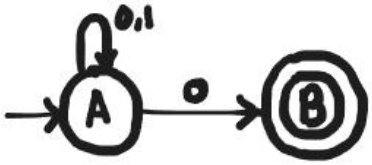
DFA vs. NFA

DFA	NFA
- Given the current state, we know what the next state will be - Has only one unique state - Has no choices or randomness - Easy and simple to design	- Given the current state, there could be multiple next states - Next state may be chosen at random - All the next states may be chosen in parallel

NFA

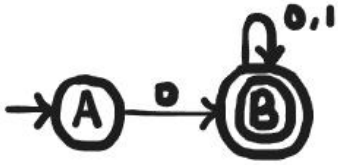
L = { Set of all strings that end with 0 }



If there's any way that ends in any set of states out of which at least one state is a final state, then it's a valid NFA

L = { Set of all strings that start with 0 }

= { 0, 00, 01, 000, ... }



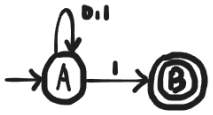
Construct a NFA that accepts sets of all strings over {0,1} of length 2

$\Sigma = \{ 0, 1 \}$

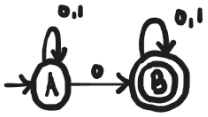
L = { 00, 01, 11, 10 }

NFA

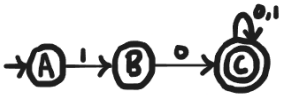
Ex 1) L1 = { Set of all strings that ends with '1' }



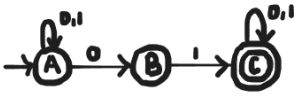
Ex 2) L2 = { Set of all strings that contain '0' }



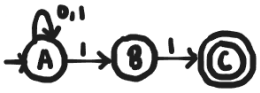
Ex 3) L3 = { Set of all strings that starts with '10' }



Ex 4) L4 = { Set of all strings that contain '01' }



Ex 5) L5 = { Set of all strings that ends with '11' }



NFA

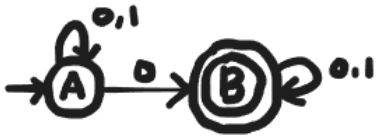
- L1 = set of all strings over {0,1} that ends with 1
- L2 = set of all strings over {0,1} that contains 0
- L3 = set of all strings over {a,b} that starts with ba

1.

$\Sigma = \{0,1\}$

2.

$\Sigma = \{0,1\}$



3.

$\Sigma = \{a,b\}$