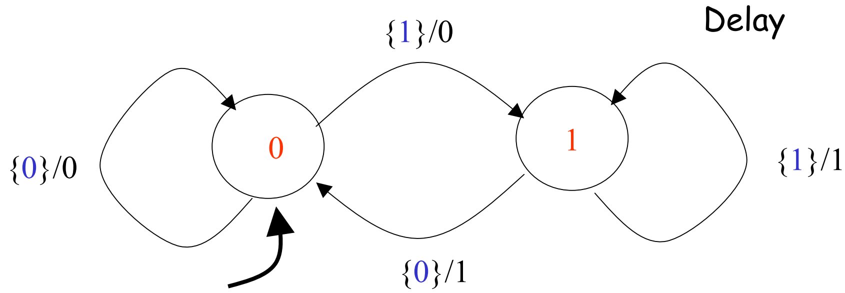
Week 4

1. More examples
2. Nondeterminism, equivalence, simulation (Ch 3)
3. Composition (Ch 4)

Delay



InputSignals = OutputSignals = [$Nats_0 \to$ {0,1, absent}]

x = 0 0 1 1 0 …
s = 0 0 0 1 1 0 …
y = 0 0 0 1 1 0 …

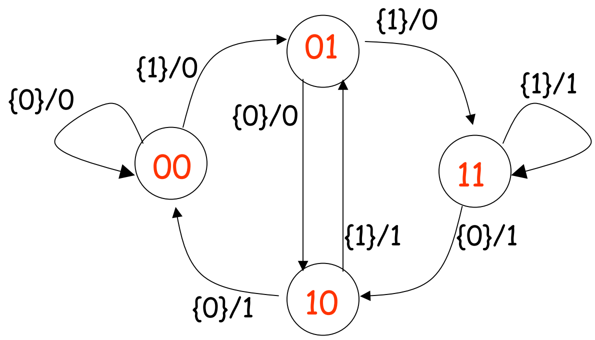
$\forall x \in$ InputSignals, $\forall n \in Nats_0$, Delay(x)(n)=0, n=0;
= x(n-1), n > 0

$\forall x \in$ InputSignals, $\forall n \in Nats_0$,

$Delay_2(x)(n) = 0, \quad n = 0,1;$

$= x(n-2), \quad n = 2,3,\ldots$

Implement $Delay_2$ as state machine



We will see later that $Delay_2 \sim Delay_1 \circ Delay_1$

Nondeterministic state machines

In deterministic machines guards from state state are disjoint

In nondeterministic machines guards may not be disjoint. What does that mean?

<u>Topics/determinism/example</u>

The same input signal can lead to more than one state response and output signal

Set and function model

N = (States, Inputs, Outputs, possibleUpdates, initialState)

possibleUpdates: States x Inputs $\rightarrow$ P(States x Outputs)

where P(States x Outputs) is the set of all non-empty subsets of States x Outputs

Topics/deterministic/possible updates

Always: possibleUpdate(s, absent) = {(s,absent)}

A deterministic machine determines a function

H: InputSignals $\rightarrow$ OutputSignals

A nondeterministic machine determines a relation
Behaviors = {(x,y) | y is a possible output signal corresponding to x}
$\subset$ InputSignals x OutputSignals

Why non-deterministic machines?

1. Topics/determinism/Abstraction

2. Topics/determinism/Equivalence

3. Topics/determinism/Simulation

The matching game

Two (nondeterministic) machines,

A = ($States_A$, Inputs, Outputs, $possibleUpdates_A$, $s_A(0)$)

B = ($States_B$, Inputs, Outputs, $possibleUpdates_B$, $s_B(0)$)

Suppose input symbol x and
A moves from $s_A(0)$ to $s_A(1)$ and produces output y

Then for same input symbol x
B can select move from $s_B(0)$ to $s_B(1)$, to produce y

and continue the game from states $s_A(1)$, $s_B(1)$

B simulates A if there is a subset

$S \subset States_A \times States_B$

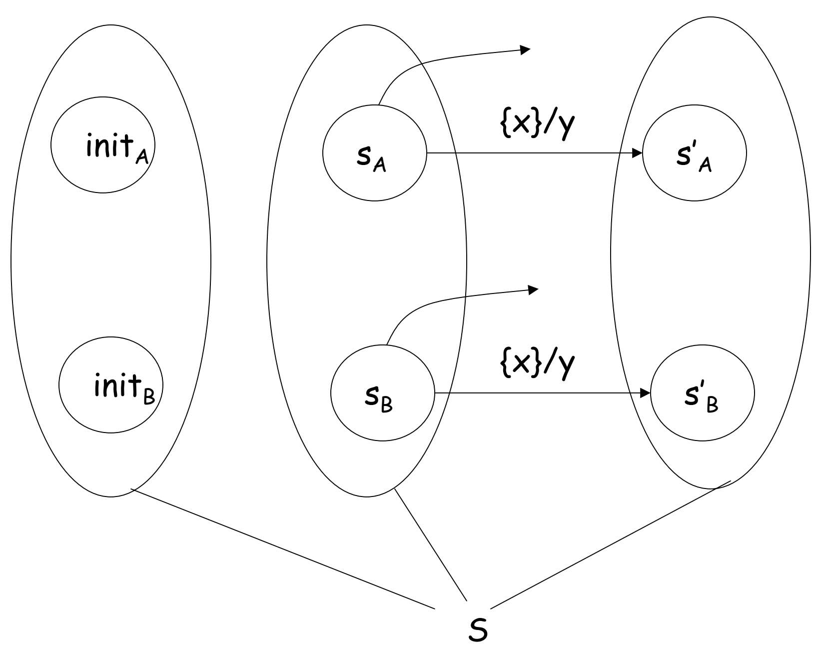
such that

1. $(initialState_A, initialState_B) \in S$, and

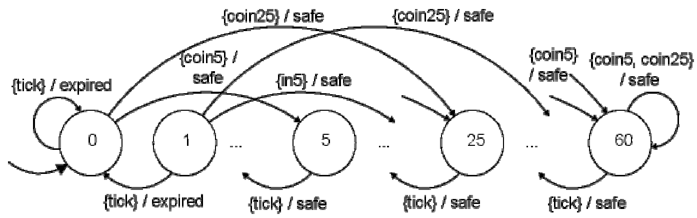
2. $\forall (s_A, s_B) \in S, \forall x \in Inputs,$
   $\forall (s'_A, y) \in possibleUpdates_A(s_A, x)$

   $\exists (s'_B, y) \in possibleUpdates_B(s_B, x)$
   such that

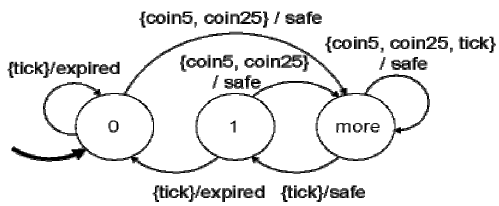
   $(s'_A, s'_B) \in S$



A



B



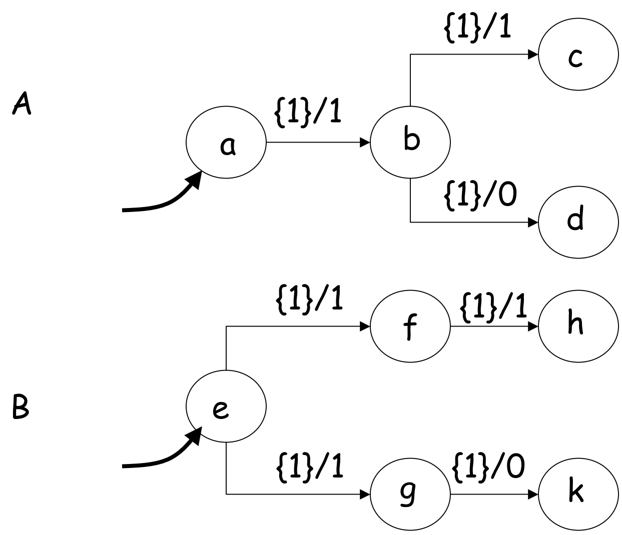
$S = \{(0_A, 0_B), (1_A, 1_B), (2_A, \text{more}), \ldots (60_A, \text{more})\}$

Theorem Suppose B simulates A. Then,

$$\text{Behaviors}_A \subset \text{Behaviors}_B$$

i.e. if y is a possible output response to x by machine A, y is also a possible output response to x by machine B.

Question Suppose B simulates A and C simulates B. Does C simulate A?



Behaviors of A = Behaviors of B. Why?

But B does not simulate A. Why?

Topics/Composition/Synchrony

1. Each component reacts once for every input symbol
2. The following happens simultaneously for each component
   - The input symbol is consumed
   - A state update occurs leading to next state and producing current output
   - If there is a feedback loop, the output appears at the input port

Topics/Composition/Side-by-side

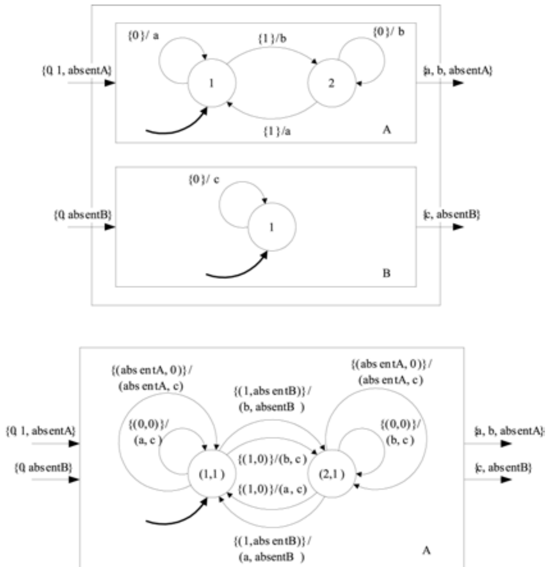
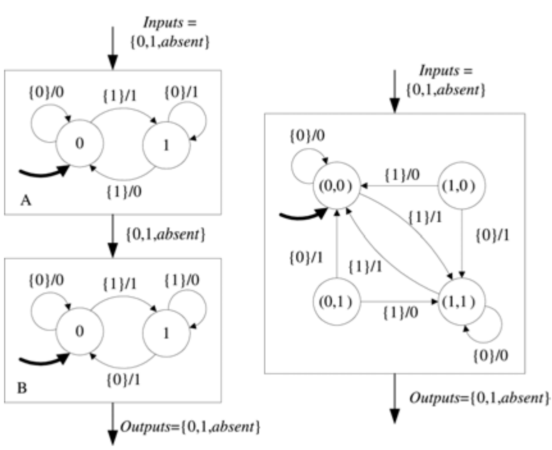


Fig 4.2, p. 127

Topics/Composition/Cascade



States (0,1) and (1,0) of cascade machine are NOT reachable

Fig 4.4, p 131

Topics/Composition/Productform

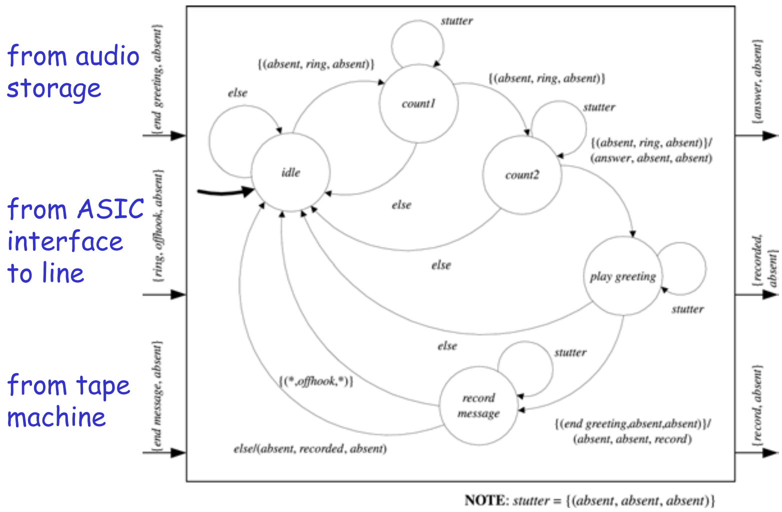


Fig 4.6, p. 134 Answering machine

Topics/Composition/Series-Parallel

Topics/Composition/Playback

Topics/Composition/Playback

Topics/Composition/Composition