CSE 311 Foundations of Computing I

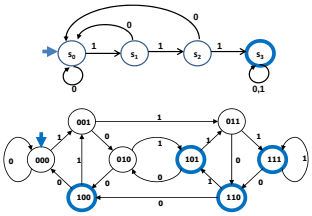
Lecture 22
Finite State Machines: Output and Minimization
Autumn 2011

Announcements

- Reading assignments
  - 7th Edition, Sections 13.3 and 13.4
  - 6th Edition, Section 12.3 and 12.4
  - 5th Edition, Section 11.3 and 11.4

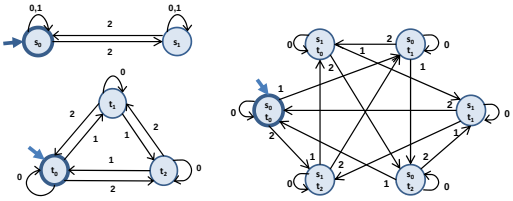
Last lecture highlights

Finite state machines
- States, transitions, start state, final states
- Languages recognized by FSMs



Last lecture highlights

- Combining FSMs to check two properties at once
  - New states record states of both FSMs

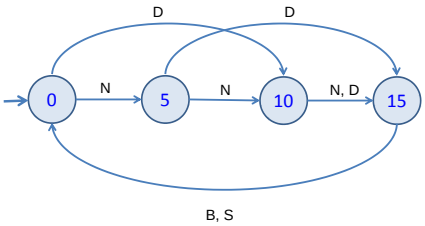


State Machines with Output

SNICKERS Vending Machine Butterfinger

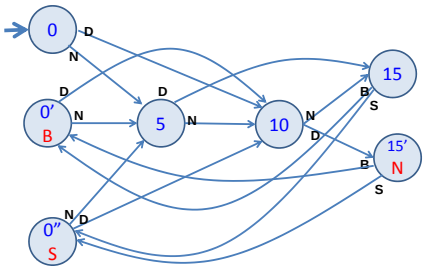
Enter 15 cents in dimes or nickels
Press S or B for a candy bar

Vending Machine, Version 1



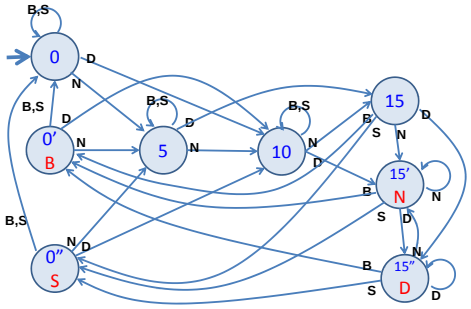
Basic transitions on N (nickel), D (dime), B (butterfinger), S (snickers)

Vending Machine, Version 2



Adding output to states: N – Nickel, S – Snickers, B – Butterfinger

Vending Machine, Final Version



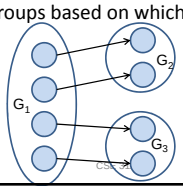
Adding additional "unexpected" transitions

State Minimization

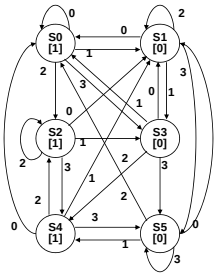
- Many different FSMs (DFAs) for the same problem
- Take a given FSM and try to reduce its state set by combining states
  - Algorithm will always produce the unique minimal equivalent machine (up to renaming of states) but we won't prove this

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***



State Minimization Example

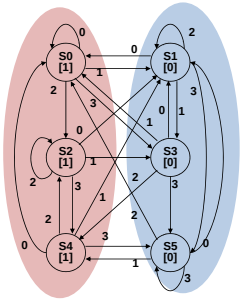


present state	next state 0	1	2	3	output
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

state transition table

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

State Minimization Example

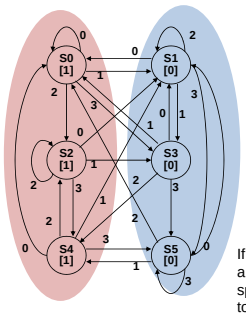


present state	next state 0	1	2	3	output
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

state transition table

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

State Minimization Example



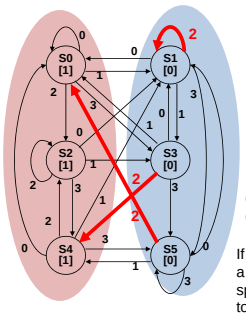
present state	next state 0	1	2	3	output
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

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



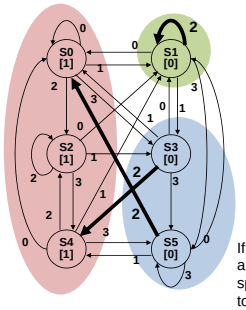
present state	next state 0	1	2	3	output
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

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



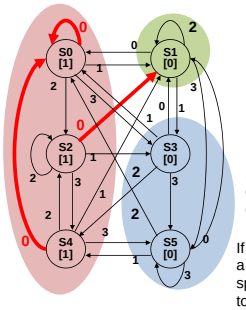
present state	next state 0	1	2	3	output
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

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



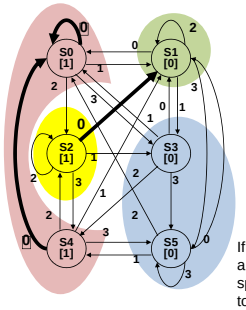
present state	next state 0	1	2	3	output
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

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



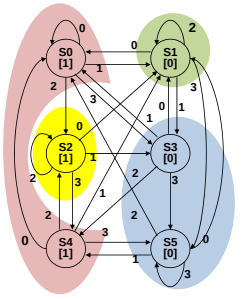
present state	next state 0	1	2	3	output
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

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



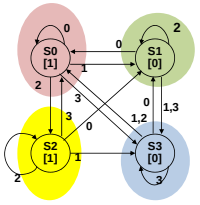
present state	next state 0	1	2	3	output
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S5	0
S2	S1	S3	S2	S4	1
S3	S1	S0	S4	S5	0
S4	S0	S1	S2	S5	1
S5	S1	S4	S0	S5	0

state transition table

Can combine states S0-S4 and S3-S5.

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

Minimized Machine



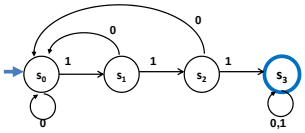
present state	next state 0	1	2	3	output
S0	S0	S1	S2	S3	1
S1	S0	S3	S1	S3	0
S2	S1	S3	S2	S0	1
S3	S1	S0	S0	S3	0

state transition table

Another way to look at DFAs

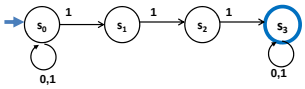
Definition: The label of a path in a DFA is the concatenation of all the labels on its edges in order

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



Nondeterministic Finite Automaton (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 **λ**
- Definition: x is in the language recognized by an NFA iff x labels a path from the start state to some final state



Design an NFA to recognize the set of binary strings that contain 111 or have an even # of 1's