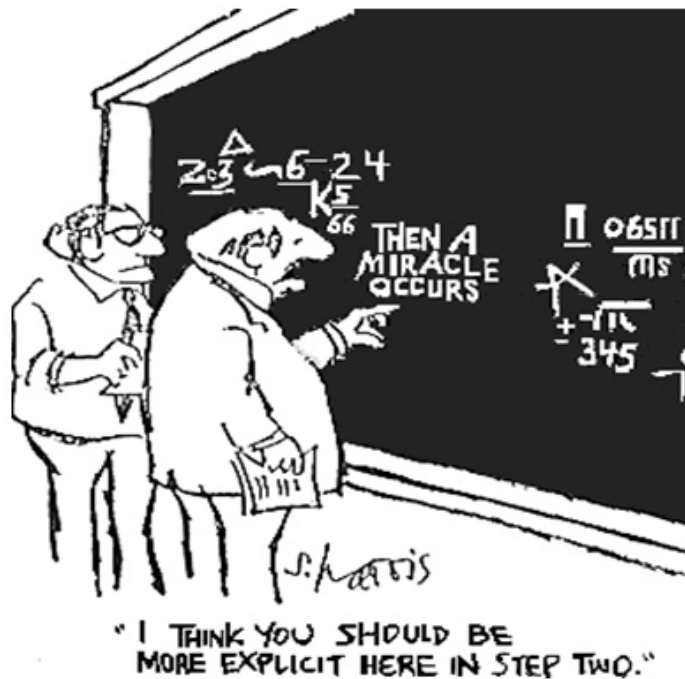


CSE 311: Foundations of Computing

Lecture 23: Finite State Machine Minimization & NFAs

Reminder
ACM BBQ today
4:00pm

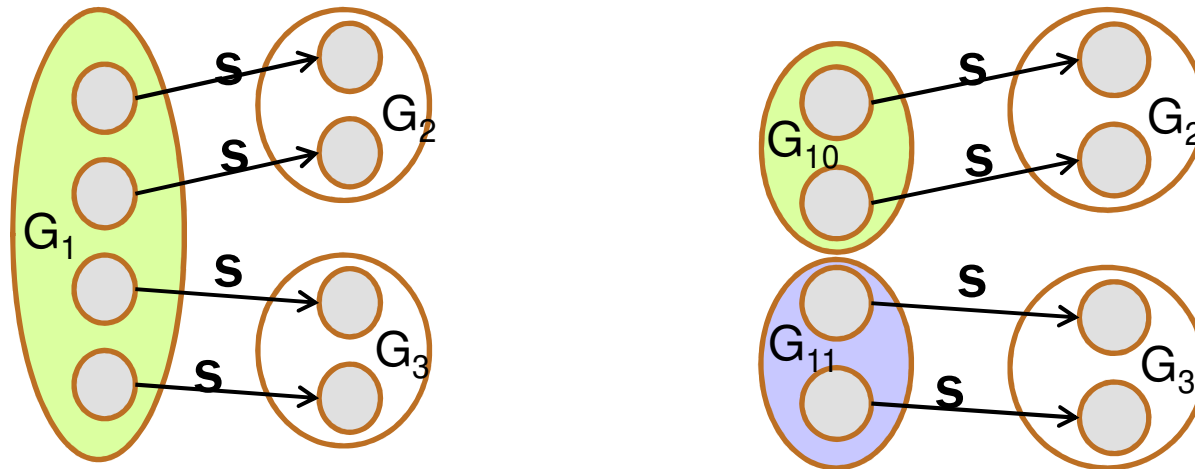


State Minimization

- **Many different FSMs (DFAs) for the same problem**
- **Take a given FSM and try to reduce its state set by combining states**
 - **Algorithm will always produce the unique minimal equivalent machine (up to renaming of states) but we won't prove this**

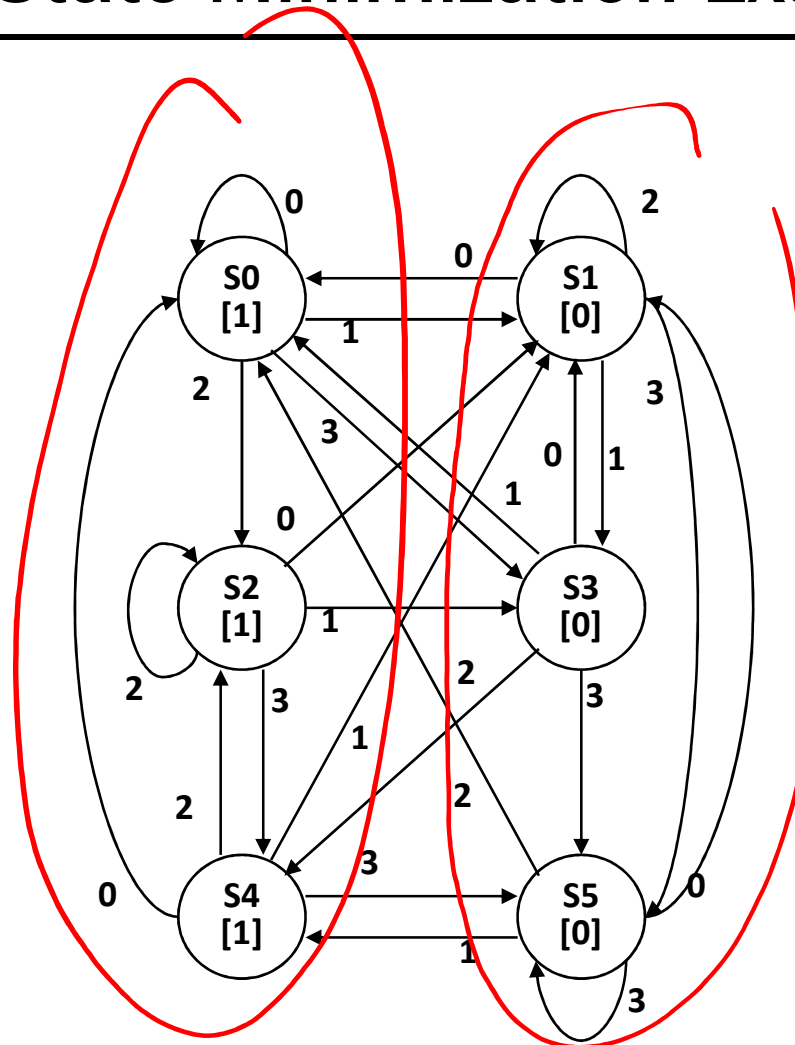
State Minimization Algorithm

1. Put states into groups based on their outputs (or whether they are final states or not)
2. Repeat the following until no change happens
 - a. If there is a symbol **s** so that not all states in a group **G** agree on which group **s** leads to, split **G** into smaller groups based on which group the states go to on **s**



3. Finally, convert groups to states

State Minimization Example

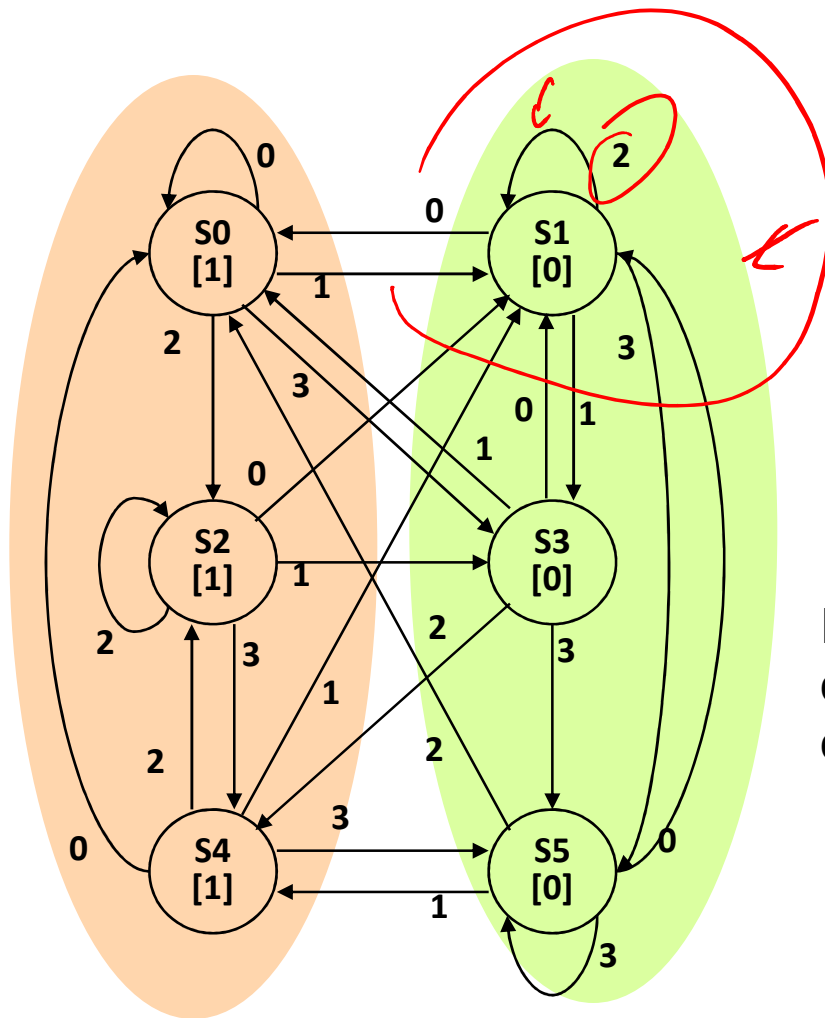


present state	next state				output
	0	1	2	3	
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

state transition table

Put states into groups based on their outputs (or whether they are final states or not)

State Minimization Example

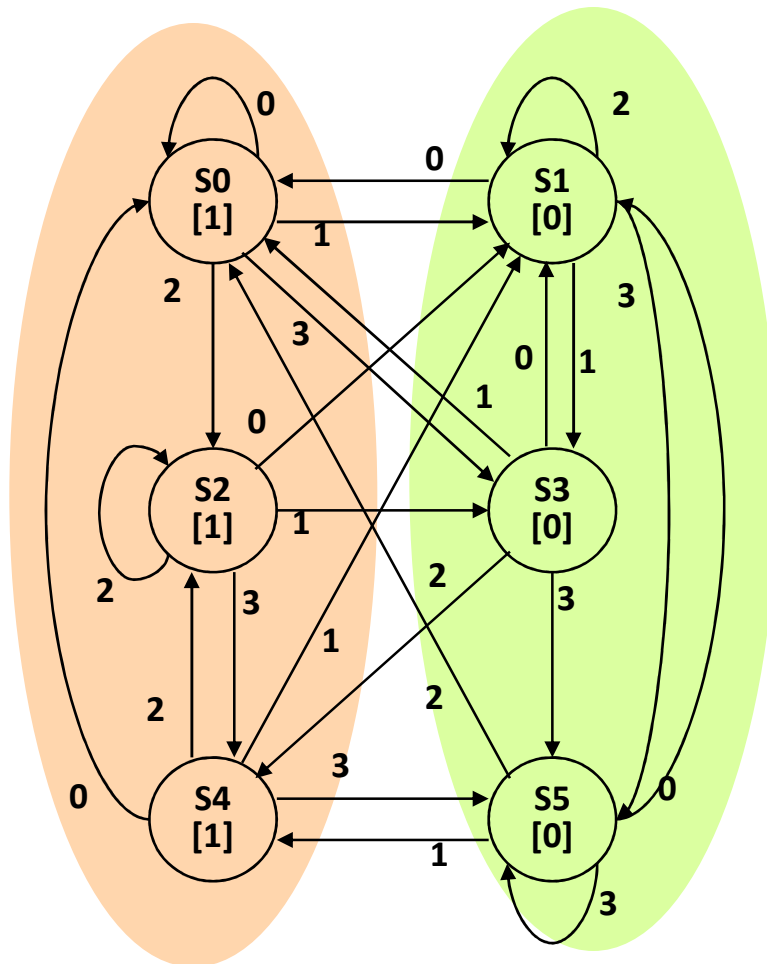


present state	next state				output
	0	1	2	3	
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

state transition table

Put states into groups based on their outputs (or whether they are final states or not)

State Minimization Example



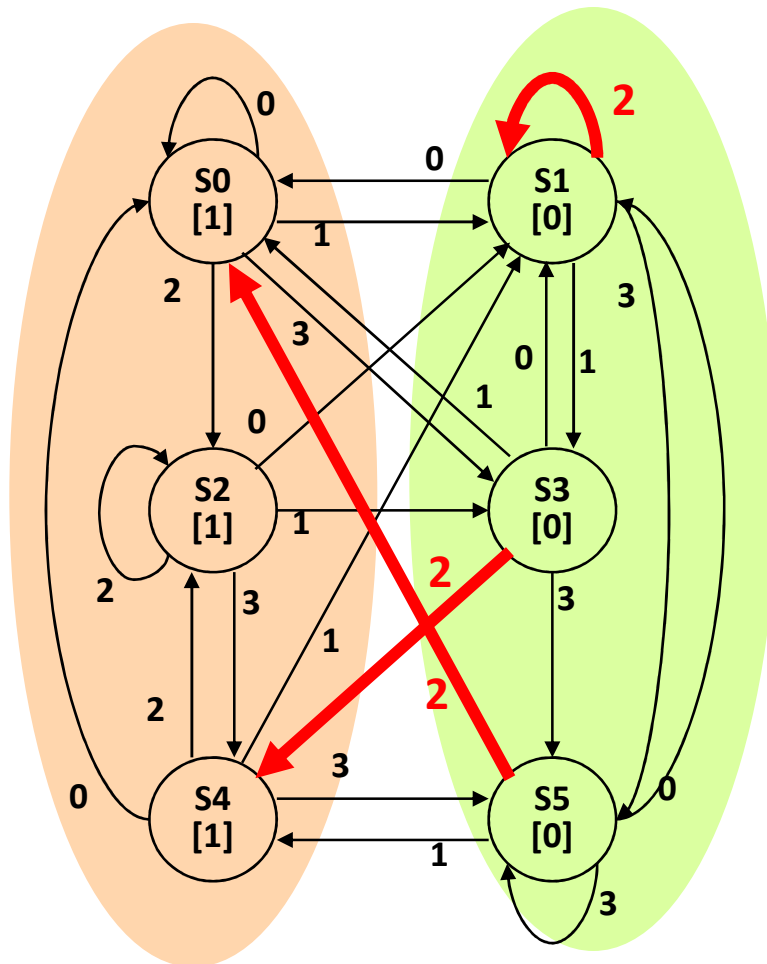
present state	next state				output
	0	1	2	3	
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

state transition table

Put states into groups based on their outputs (or whether they are final states or not)

If there is a symbol **s** so that not all states in a group **G** agree on which group **s** leads to, split **G** based on which group the states go to on **s**

State Minimization Example



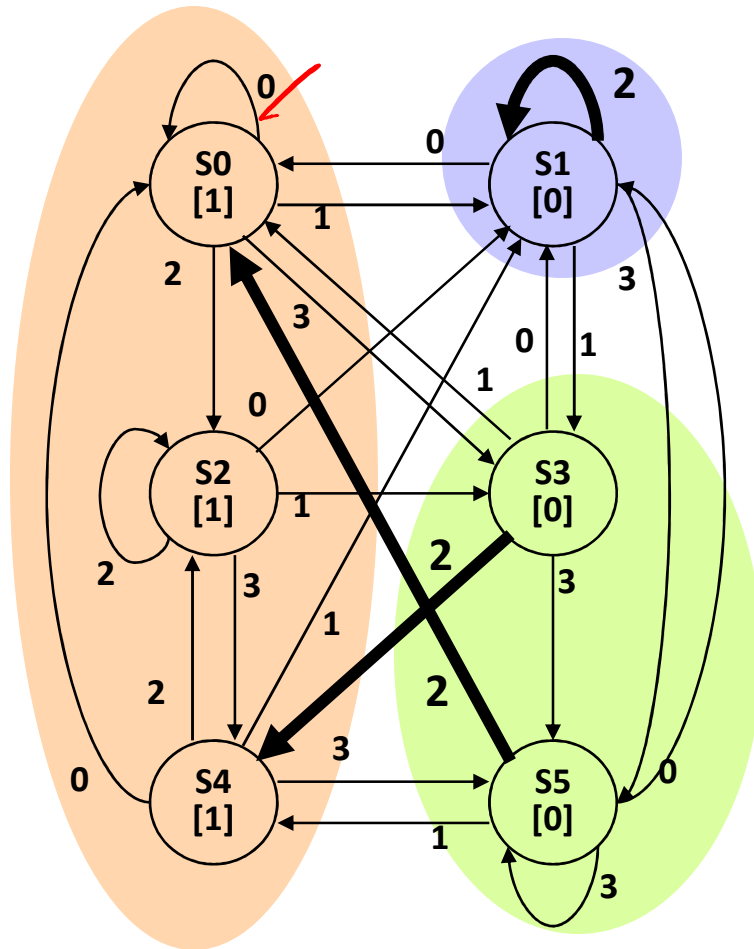
present state	next state				output
	0	1	2	3	
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

state transition table

Put states into groups based on their outputs (or whether they are final states or not)

If there is a symbol **s** so that not all states in a group **G** agree on which group **s** leads to, split **G** based on which group the states go to on **s**

State Minimization Example



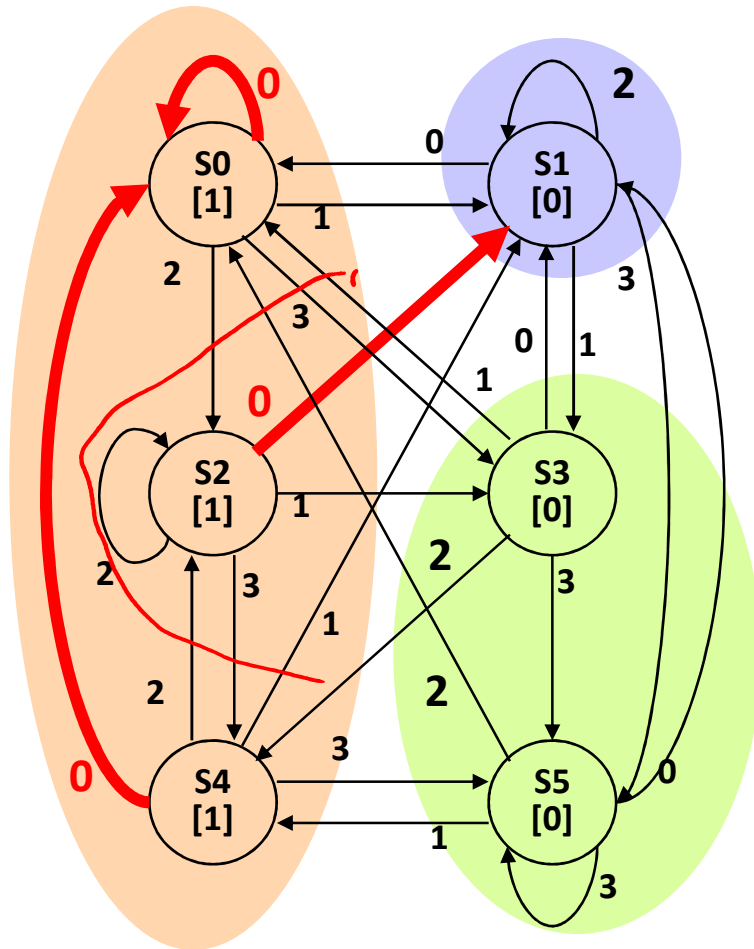
present state	next state				output
	0	1	2	3	
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

state transition table

Put states into groups based on their outputs (or whether they are final states or not)

If there is a symbol **s** so that not all states in a group **G** agree on which group **s** leads to, split **G** based on which group the states go to on **s**

State Minimization Example



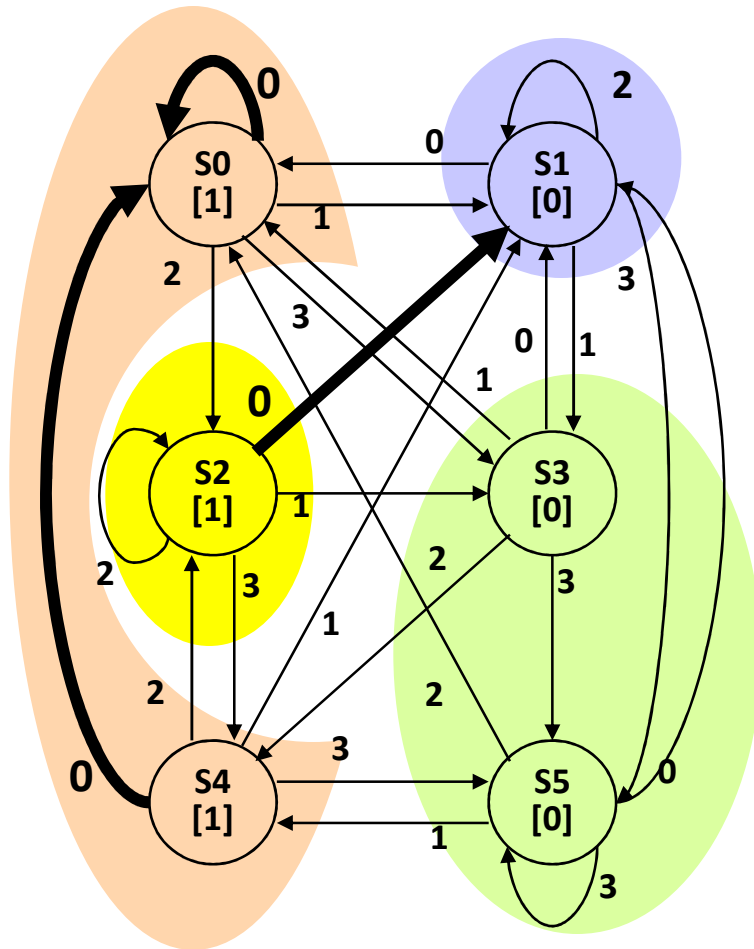
present state	next state				output
	0	1	2	3	
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

state transition table

Put states into groups based on their outputs (or whether they are final states or not)

If there is a symbol **s** so that not all states in a group **G** agree on which group **s** leads to, split **G** based on which group the states go to on **s**

State Minimization Example



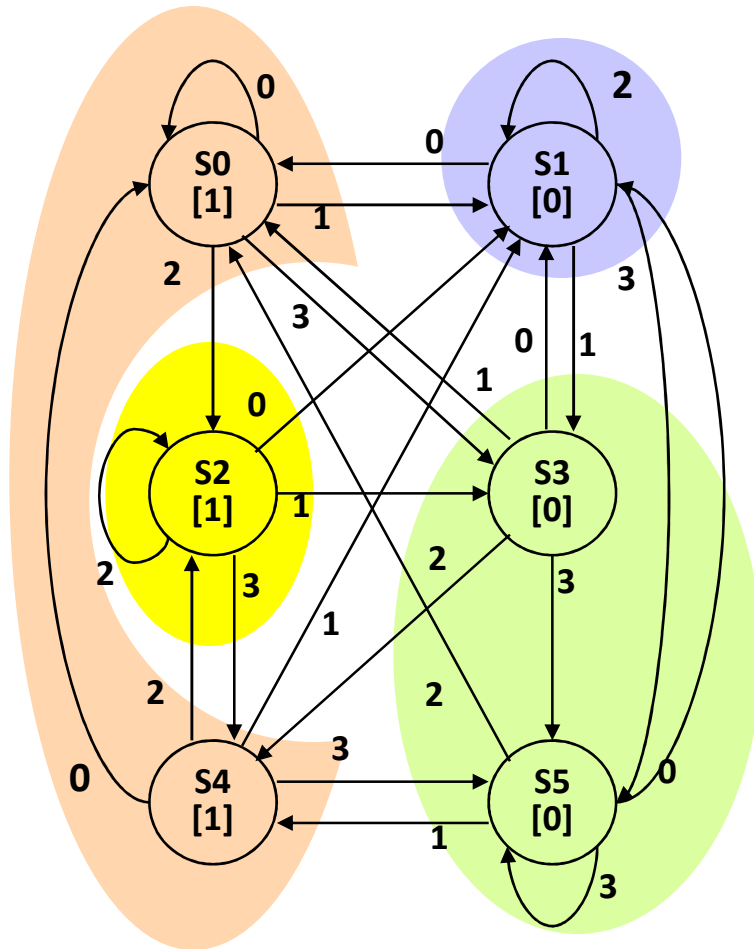
present state	next state				output
	0	1	2	3	
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

state transition table

Put states into groups based on their outputs (or whether they are final states or not)

If there is a symbol **s** so that not all states in a group **G** agree on which group **s** leads to, split **G** based on which group the states go to on **s**

State Minimization Example



present state	next state				output
	0	1	2	3	
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

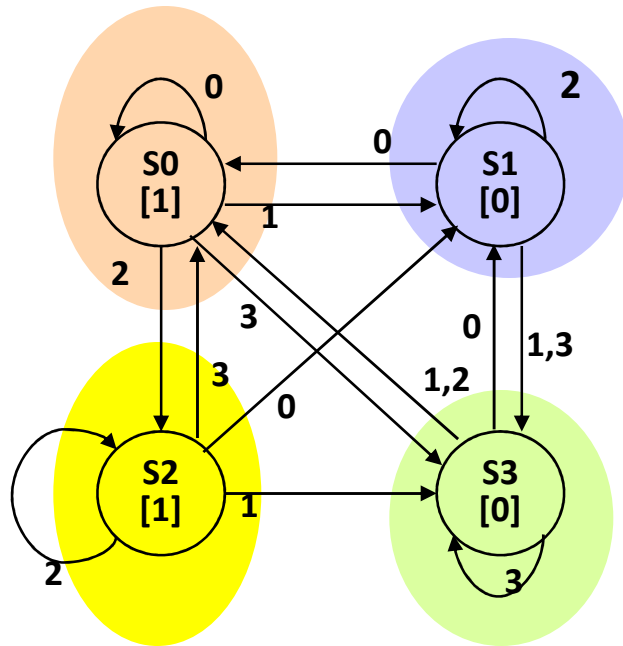
state transition table

Finally convert groups to states:

Can combine states S0-S4 and S3-S5.

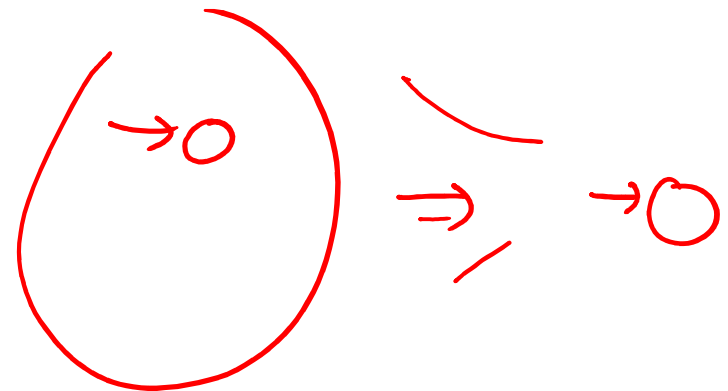
In table replace all S4 with S0 and all S5 with S3

Minimized Machine

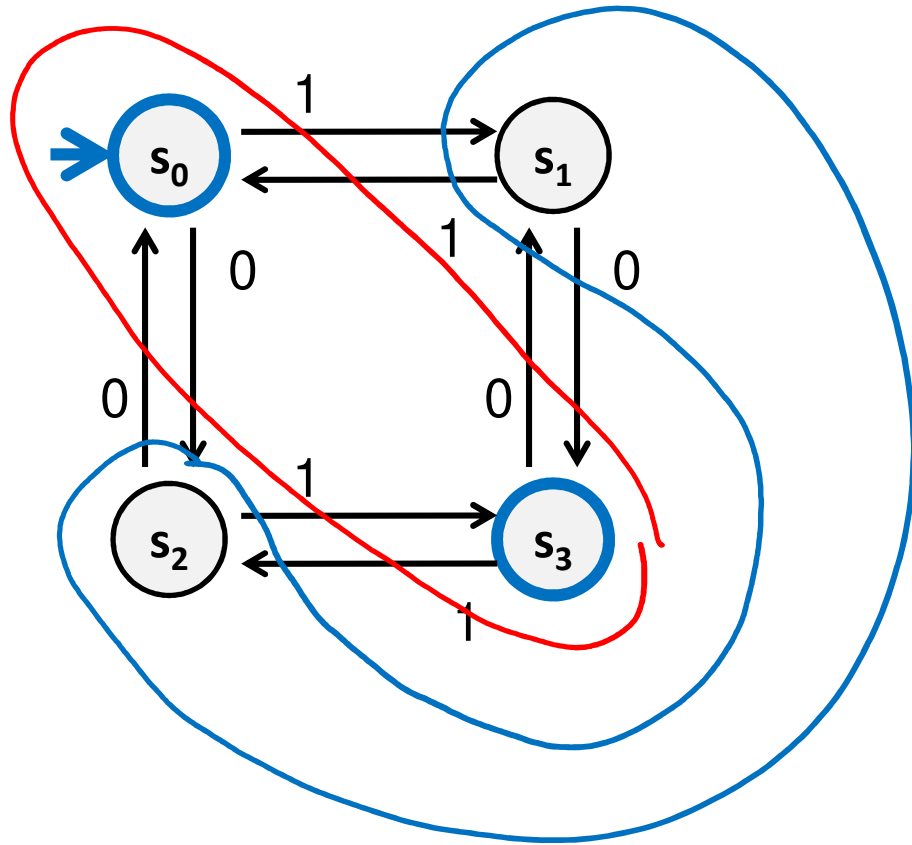


present state	next state				output
	0	1	2	3	
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S3	0
S2	S1	S3	S2	S0	1
S3	S1	S0	S0	S3	0

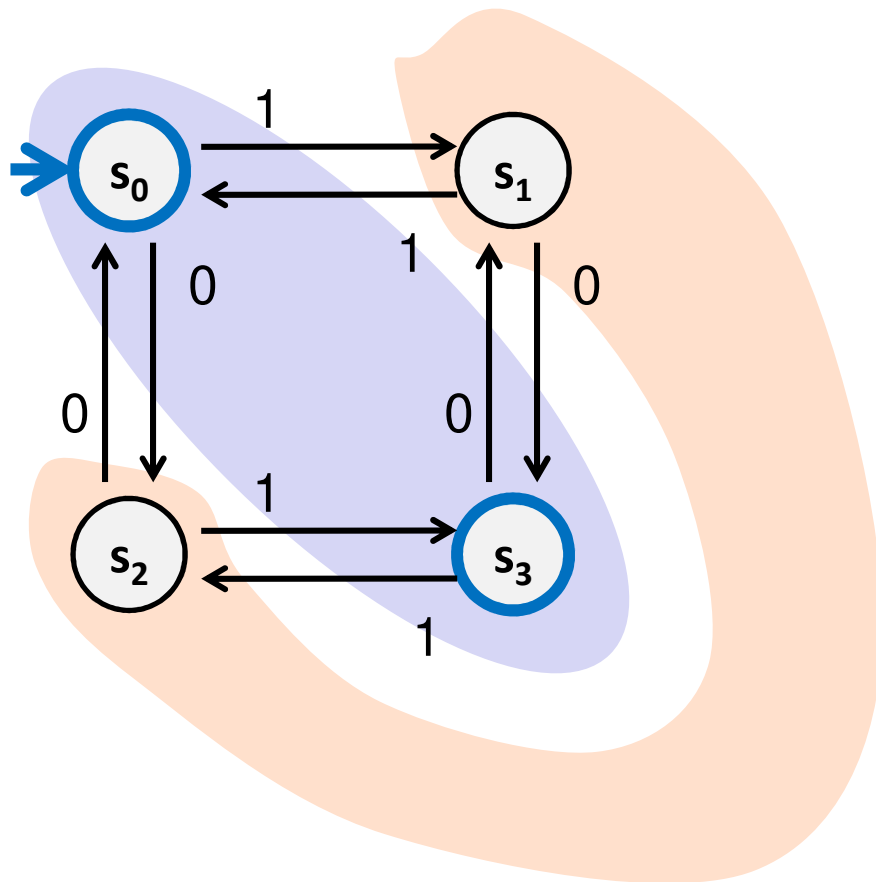
state transition table



A Simpler Minimization Example



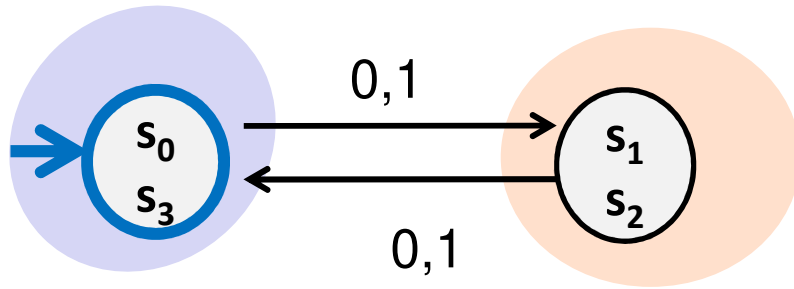
A Simpler Minimization Example



**Split states into
final/non-final groups**

**Every symbol causes
the DFA to go from one
group to the other so
neither group needs to
be split**

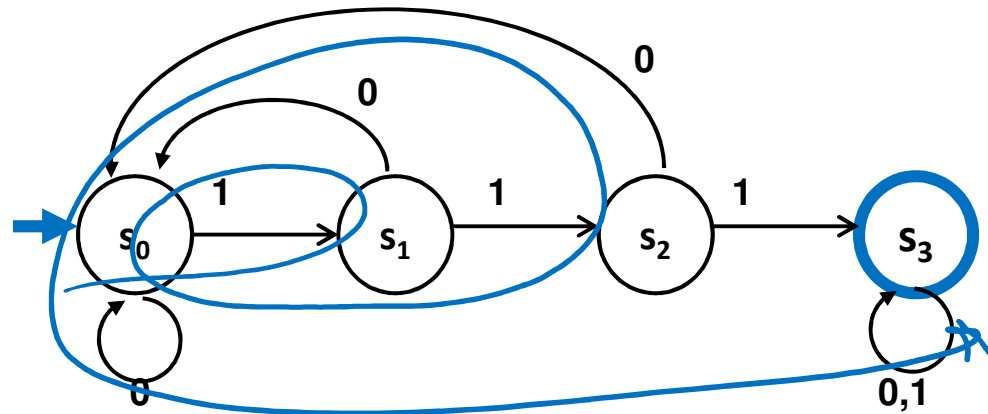
Minimized DFA



Another way to look at DFAs

Definition: The label of a path in a DFA is the concatenation of all the labels on its edges in order

Lemma: x is in the language recognized by a DFA iff x labels a path from the start state to some final state

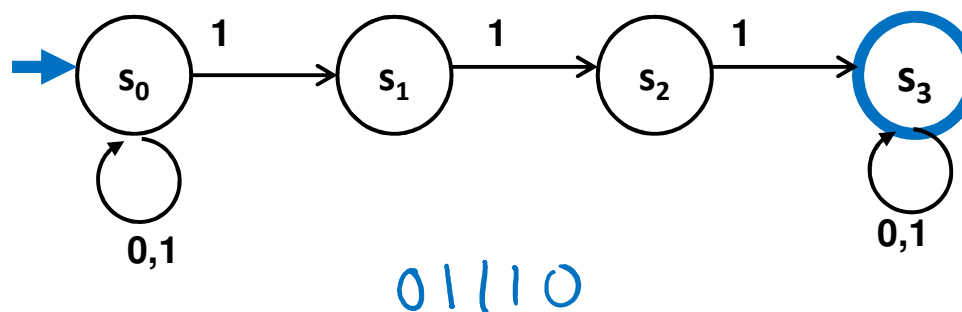


10110111

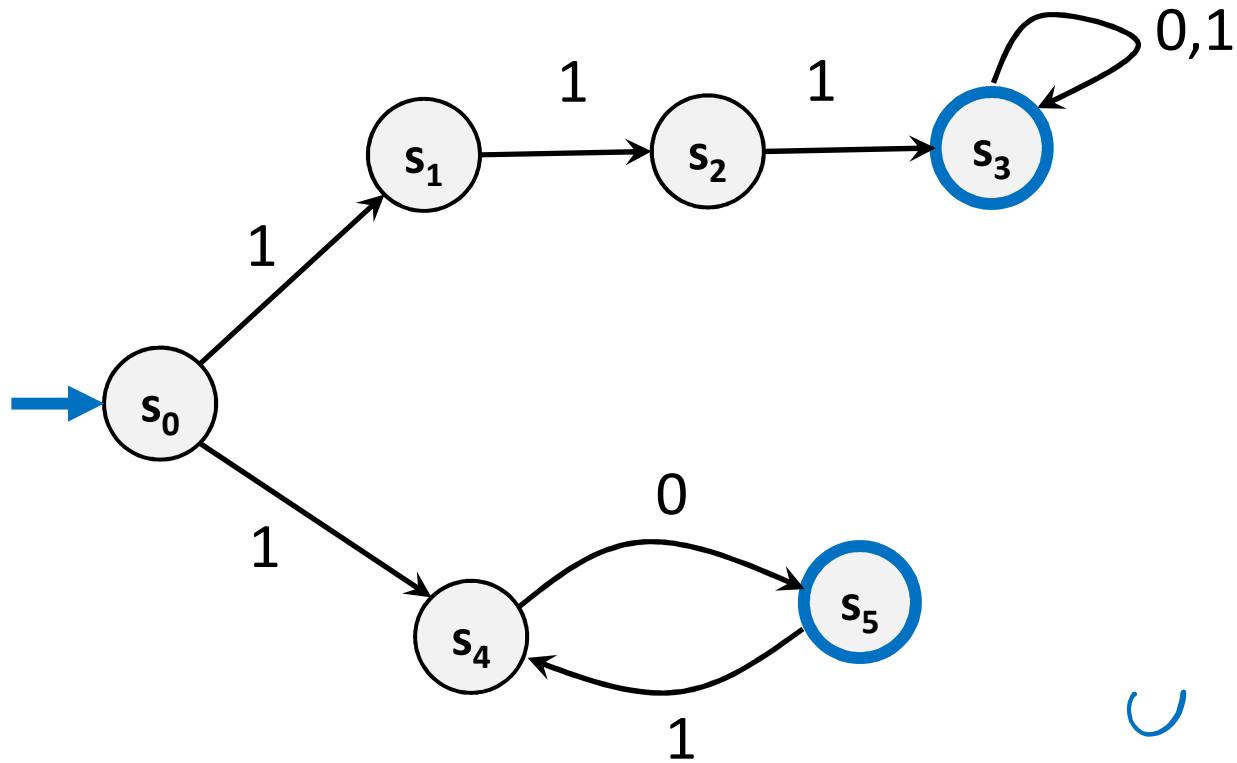
Nondeterministic Finite Automata (NFA)

- Graph with start state, final states, edges labeled by symbols (like DFA) but
 - Not required to have exactly 1 edge out of each state labeled by each symbol— can have 0 or >1
 - Also can have edges labeled by empty string ϵ

• **Definition:** x is in the language recognized by an NFA if and only if x labels a path from the start state to some final state



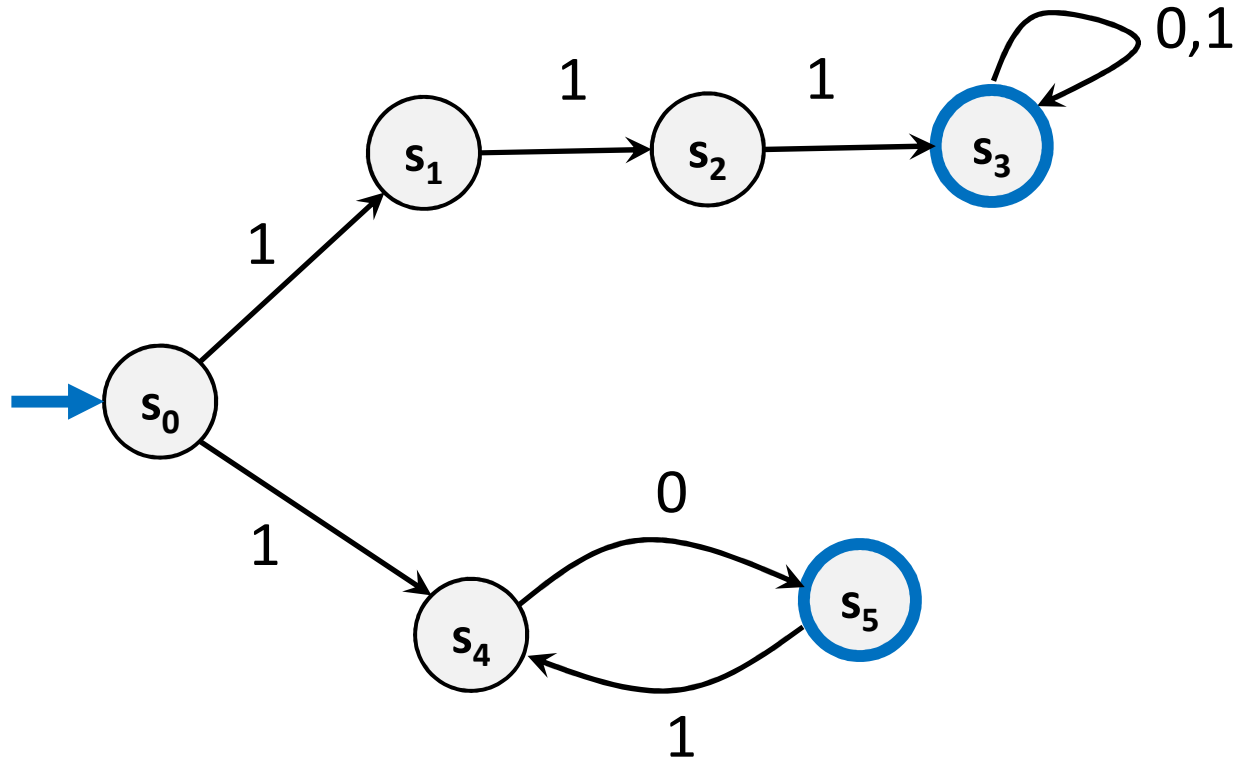
Consider This NFA



$111(001)^*$
 $\cup 10(10)^*$

What language does this NFA accept?

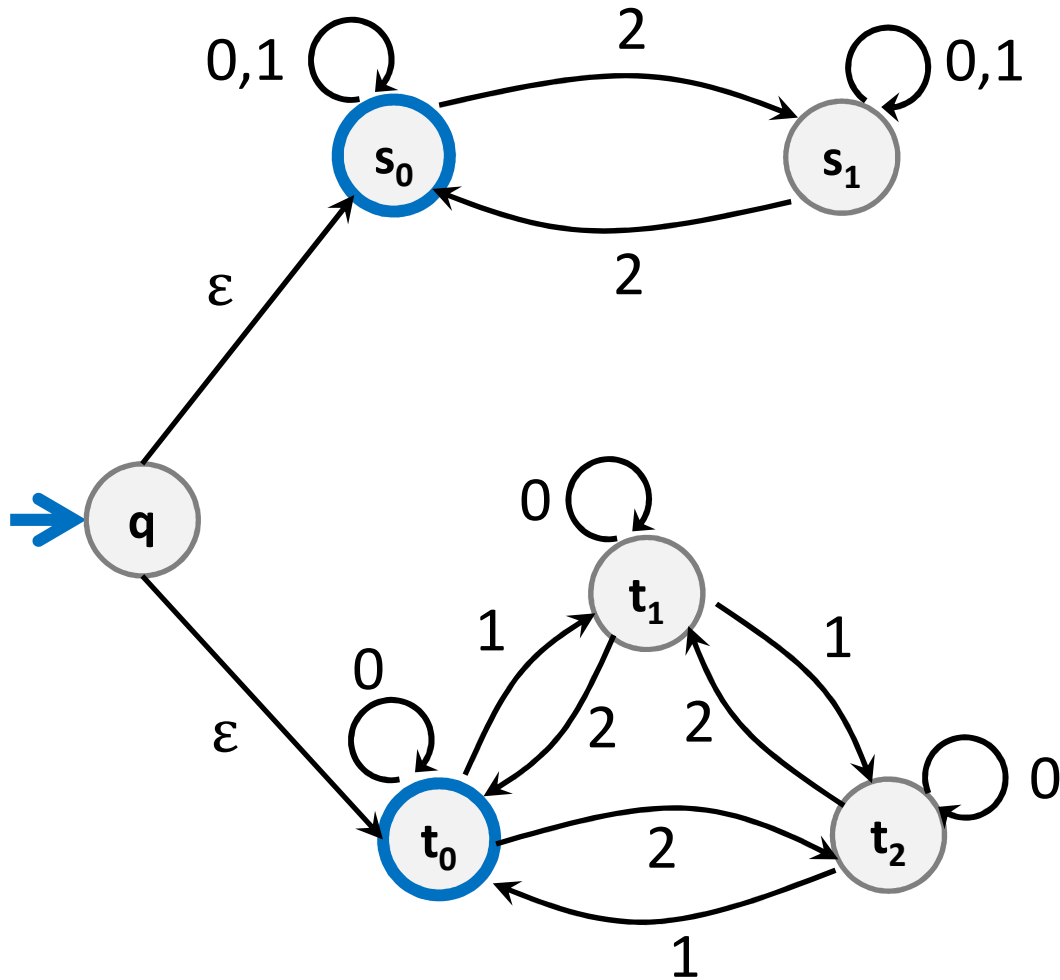
Consider This NFA



What language does this NFA accept?

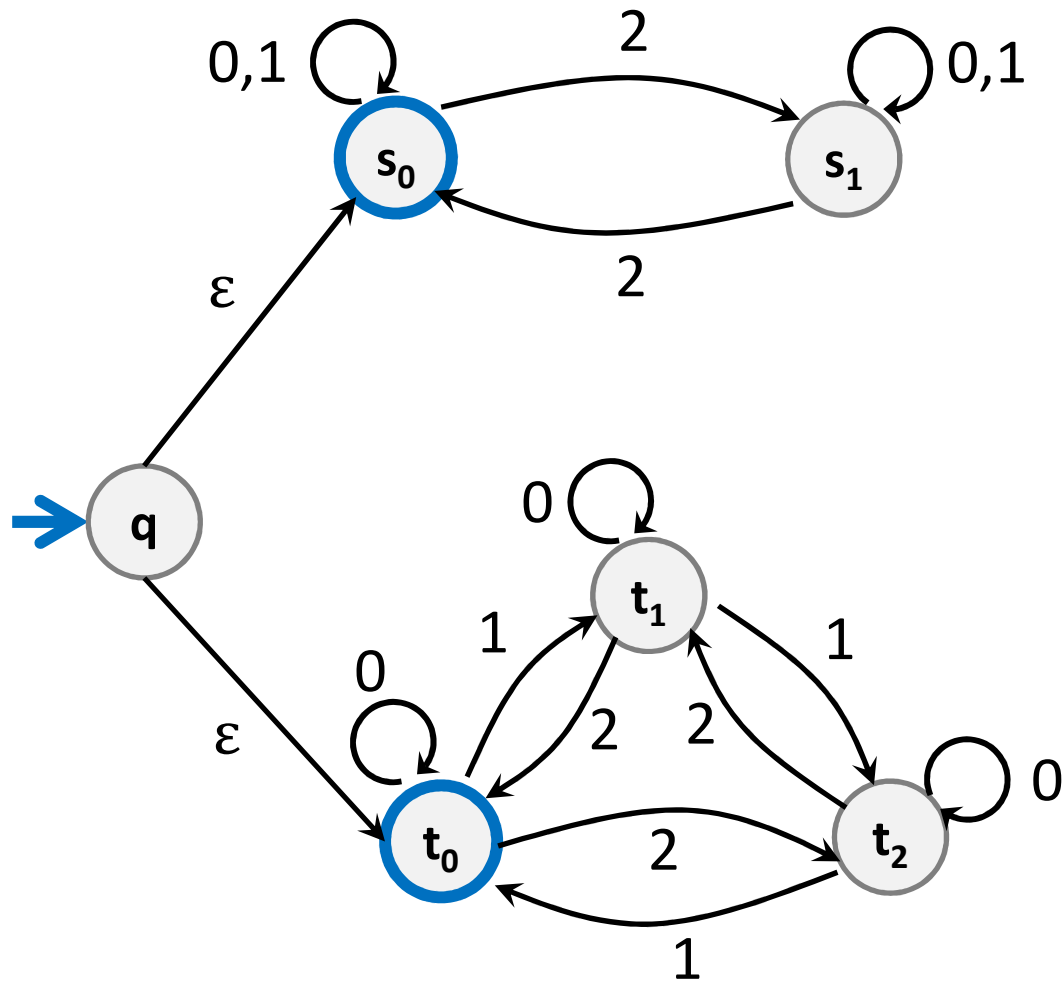
$$10(10)^* \cup 111(0 \cup 1)^*$$

NFA ϵ -moves



NFA ϵ -moves

Strings over $\{0,1,2\}$ w/even # of 2's OR sum to $0 \pmod 3$



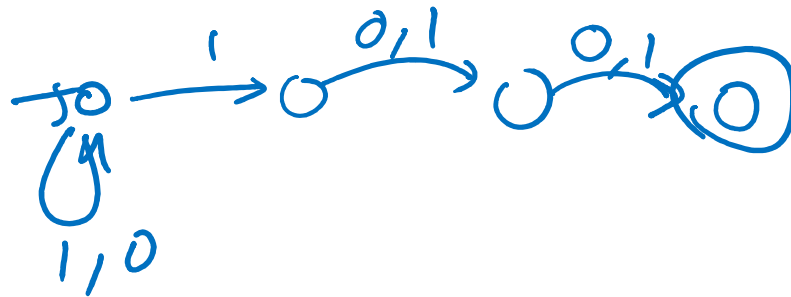
Three ways of thinking about NFAs

- **Outside observer:** Is there a path labeled by x from the start state to some final state?

- **Perfect guesser:** The NFA has input x and whenever there is a choice of what to do it magically guesses a good one (if one exists)

- **Parallel exploration:** The NFA computation runs all possible computations on x step-by-step at the same time in parallel

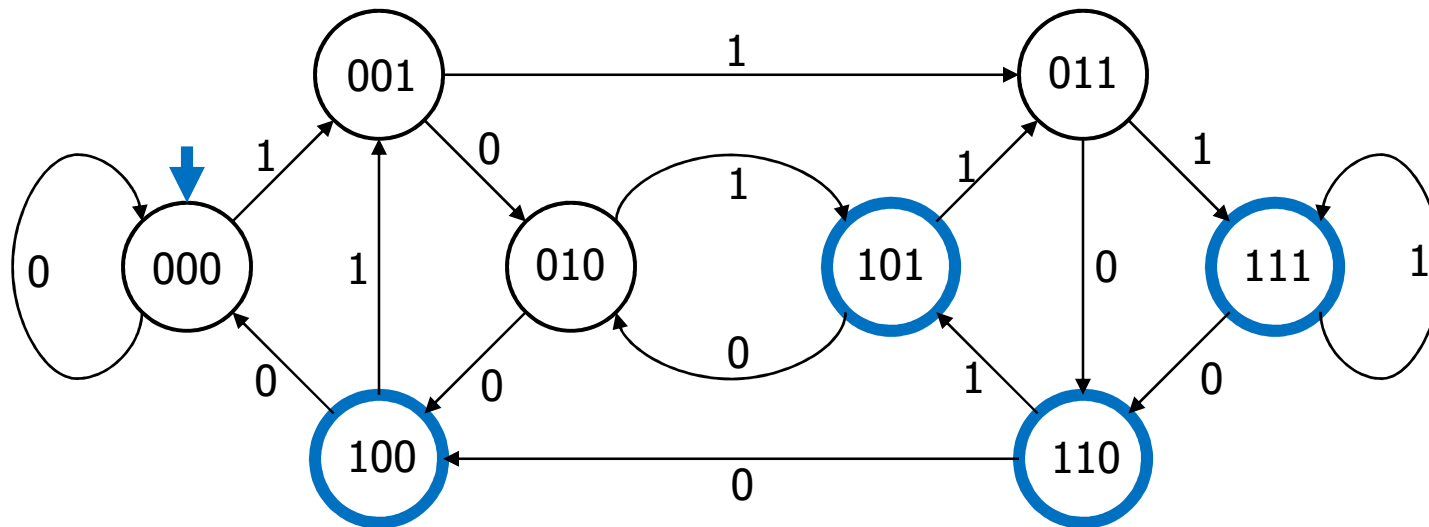
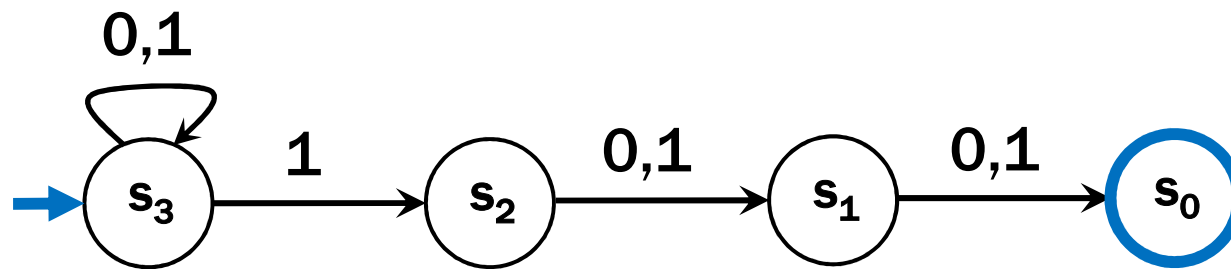
NFA for set of binary strings with a 1 in the 3rd position from the end



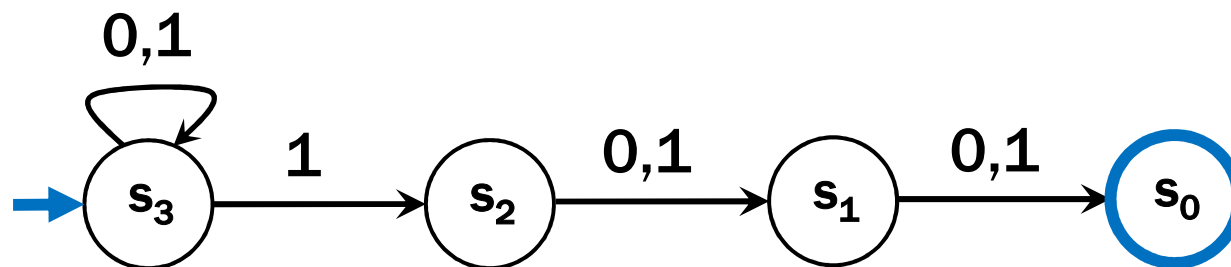
NFA for set of binary strings with a 1 in the 3rd position from the end



Compare with the smallest DFA



Parallel Exploration view of an NFA



Input string 0101100

