

Theory of Computation

Chapter 1

Pumping Lemma to Prove Non-regular Languages



School of Engineering | Computer Science
1

Linus Torvalds

- Started creation of Linux operating system core (1991); still coordinates new versions
- Linux is a free, open-source operating system used worldwide
- Since then, thousands have contributed
- In 2005 began development on Git, version control software



cu

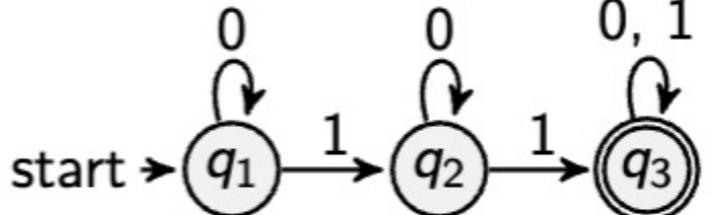
Proving a Languages is Not Regular

- Sometimes we would like to easily prove that a language is not part of the set of regular languages
- We can use the pumping lemma to prove a language is not part of this set
- Here is how it works:
 - If your parents gave you a bicycle and tell you it is a motorbike, how can you prove it is not a motorbike?
 - You must think of a feature that a motorbike has that a bicycle does not, e.g. a motor

Proving Languages to be Non-regular

- Fact: Every regular language has some property, P, such that if a language L does not have this property, then L is not regular
- Contrapositive: $(A \Rightarrow B) \rightarrow (\neg B \Rightarrow \neg A)$
 - If you have condition A, then condition B is true as well
 - If you do not have condition B, then you also do not have condition A
- What property P does every regular language have?

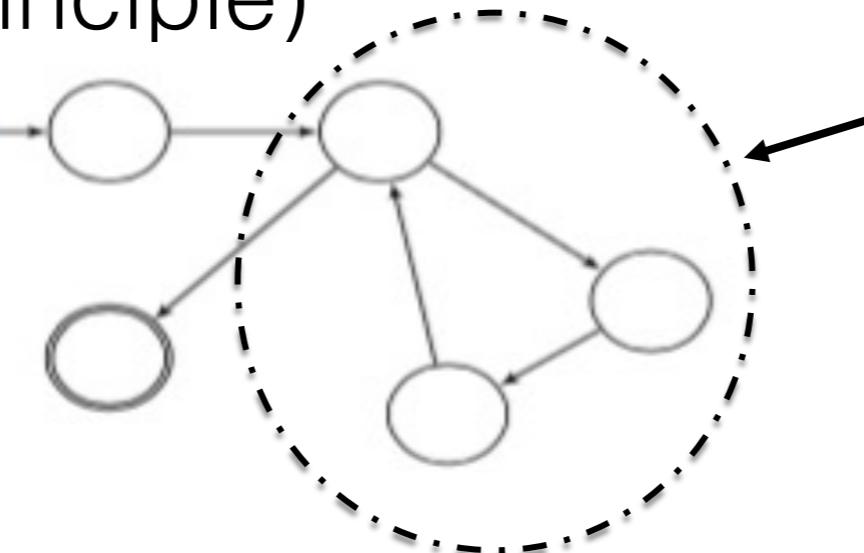
Property P – Pigeonhole Principle

- Observation: Let M be a DFA with k states. Let $w \in \Sigma^*$ be an input string of length $|w| \geq k$
- Conclusion 1: There exists a state q that has been visited at least twice when running M on w (pigeonhole principle)
 - Ex:
 $k = 3, w = 011, |w| = 3$
move through the DFA from $q_1 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3$, travelled to state q_1 two times
 - No matter what the input string is, you will always travel to one state at least twice with input the size of the number of states or larger if it is a regular language.

Property P – Pigeonhole Principle

- Observation: Let M be a DFA with k states. Let $w \in \Sigma^*$ be an input string of length $|w| \geq k$
- Conclusion 1: There exists a state q that has been visited at least twice when running M on w (pigeonhole principle)

- Generic:



Cycle C:
This is the set of states that can be repeated

- Pigeonhole – You have more input characters in your string than you have states to place them into

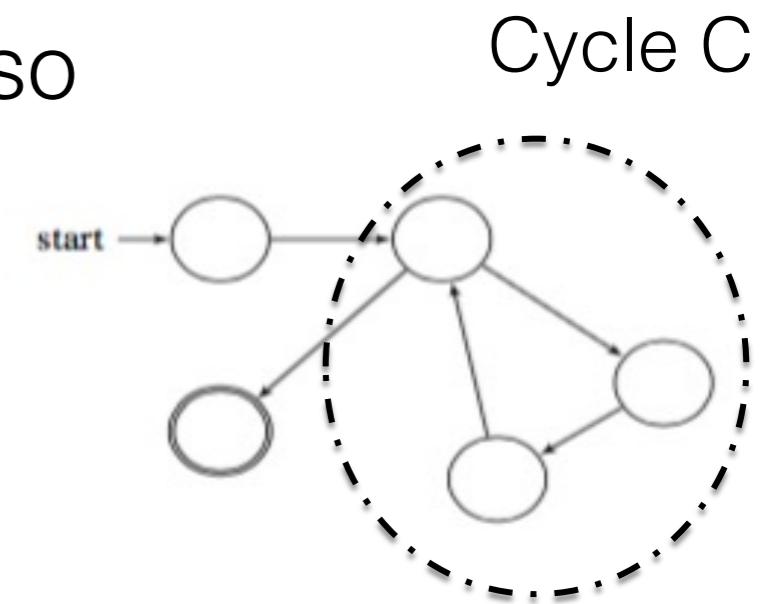
Property P – Pigeonhole Principle

- Observation: Let M be a DFA with k states. Let $w \in \Sigma^*$ be an input string of length $|w| \geq k$
- Conclusion 2: The substring of w read while following a cycle C can be repeated as many times as needed and the resulting string will still be accepted

- Ex: $w = \underline{\dots} \underline{\dots} \underline{\dots}$ is accepted, so

$\begin{array}{ccc} \underline{\dots} & \underline{\dots} & \underline{\dots} \\ | & | & | \\ x & y & z \end{array}$

$w' = \underline{\dots} (\underline{\dots})^* \underline{\dots}$ is accepted



Pumping Lemma

- Theorem 1.70: Let A be a regular language. There exists a “pumping length” $p > 0$, such that for all strings $s \in A$ of length $|s| \geq p$, there exists $x, y, z \in \Sigma^*$ such that $s = xyz$ and:
 1. For all $i \geq 0, xy^i z \in A$ means accepted by A
 2. $|y| > 0$ y can't be empty, but x, z can be
 3. $|xy| \leq p$ xy is the concatenation of x and y

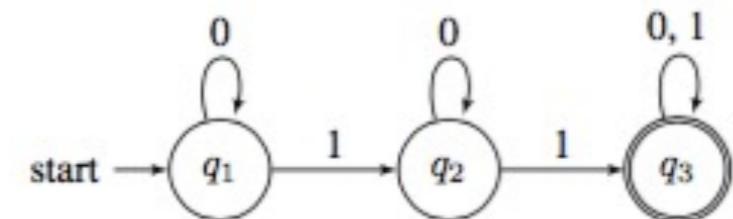
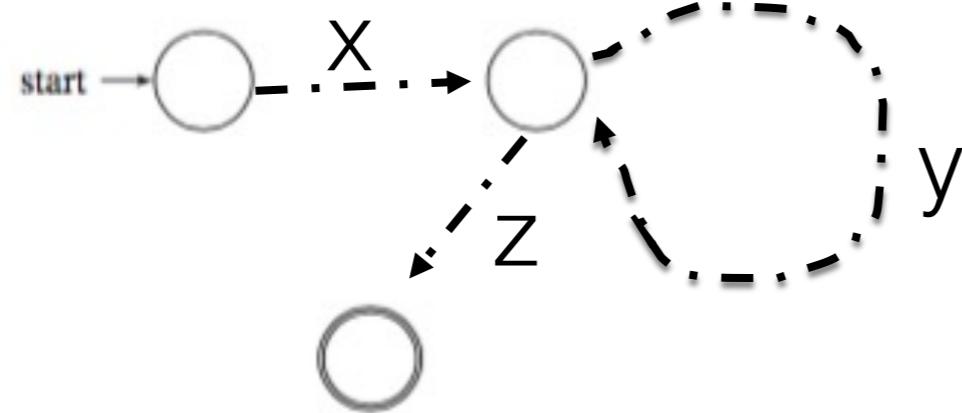
Pumping Lemma

- Theorem 1.70 Proof:
 - Let $M = (Q, \Sigma, \delta, q_1, F)$ be a DFA recognizing A with k states.
 - Let $s = s_1s_2\dots s_n \in A$ such that $|s| = n \geq k$, suppose M enters states r_1, r_2, \dots, r_{n+1} when processing s , by the pigeonhole principle, there exists $i < j$ such that $r_i = r_j$
 - Pick the smallest such j . Note that $j \leq k + 1$, since M has only k states and at least one state was repeated.

Pumping Lemma

- Theorem 1.70 Proof:

- Therefore, let $x = s_1s_2 \dots s_{i-1} | y = s_i \dots s_{j-1} | z = s_j \dots s_n$
- Let p (pumping length) = k (# of states)



Ex: $101 = q_1 \rightarrow q_2 \rightarrow q_2 \rightarrow q_3$

- Since x takes M from r_1 to r_i , y takes M from r_i to r_j , and z takes M from r_j to r_{n+1} , M accepts $xy^l z$ for all $l \geq 0$
- Since $i \neq j$, $|y| > 0$
- Since $j \leq k + 1$, $|xy| \leq k = p$

Pumping Lemma

- Example 1.73: Let $B = \{0^n 1^n \mid n \geq 0\}$
 - Claim: B is not regular
 - Proof by contradiction:

Pumping Lemma

- Example 1.73: Let $B = \{0^n 1^n \mid n \geq 0\}$
 - Claim: B is not regular
 - Proof by contradiction:
 - Assume B is regular, so there exists a pumping length $p > 0$, pick string $s = 0^p 1^p$
 - preconditions: $s \in B$ and $|s| = 2p \geq p$
 - There exists $s = xyz$ and by:
 3. $|xy| \leq p$ (x and y contain only 0's)
 2. $|y| > 0$ (y contains at least one 0)
 1. $xy^2z = xyyz = 0^{n'} 1^n$ where the result has more 0s than 1s, $n' > n$, which is a contradiction ($xyz \in B$, but $xy^2z \notin B$)

Or xyz , when $p = 2$ is $\begin{matrix} 0 & 0 & 1 & 1 \\ x & y & z \end{matrix}$, so when y^2 we get $\begin{matrix} 0 & 0 & 1 & 1 \\ x & y^2 & z \end{matrix} \notin B$

Pumping Lemma

- Example 1.75: Let $C = \{ww \mid w \in \{0, 1\}^*\}$
 - Claim: C is not regular
 - Proof by contradiction:

Pumping Lemma

- Example 1.75: Let $C = \{ww \mid w \in \{0, 1\}^*\}$
 - Claim: C is not regular
 - Proof by contradiction:
 - Assume C is regular, therefore there exists a pumping length $p > 0$, pick string $s = 0^p 1 0^p 1$
 - Preconditions: $s \in C$ and $|s| = 2p + 2 \geq p$
 - There exists $s = xyz$ and by:
 3. $|xy| \leq p$ (x and y contain only 0's)
 2. $|y| > 0$ (y contains at least one 0)
 1. $xy^2z = xyyz = 0^{p'} 1 0^p 1$, for $p' > p$, so $xyz \in C$, but $xy^2z \notin C$

Or xyz , $p = 2$ is $\begin{matrix} 0 & 0 & 1001 \\ x & y & z \end{matrix}$, so when y^2 we get $\begin{matrix} 0 & 00 & 1001 \\ x & y^2 & z \end{matrix} \notin C$

Pumping Lemma

- Example 1.77: Let $B = \{0^i 1^j \mid i > j\}$
 - Claim: B is not regular
 - Proof by contradiction:

Pumping Lemma

- Example 1.77: Let $B = \{0^i 1^j \mid i > j\}$
 - Claim: B is not regular
 - Proof by contradiction:
 - Assume B is regular, therefore there exists a pumping length $p > 0$, pick string $s = 0^{p+1}1^p$
 - Preconditions: $s \in B$ and $|s| = 2p + 1 \geq p$
 - There exists $s = xyz$ and by:
 3. $|xy| \leq p$ (x and y contain only 0's)
 2. $|y| > 0$ (y contains at least one 0)
 1. $xy^2z = xyyz = 0^{p+2}1^p$, which is still in the languageSo, use $xy^0z = xz = 0^{p'}1^p$, for $p' \leq p$, so $i > j$ and $xy^0z \notin B$

Remember that Case 1
should hold for all $i \geq 0$, $xy^iz \in B$

Try It

1. Use the Pumping Lemma to show that $D = \{0^n 1^m 0^n \mid m, n \geq 0\}$ is not a regular language

2. Construct an NFA from the Regular Expression
 $((00)^* 11) \cup 01$

Try It

- Use the Pumping Lemma to show that $D = \{0^n 1^m 0^n \mid m, n \geq 0\}$ is not a regular language
 - Claim: D is not regular
 - Proof by contradiction:
 - Assume D is regular, therefore there exists a pumping length $p > 0$, pick string $s = 0^p 1^{p+1} 0^p$
 - preconditions: $s \in B$ and $|s| = 3p + 1 \geq p$
 - There exists $s = xyz$ and by:
 3. $|xy| \leq p$ (x and y contain only 0's)
 2. $|y| > 0$ (y contains at least one 0)
 1. $xy^2z = xyyz = 0^{n'} 1^{n+1} 0^n$ where the result has more 0s on one side than the 0's on the other side, $n' > n$, which is a contradiction ($xyz \in B$, but $xy^2z \notin B$)

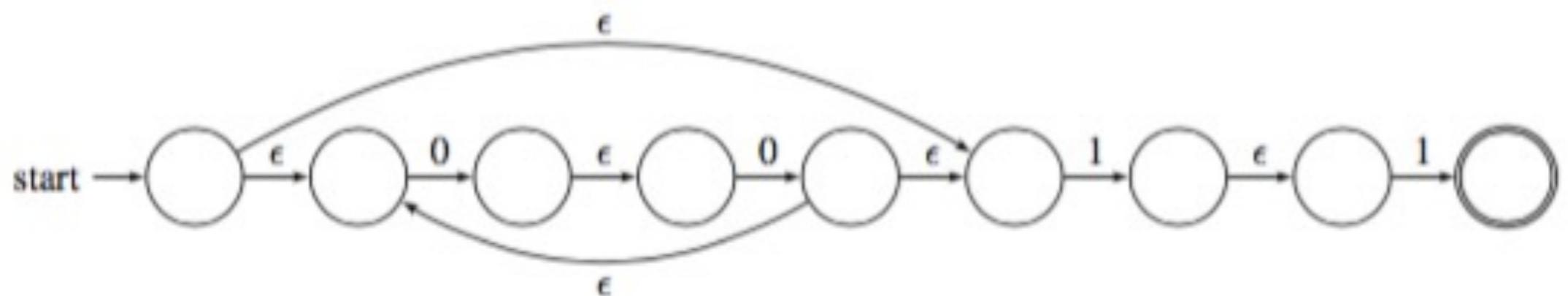
Or $xyz, p = 2$ is 0 0 11100, so when y^2 we get 0 00 11100 $\notin B$

x y z
18

x y² z
VCU
School of Engineering | Computer Science

Try It

1. Construct an NFA from the Regular Expression $((00)^*11) \cup 01$, cont.
 - $(00)^*11$



Try It

1. Construct an NFA from the Regular Expression $((00)^*11) \cup 01$, cont.
 - $((00)^*11) \cup 01$

