COMS3261: Computer Science Theory

Spring 2013

Mihalis Yannakakis

Lecture 7, 9/25/13

Showing a language is not regular

- Method 1: Pumping lemma

  Note: There are some non-regular languages, for which we cannot show them by the pumping lemma

- Method 2: Use Closure properties

  to deduce that a language is not regular from the fact that some other language is known to be non-regular.

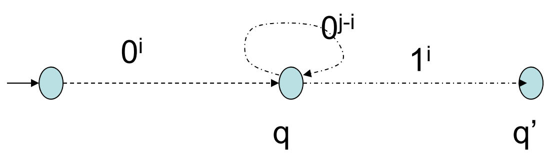
- Method 3: Myhill-Nerode theorem (not in the book)

  Necessary and sufficient condition for a language to be regular.

  Can be used for every non-regular language to show that it is not regular.

Pumping Lemma – motivating example

- Pumping Lemma: A method for showing that a language is not regular
- Example: Language $L = \{ 0^n 1^n \mid n \geq 1 \}$
- Suppose DFA A with n states accepts L
- On input 0 0 0 0 … , A goes through states q_0 (start state), q_1 q_2 …. Will repeat a state by the n-th step: $q_i = q_j = q$
- Now A has forgotten if it saw i or j 0’s and we can fool it.
- If input 1^i from q, goes to q’: either q’ is in F and A is wrong on 0^j1^i or q’ is not in F and A is wrong on 0^i1^i



Pumping Theorem

- Let L be a regular language. Then

$\exists$n (depends on L) such that

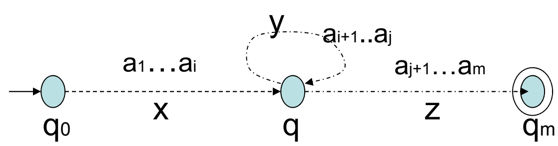
$\forall w \in L \,.\, |w| \geq n$

$\exists$ partition of w as w=xyz
(i.e. can pick strings x,y,z)
with $|xy| \leq n$, $y \neq \varepsilon$
such that

$\forall k \geq 0$, $xy^kz \in L$
i.e. $xz, xyz, xy^2z, \ldots \in L$

Proof

- Suppose L accepted by DFA A with n states.
- $w = a_1 a_2 \ldots a_m \in L$, $m \geq n$
- A on input w goes through states $q_0, \ldots, q_m$
- $m \geq n \Rightarrow$ at least two equal states, $q_i = q_j = q$



$x = a_1 \ldots a_i$

$y = a_{i+1} \ldots a_j$

$z = a_{j+1} \ldots a_m$

Clearly $xy^kz \in L$ for all $k \geq 0$

Pumping Theorem (contrapositive version)

- A language L is not regular if

$\forall\, n \geq 0$

$\exists\, w \in L\ .\ |w| \geq n$ (w depends on n)

$\forall$ partition of w as w=xyz
with $|xy| \leq n$, $y \neq \varepsilon$

$\exists\, k \geq 0$ such that $xy^kz \notin L$
i.e. one of $xz, xyz, xy^2z, \ldots \notin L$

Can use it to show that a language L is not regular.

Example

- $L = \{ w \in \{0,1\}^* \mid$ w has equal number of 0s and 1s $\}$
- Suppose L regular. Then $\exists n\ \forall w \in L\ .\ |w| \geq n \Rightarrow$ etc

Game: Adversary claims L regular; We'll refute it

- A: Picks any n
- We: Take $w = 0^n1^n$; it is in L, has length $>n \Rightarrow$
- A: $w = xyz$, such that $y \neq \varepsilon$ and $|xy| \leq n$
- We: $|xy| \leq n \Rightarrow$ xy is all 0's . Since $y \neq \varepsilon$, string xz has fewer 0 than 1's (missing the 0's of y) $\Rightarrow$ not in L, contradicting pumping theorem
- (Alternatively: xy^2z has more 0's than 1's, etc…)
- Conclusion: L is not regular

- HW: Show $\{0^i1^j \mid 0 \leq i \leq j\}$ and $\{0^i1^j \mid 0 \leq j \leq i\}$ not regular

Structure of the argument

- Adversary

1. Pick n

- We

2. Pick string w in L, $|w| \geq n$

3. Break up w into xyz s.t. $y \neq \varepsilon$ and $|xy| \leq n$

4. Pick a $k \geq 0$, and show $xy^kz \notin L$

- Have to argue that no matter what the adversary does in steps 1 and 3 (i.e. what n he picks, and how he breaks up the string w), we can succeed.

Use of Closure properties

- Can use closure properties to show that a language is regular, or to show that a language is not regular

- Example: We showed $M = \{ 0^n1^n \mid n \geq 0 \}$ is not regular
- Show N = set of binary strings with equal # of 0s, 1s not regular:
  Proof: $M = L(0^*1^*) \cap N$. If N was regular then M would also be regular, because of the closure under $\cap$.

Examples ctd.

- Show $M' = \{0^i1^j \mid i \neq j \}$ not regular:
  Proof: $L(0^*1^*) \cap M'^c = \{ 0^n1^n \mid n \geq 0 \} = M$.
  If M' was regular, so would M (by closure under c and $\cap$)
  -Impossible to show with pumping lemma! (adversary wins)

- Show $N' = \{ w \in \{0,1,2\}^* \mid$ w has equal # of 0s, 1s $\}$ not regular:
  Proof: $N=h(N')$ where h maps 0,1 to 0,1, and $h(2)=\varepsilon$

- Show $N'' = \{ w \in \{0,1,2\}^* \mid$ #of 0's = #1's+#2's$\}$ not regular
  Proof: $N=h(N'')$ where h maps 0,1 to 0,1, and $h(2)=1$
  Alternative proof: $N = N'' \cap \{0,1\}^*$

String Equivalence/Distinguishability

- String equivalence for a language L:

  Two strings x,y are distinguishable with respect to L if ∃ string z such that one of xz, yz is in L and the other is not; otherwise x,y are called equivalent, x≡y
- ≡ is an equivalence relation on strings
- Partition of strings into equivalence classes

- Index of language L = # of equivalence classes = max # of strings that are pairwise distinguishable (could be ∞)

Myhill – Nerode Theorem

- 1. A language L is regular iff L has finite index
- 2. Furthermore, index of L = min #states in a DFA for L

- In other words, L is not regular iff there is an infinite set S of strings that are pairwise distinguishable
- Proof of the (if) direction:
- Suppose the strings in set S={x_1,x_2,…} are pairwise distinguishable. For every two distinct strings x_i, x_j of S there is a z such that one of x_iz, x_jz is in L and one is not. The strings x_i, x_j must lead a DFA for L to distinct states, because otherwise their extensions x_iz, x_jz will lead the DFA to the same state, hence the DFA will accept both or reject both – which is wrong.
- Therefore the DFA must have at least |S| different states.

Finite index $\Rightarrow$ DFA

- In each equivalence class either all strings in L or all strings are not in L

  Proof: If $x \in L$, $y \notin L$ then ε distinguishes x,y
- If $x \equiv y$ then for every $a \in \Sigma$, $xa \equiv ya$

  Proof: If z distinguishes xa, ya then az distinguishes x,y

Construction of DFA

- States $\leftrightarrow$ Equivalence classes
- Initial state = $[\varepsilon]$ equivalence class of ε
- Accepting states: Equivalence classes with strings in L
- Transition function δ: For each class C and symbol $a \in \Sigma$, take arbitrary string $x \in C$ and define $\delta(C,a) = [xa]$

Example

- Show Language $L = \{ 0^n 1^n \mid n \geq 1 \}$ is not regular

- Consider the set of strings $S = \{ 0^n \mid n \geq 1 \}$
- Pairwise distinguishable: for all $i \neq j$, the strings 0^i, 0^j are distinguished by the string 1^i because $0^i 1^i \in L$, $0^j 1^i \notin L$

- S is infinite $\Rightarrow$ L is not regular

Example (of regular language)

- For binary string x, let <x> denote the number it represents
- L = { x∈{0,1}* | <x> is divisible by 3 (i.e. <x> mod3 = 0) }
- Three equivalence classes C_0,C_1,C_2 containing the strings x such that <x> mod 3 is respectively 0, 1, 2.
- Note: string x0 represents the number 2<x>, hence <x0>mod3 = 2<x>mod3 = 0,2,1 if x∈ C_0,C_1,C_2.
- String x1 represents the number 2<x>+1, hence <x1>mod3 = (2<x>+1)mod3 = 1,0,2 if x∈ C_0,C_1,C_2.