Regular and Nonregular Languages

Chapter 8

Languages: Regular or Not?

a*b* is regular.
$\{a^n b^n : n \geq 0\}$ is not.

$\{w \in \{a, b\}^* :$ every a is immediately followed by b$\}$ is regular.
$\{w \in \{a, b\}^* :$ every a has a matching b somewhere$\}$ is not

- Showing that a language is regular.
- Showing that a language is not regular.

How Many Regular Languages?

Theorem 8.1: There is a countably infinite number of regular languages.

Proof:

- Upper bound: number of FSMs (or regular exp.)
- Lower bound on number of regular languages:

{a},{aa},{aaa},{aaaa},{aaaaa},{aaaaaa},…

There are ***many* more nonregular** languages than there are regular ones.

Showing that a Language is Regular

- Every finite language is regular.

- $L = L_1 \cap L_2$, where:
  $L_1 = \{a^n b^n, n \geq 0\}$, and
  $L_2 = \{b^n a^n, n \geq 0\}$
  L =

- $L = \{w \in \{0 - 9\}^*$: w is the social security number of the current US president$\}$.

Showing That L is Regular

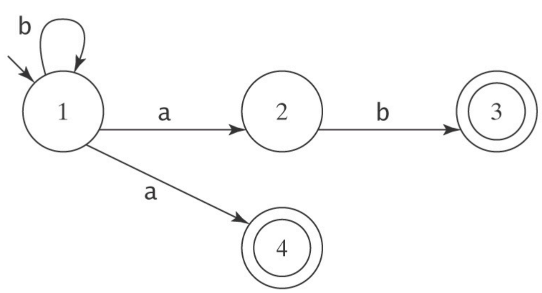
- Construct an FSM for L.
- Construct a regular grammar for L.
- Construct a regular expression for L.
- Show that the number of equivalence classes of $\approx_L$ is finite.
- Use closure theorems.

Closure Properties of Regular Languages

- Union
- Concatenation
- Kleene star
- Complement
- Intersection
- Difference
- Reverse

Closure of Regular Languages Under Complement (¬)

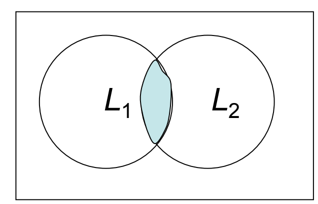
- Construct a DFSM for L
- Complete the DFSM
- Flip accepting states to get a DFSM for $\neg L$



Closure of Regular Languages Under Intersection

$L_1 \cap L_2 =$

$\neg(\neg L_1 \cup \neg L_2)$



Write this in terms of operations we have already proved closure for:

- Union
- Concatenation
- Kleene star
- Complementation

Closure of Regular Languages Under Difference

$L_1 - L_2 = L_1 \cap \neg L_2$

Use operations - Example

Let $L = \{w \in \{\texttt{a}, \texttt{b}\}^* : w$ contains an even number of `a`'s and an odd number of `b`'s and all `a`'s come in runs of three$\}$.

$L = L_1 \cap L_2$, where:

- $L_1 = \{w \in \{\texttt{a}, \texttt{b}\}^* : w$ contains an even number of `a`'s and an odd number of `b`'s$\}$, and
- $L_2 = \{w \in \{a, b\}^* :$ all `a`'s come in runs of three$\}$

Don't Try to Use Closure Backwards

One Closure Theorem:

If L_1 and L_2 are regular, then so is

$L = L_1 \cap L_2$

But if L is regular, what can we say about L_1 and L_2?

$L = L_1 \cap L_2$

Don't Try to Use Closure Backwards

One Closure Theorem:

If L_1 and L_2 are regular, then so is

$L = L_1 \cap L_2$

But if L is regular, what can we say about L_1 and L_2?

$L = L_1 \cap L_2$

$\texttt{ab} = \texttt{ab} \cap (\texttt{a} \cup \texttt{b})^*$ (they are regular)

Don't Try to Use Closure Backwards

One Closure Theorem:

If L_1 and L_2 are regular, then so is

$L = L_1 \cap L_2$

But if L is regular, what can we say about L_1 and L_2?

$L = L_1 \cap L_2$

$\mathtt{ab} = \mathtt{ab} \cap (\mathtt{a} \cup \mathtt{b})^*$ (they are regular)

$\mathtt{ab} = \mathtt{ab} \cap \{\mathtt{a}^n\mathtt{b}^n, n \geq 0\}$ (they may not be regular)

Showing that a Language is Not Regular

Every regular language can be accepted by some FSM.

It can only use a finite amount of memory to record essential properties.

Example:
$\{\mathtt{a}^n\mathtt{b}^n, n \geq 0\}$ is not regular

Showing that a Language is Not Regular

The only way to generate/accept an infinite language with a finite description is to use:

- Kleene star (in regular expressions), or
- cycles (in automata).

This forces some kind of simple repetitive cycle within the strings.

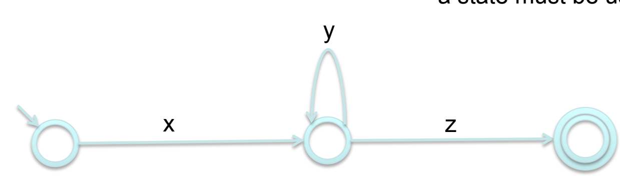
Fact: If a DFSM $M = (K, \Sigma, \delta, s, A)$ accepts a string of length $|K|$ or greater, then that string will force M to visit some state more than once (thus traversing at least one loop).

The Pumping Theorem for Regular Languages

Theorem 8.6 (Pumping Theorem) If L is regular, then there exists $k \geq 1$ such that any $w \in L$ with $|w| \geq k$ can be written as $w = xyz$, for some $x, y, z \in \Sigma^*$, such that

- $|xy| \leq k$,
- $y \neq \varepsilon$,
- $\forall q \geq 0, \; xy^qz \in L$

Proof: choose $k = |K|$;
a state must be used twice



Example

L= {a^nb^n: $n \geq 0$} is not regular

k is the number from the Pumping Theorem.

Choose w to be a^kb^k ("long enough").

```
              1                         2
 a a a a a … a a a a a b b b b b  … b b b b b b
             x      y                      z
```

Pumping Theorem implies $xy^qz \in L$, for any q.
But then we can pump more a's than b's, a contradiction.

Therefore, L is not regular.

Using the Pumping Theorem

If L is regular, then every "long" string in L is pumpable.

To show that L is not regular, we find one that isn't.

To use the Pumping Theorem to show that a language L is not regular, we must:

1. Choose a string w where $|w| \geq k$. Since we do not know what k is, we must state w in terms of k.
2. Consider all possibilities for y
3. In each case choose an q such that xy^qz is not in L.

Bal = {$w \in$ {), (}* :the parens are balanced}

PalEven = {ww^R : $w \in$ {a, b}*}

$\{a^n b^m : n > m\}$

$L = \{a^n : n \text{ is prime}\}$

$L = \{w = a^n : n \text{ is prime}\}$

Let $w = a^j$, where j = a prime number greater than $k+1$:

$$\underbrace{a\ a\ a\ a\ a\ a\ a}_{x}\ \underbrace{a\ a\ a}_{y}\ \underbrace{a\ a\ a}_{z}$$

$|x| + |z|$ is prime.
$|x| + |y| + |z|$ is prime.
$|x| + 2|y| + |z|$ is prime.
$|x| + 3|y| + |z|$ is prime, and so forth.

We have $xy^qz \in L$, for any q. Choose $q = |x| + |z|$. Then:

$$|x| + |z| + n|y| = |x| + |z| + (|x| + |z|)y$$
$$= (|x| + |z|)(1 + |y|)$$

But $(|x| + |z|)(1 + |y|)$ is NOT a prime, a contradiction.

Using the Pumping Theorem Effectively

- To choose w:
  - Choose a w that is in the part of L that makes it not regular.
  - Choose a w that is only barely in L.
  - Choose a w with as homogeneous as possible an initial region of length at least k.
- To choose q:
  - Try letting q be either 0 or 2.
  - If that doesn't work, analyze L to see if there is some other specific value that will work.

Using the Closure Properties

The two most useful ones are closure under:

- Intersection
- Complement

How to use? Assume a language L is regular and then:

- Show the intersection with a know regular language is NOT regular.
- Show that the complement is not regular.

Using the Closure Properties

$L = \{w \in \{\mathtt{a}, \mathtt{b}\}^*: \#_{\mathtt{a}}(w) = \#_{\mathtt{b}}(w)\}$

If L were regular, then:

$L' = L \cap$ _______

would also be regular. But it isn't.

$L = \{\mathtt{a}^i\mathtt{b}^j: i, j \geq 0 \text{ and } i \neq j\}$

Try to use the Pumping Theorem by letting $w = \mathtt{a}^{k+1}\mathtt{b}^k$:

$L = \{\mathtt{a}^i\mathtt{b}^j: i, j \geq 0 \text{ and } i \neq j\}$

Try to use the Pumping Theorem by letting $w = \mathtt{a}^k\mathtt{b}^{k+k!}$.

Then $y = \mathtt{a}^p$ for some nonzero p.

Let $q = (k!/p) + 1$ (i.e., pump in $(k!/p)$ times).

Note that $(k!/p)$ must be an integer because $p < k$.

The number of a's in the new string is $k + (k!/p)p = k + k!$.

So the new string is $\mathtt{a}^{k+k!}\mathtt{b}^{k+k!}$, which has equal numbers of a's and b's and so is not in L.

$L = \{\mathtt{a}^i\mathtt{b}^j: i, j \geq 0 \text{ and } i \neq j\}$

An easier way:

If L is regular then so is $\neg L$. Is it?

$L = \{a^i b^j: i, j \geq 0 \text{ and } i \neq j\}$

An easier way:

If L is regular then so is $\neg L$. Is it?

$\neg L = A^n B^n \cup \{\text{out of order}\}$

If $\neg L$ is regular, then so is $L' = \neg L \cap a^*b^*$

$= \_\_\_\_\_\_\_\_\_\_\_\_$

$L = \{a^i b^j c^k: i, j, k \geq 0 \text{ and } (i \neq 1 \text{ or } j = k)\}$

Every string in L of length at least 1 is pumpable.

- If $i = 0$: $y = \texttt{b}$ or $y = \texttt{c}$
- If $i = 1$ or $i > 2$: $y = \texttt{a}$
- If $i = 2$: $y = \texttt{aa}$

Pumping theorem cannot be used to prove that L is not regular.

$L = \{a^i b^j c^k: i, j, k \geq 0 \text{ and } (i \neq 1 \text{ or } j = k)\}$

But the closure theorems help. Suppose we guarantee that $i = 1$. If L is regular, then so is:

$L' = L \cap \texttt{ab}^*\texttt{c}^*$.

$L' = \{\texttt{ab}^j\texttt{c}^k : j, k \geq 0 \text{ and } j = k\}$

Use Pumping Theorem to show that L' is not regular: