Proving non-regularity
Non-regular languages

• A finite state machine is finite. It has only some number \( n \) of states.

• If a language has strings in it where the memory requirements would appear to grow with the length, then it is likely not regular.

• Canonical example: \( \{0^k1^k \mid k \geq 0\} \), seems to require counting the 0s.

• This is not a proof.
Three methods for proving nonregularity

• The pumping lemma
  – If L is regular, any sufficiently long string in L can be “pumped” to obtain new strings in L. A language failing this test cannot be regular.

• Distinguishing suffixes
  – Show that there are infinitely many strings each of which would require a different state.

• Closure properties
  – Combine L with known regular languages using regularity-preserving operations, to obtain a known non-regular language.
1. The Pumping Approach

• Consider the language $L = \{0^k1^k \mid k \geq 0\}$
• Suppose there was a DFA $M$ with $n$ states for $L$
• Look at the accepting path for $0^n1^n$

By the pigeonhole principle, a state must be revisited on reading the first $n$ 0’s.

$0^n1^n$ is accepted...

but so is $0^{n+im}1^n$ for all $i$, so $M$ is incorrect

$0^{n-m}1^n$ is also incorrectly accepted – why?
1. The Pumping Lemma

- Suppose M is a DFA of $n$ states.
- Consider a string $w$ in $L(M)$, with length $\geq n$
- It visits $n+1$ states, so two must be the same:

$$w = xyz \in L(M)$$

$$w = x^{i}yz \in L(M) \text{ for all } i \geq 0$$

BUT ALSO
Recap...

If $L$ regular, then for all “sufficiently long” $w$ in $L$

$w$ can be “pumped” to obtain more strings in $L$:

$w = xyz$, $1 \leq |y| \leq n$, and $xy^iz$ is also in $L$ for all $i$

COROLLARY:

If for every $n$ there is a string $w$ in $L$ of length at least $n$ that “pumps” to a string not in $L$

... Then $L$ CANNOT BE REGULAR
\[ L = \{0^k1^k \mid k \geq 0\} \text{ is not regular} \]

(canonical non-regular language)

- Suppose there is an \( n \)-state DFA \( M \) recognizing \( L \).
- Then \( w = 0^n1^n \) is in \( L \), and en route to acceptance, there is a loop in \( M \) of size \( m \), with \( 1 \leq m \leq n \).
- So \( 0^{n+im}1^n \) is also accepted.
- But this string is not in \( L \) when \( i = 1 \).
- So \( M \) does not recognize \( L \).
$L = \{0^k \mid k \text{ is prime}\}$ is not regular

• Suppose there is an $n$-state DFA $M$ recognizing $L$.
• Let $p$ be a prime, $p > n$, so that $0^p$ is in $L$, and en route to acceptance, there is a loop in $M$ of size $m$, with $1 \leq m \leq n$.
• Then for each $i$, $0^{p+im}$ is also accepted for each $i$.
• But is $p+im$ prime for every $i$?
• Not if $i = p$.
• Since $M$ accepts $0^{p+pm} = 0^{p(1+m)}$, it doesn’t recognize $L$. 
**L = \{a^p b^q \mid p \geq q \} isn’t regular**

- Suppose there is an \( n \)-state DFA \( M \) recognizing \( L \).
- Then \( a^{n+1}b^n \) in \( L \), and en route to acceptance there is a loop in \( M \) of size \( m \), with \( 1 \leq m \leq n \).
- So \( a^{n+1+im}b^n \) is also accepted
- And this results in no contradiction
- So, is \( L \) regular since the string can be pumped?
- NO! Cannot show regularity by pumpability
• NO, \( L = \{a^p b^q \mid p \geq q \} \) is not regular.
• PL says all sufficiently long strings must be pumpable.
• We only need demonstrate ONE which ISN’T.
• \( a^{n+1} b^n \) was a bad choice; it could be pumped.

What would be a good choice?

Let’s try \( a^n b^n \)
$L = \{ a^p b^q \mid p \geq q \}$ is not regular

- Suppose there is an $n$-state DFA $M$ recognizing $L$.
- Then $a^n b^n$ in $L$, and en route to acceptance there is a loop in $M$ of size $m$, with $1 \leq m \leq n$.
- So $a^{n+im} b^n$ is also accepted.
- Since $n+im \geq n$, $a^{n+im} b^n$ is still in $L$ for each $i$.
- ???
\[ L = \{ a^p b^q \mid p \geq q \} \text{ is not regular} \]

- Recall that the loop can also be *eliminated* to yield a new string that is accepted.
- Suppose there is an \( n \)-state DFA \( M \) recognizing \( L \).
- Then \( a^n b^n \) in \( L \), and en route to acceptance there is a loop in \( M \) of size \( m \), with \( 1 \leq m \leq n \).
- So \( a^{n-m} b^n \) is also accepted.
- Since \( n-m < n \), \( a^{n-m} b^n \) is not in \( L \).
- So \( M \) does not recognize \( L \).
Summary

To show a language not regular using P.L

- Assume there is a DFA $M$ with $n$ states recognizing $L$
- Carefully select a string $w$ in $L$ of length $\geq n$.
- This string forces a loop in $M$ en route to acceptance of size $m$, with $1 \leq m \leq n$.
- Show that no matter what characters are on the loop, by “pumping” we can obtain a string not in $L$ that $M$ accepts.
- Conclude that $M$ doesn’t recognize $L$ after all
Class Exercise

Use the pumping lemma to show that the language $L = \{0^s \mid s \text{ is a perfect square}\}$ is not regular.
2. Distinguishing Suffixes

If $L$ is any language, and $x$, $y$, $z$ strings, we say that $z$ distinguishes $x$ from $y$ (with respect to $L$) if exactly one of $xz$ and $yz$ is in $L$. 
Distinguishing Suffixes

Examples

• If \( L = \{w \mid w \text{ contains an odd number of 1s and an odd number of 0s}\} \)
  – the suffix \( z = 01 \) distinguishes 00 from 100 since 0001 is in \( L \), but 10001 is not in \( L \).

• If \( L = \{0^n1^{2n} \mid n \geq 0\} \)
  – the suffix \( z = 1111 \) distinguishes 00 from 000
  – the suffix _____ distinguishes 01 from 001
  – the suffix _____ distinguishes 01 from 00111
Distinguishability

• Two strings $x$ and $y$ are distinguishable $L$ if there exists a distinguishing suffix $z$ for them.
• Otherwise, $x$ and $y$ are indistinguishable $L$
• Homework:
  
  indistinguishability is an equivalence relation.

• If $x$ and $y$ are distinguishable, then in any DFA for $L$, $x$ and $y$ must lead to different states
• Why? <one sentence proof>
Proving L Nonregular

• Describe an infinite set of strings $D$ such that any distinct pair $x$ and $y$ in $D$ are distinguishable
• (Thus, they all must go to different states – and there are infinitely many of them.)
• Example: $L = \{0^n1^n \mid n \geq 0\}$, $D = \{0^k \mid k \geq 0\}$. $D$ is infinite. And, the suffix $1^k$ distinguishes $0^k$ from $0^j$ because $0^k1^k$ is in $L$ but $0^j1^k$ is not.
• Thus, $L = \{0^n1^n \mid n \geq 0\}$ would need infinitely many states, and so is not regular.
Proving $L$ Nonregular

• Example: $L = \{a^p b^q \mid p \geq q \}$, $D = \{a^k \mid k \geq 0\}$
  – What suffix distinguishes $a^m$ from $a^n$?

• Example: $L = \{0^p 1^q \mid p \neq q \}$
  – Is $0^5 1^7$ distinguishable from $0^5 1^8$?
  – Is $0^7 1^5$ distinguishable from $0^8 1^6$?
  – What infinite set $D$ of strings has elements that are pairwise distinguishable?
Challenge

Let \( L = \{ w \mid \#_0(w) = \#_1(w) \} \).

Find an infinite set \( D \) of pairwise-distinguishable strings.
FYI

• Indistinguishability$_L$ is an equivalence relation
• The smallest DFA for $L$ has states that correspond to the equivalence classes
• Homework problem (may) hint at this
• See Myhill-Nerode theorem

http://en.wikipedia.org/wiki/Myhill–Nerode_theorem
3. Proving Nonregularity with Closure

Let \( L_1, L_2, \ldots, L_n \) be languages where \( L_1, L_2, \ldots, L_{n-1} \) are known regular languages, and \( L_n \) is an unknown language. Apply closure properties to these languages.

**THEN:** \( L_n \) must be non-regular.
\{w: \#0(w) = \#1(w)\} is not regular

• Let \( L = \{w \mid \#0(w) = \#1(w)\} \). Suppose \( L \) regular.
• Let \( L' = L \cap 0^*1^* \)
• \( L' \) is regular, since it is intersection of two regular languages.
• But \( L' = \{0^n1^n \mid n \geq 0\} \) which is NOT regular.
• Contradiction. Thus \( L \) is not regular.
Can be more complicated

• Show \( L = \{a^nba^{2n} : n \geq 0\} \) is not regular.
• Define \( h(0) = h(1) = a, h(2) = b \) and then
  \[ L' = h^{-1}(L) = \{(0+1)^n2(0+1)^{2n} : n \geq 0\} \]
• Intersect this with \( 0^*21^* \) gives
  \[ L'' = \{0^n21^{2n} : n \geq 0\} \]
• Define \( g(0)=0, g(1) = 11, g(2) = 2 \) and then
  \[ L''' = g^{-1}(L'') = \{0^n21^n : n \geq 0\} \]
• Define \( f(0)=0, f(1)=1, f(2)=\varepsilon \) and then
  \[ L'''' = f(L''') = \{0^n1^n : n \geq 0\} \) which is known nonregular
Summary: three methods for proving nonregularity

• The pumping lemma
  – If L is regular, any sufficiently long string in L can be “pumped” to obtain new strings in L. A language failing this test cannot be regular.

• Distinguishing suffixes
  – Show that there are infinitely many pairwise indistinguishable states

• Closure properties
  – Combine L with known regular languages using regularity-preserving operations, to obtain a known non-regular language.
Regular Languages Summary

• Regular languages: simple base cases, then closure under union, concat, and *, and many other ops
• Regular expressions capture regular languages
• Finite state machines, (det, nondet) recognize regular languages.
• Notion of state as memory: finite
• Nonregular.... where more than finite memory is required.
• Pumping, distinguishing suffixes, closure properties to prove nonregularity.