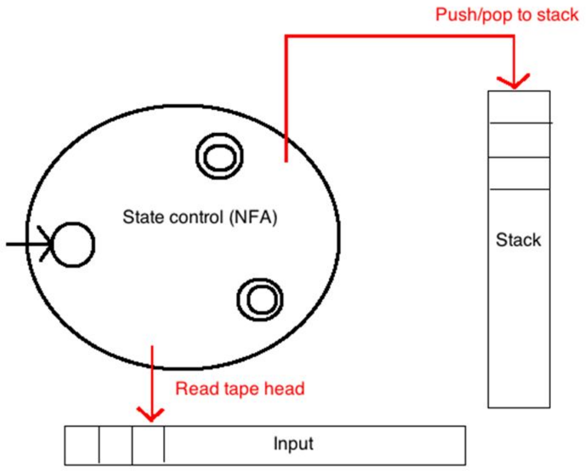
④ pick $i = 2$ s.t. $xy^iz \notin L$
$xyyz = a^m a^{2n} a^{p-m-n} b^p = a^{p+n} b^p \qquad n > 0 \Rightarrow p+n > p$

Language	$s \in L$	$s \notin L$	Is the language regular or nonregular?
$\{a^n b^n \mid 0 \leq n \leq 5\}$			Finite. Regular
$\{b^n a^n \mid n \geq 2\}$			
✓ $\{a^m b^n \mid 0 \leq m \leq n\}$			Nonregular
$\{a^m b^n \mid m \geq n+3, n \geq 0\}$			
$\{b^m a^n \mid m \geq 1, n \geq 3\}$			
$\{w \in \{a,b\}^* \mid w = w^{\mathcal{R}}\}$			
$\{ww^{\mathcal{R}} \mid w \in \{a,b\}^*\}$			

Push-Down Automata

Push-Down Automata

- NFA + Stack for (more) powerful computations



Witnessing acceptance

1. You read the entire input string
2. At least one of computation on the string ends in an accepting state
3. ~~The stack is empty~~

The stack contents do not directly determine the acceptance of the input string !

Edge label notation (when a, b, c are characters)

- Label a , b ; c or a , $b \longrightarrow c$ means
  - Read an a from the input
  - Pop b from the stack
  - Push c to the stack

- Label a , b ; c or a , $b \longrightarrow c$ means
  - Read an a from the input
  - Pop b from the stack
  - Push c to the stack

What edge label would indicate "Read a 0, don't pop anything from stack, don’t push anything to the stack"?

A. $0, \varepsilon \rightarrow \varepsilon$
B. $\varepsilon, 0 \rightarrow \varepsilon$
C. $\varepsilon, \varepsilon \rightarrow 0$
D. $\varepsilon \rightarrow \varepsilon, 0$
E. I don't know.

Review the formal definition of a PDA

DEFINITION 2.13

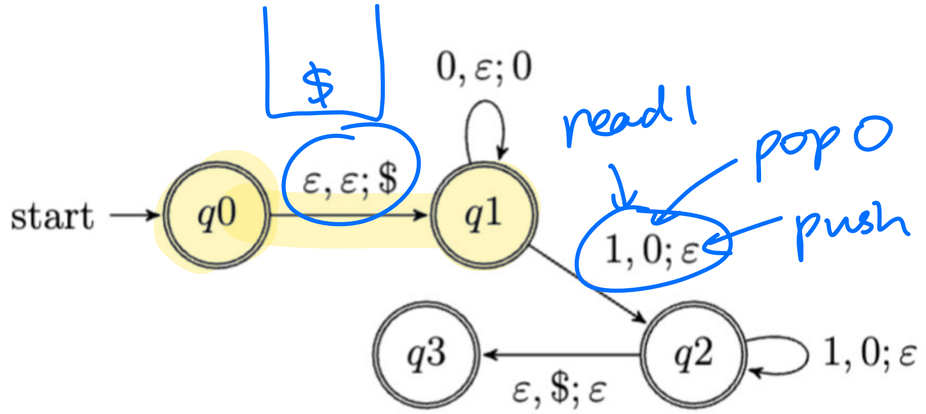
A ***pushdown automaton*** is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$, where Q, Σ, Γ, and F are all finite sets, and

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.

Sipser Definition 2.13

Review quiz

Consider the Pushdown Automaton (PDA) with input alphabet $\Sigma = \{0, 1\}$, stack alphabet $\Gamma = \{0, \$\}$ and state diagram:



read 1

pop 0

push nothing

Select all and only the strings below that are accepted by this PDA.

- [x] 01
- [ ] 111
- [x] 00
- [ ] 011

0
0
$