## Size of DFA's--Can we cut down?

*Maybe not!*

**n-1 (fixed no.)**

**Ex. L = ( {0,1})* 0 {0,1} ...{0,1} )**

*Set of strings over {0,1} with a 0 n characters to the right*

**There is no DFA with < $2^n$ states accepting L**

*Proof idea:*

*Any DFA for L needs to keep track of last n characters so far OW, it will give wrong answer*

*Since FA have no memory, and there are $2^n$ different strings of length n, the DFA needs $2^n$ different states.*

***In practice, want the DFA that program is simulating to be as small as possible.***

1

## Minimizing the number of states in DFA

**Th. Given a DFA M' recognizing L, there is a unique DFA M that accepts L, such that M has the minimal number of states among all DFA accepting L.**

*Proof idea: Group states, but split state from group if edge with label goes to* ***different*** *group (on full DFA)*

*Initial partition* ***Non-final*** ***Final***

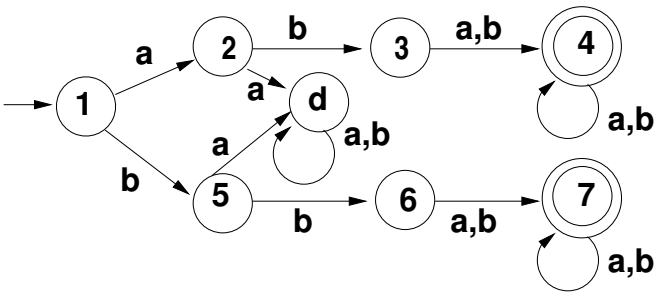
*1 3 4* | *2*

***New partition*** *1 → 3* ⇄ *3* | *4* → | *2*

*Continue till no further splitting is needed*

2

## Example Minimizing DFA




| Non-final | | | Final |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| d | 5 | 6 | 7 |

**Minimal DFA has 5 states**

3

## Minimizing the number of states in DFA, Part II

*Note: final states never grouped with non-final states*

*Could wind up with single-state groups (M′ minimal!)*

**When finish partitioning:**

**Make a new state for each group**

**Put in "summary" edges between groups**

**Start state: group with original start state**

**Final states: any group with an original final state**

**Throw out any new states not reachable from start**

*May still be large!*

4

## Limits of FA

**Can FA recognize all "computable" languages?**

**NO!  Ex. $\{ 0^n1^n \mid n \geq 0\}$ is not regular**

*Intuitively, would have to keep track of no. of 0's so far, then check have same no. of 1's. These numbers are not limited, and so cannot build it into FA states.*

*Need new technique in order to prove!*

**Pumping Lemma**

*Intuitive idea: Consider a string accepted by a FA.*

*If the string is very long, then the states we go through must repeat, since the FA only has a fixed number of states.*

*Then we can take a snippet out of the string, and still accept.*

*Or, we can repeat the snippet any no. of times, and accept.*



## Pumping Lemma

**If A is a regular language, then there is a no. p** *(pumping length)* **where, if s is any string in A of length at least p, s may be divided into three pieces x,y,z, s = xyz, such that all of the following hold:**

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

*Condition 1 lets us "pump out" elements in A*

*Note that x or z can be $\varepsilon$ , but y cannot be (by 2)*

*(Without 2, the lemma is trivially true, with $y = \varepsilon$ )*

*Condition 3 assures us we can make x and y small, if needed.*

## Pumping Lemma Example

L = $\{ 0^n 1^n \mid n \geq 0 \}$ is not regular.

Suppose L were regular. Then let p be the pumping length given by the pumping lemma.

Let s = $0^p 1^p$ in L. Note that |s| > p, so s = xyz with

1. for each i $\geq$ 0, $xy^i z$ is in L
2. |y| > 0
3. |xy| $\leq$ p

*Case 1:* y = 0 ... 0

Then xyyz will have more 0's than 1's,

so it cannot be in L, a contradiction.



## Pumping Lemma Example

*Case 2:* y = 1 ... 1

Then xyyz will have more 1's than 0's,

so it cannot be in L, a contradiction.

*Case 3:* y = 0 ... 0 1 ... 1

Then xyyz will have 0's and 1's out of order, with some 1's before 0's, a contradiction.

Since these are all possible cases, we can conclude that L is not regular.

## Proving Languages Non-Regular

1. Assume L is regular. Then the Pumping Lemma holds.
2. Let p be the pumping length for L given by the lemma.
3. Choose a string s in L of length $\geq$ p.
   *(Note that not every string in L will work!)*
4. Consider all cases s can be divided into x, y, z, s = xyz , satisfying conditions 2 and 3.
   Show for each case that there is an $i \geq 0$ with $xy^iz$ not in L.
5. This provides a contradiction to the assumption of the pumping lemma, and to L being regular.



## Examples: Showing Language L Non-Regular

L = {w | w has an equal no of 0 and 1's} *(1.39 p.80)*

1. Assume L is regular. Then the Pumping Lemma holds.
2. Let p be the pumping length for L given by the lemma.
3. We choose s= $0^p1^p$ (in L of length $\geq$ p).
4. Consider all cases s can be divided into x, y, z, s = xyz , satisfying conditions 2 (|y| > 0) and condition 3 ( $|xy| \leq p$).
   For this s, it must be that $y = 0^k$; this is the only case. We choose i=2: $xy^2z$ will have more 0's than 1's, and so cannot be in L.
5. Contradiction.

## Examples: Showing Language L Non-Regular

L = {ww | w in {0, 1} * } *(1.40 p.81)*

1. Assume L is regular. Then the Pumping Lemma holds.
2. Let p be the pumping length for L given by the lemma.
3. We choose s= $0^p 10^p 1$ (in L of length $\geq$ p).
4. Consider all cases s can be divided into x, y, z, s = xyz , satisfying conditions 2 (|y| > 0) and condition 3 ( |xy| $\leq$ p).
   For this s, it must be that y = $0^k$; this is the only case. We choose i=2: $xy^2z$ will have more 0's before the first 1 than the second 1; not in L.
5. Contradiction.



## Examples: Showing Language L Non-Regular

L = {$0^i 1^j$| i > j} *(1.42 p.82)*

1. Assume L is regular. Then the Pumping Lemma holds.
2. Let p be the pumping length for L given by the lemma.
3. We choose s= $0^{p+1} 1^p$ (in L of length $\geq$ p).
4. Consider all cases s can be divided into x, y, z, s = xyz , satisfying conditions 2 (|y| > 0) and condition 3 ( |xy| $\leq$ p).
   For this s, it must be that y = $0^k$; this is the only case. We choose i=0: xz does not have more 0's than 1's, and so cannot be in L. *(i = 2?)*
5. Contradiction.