Nondeterminism

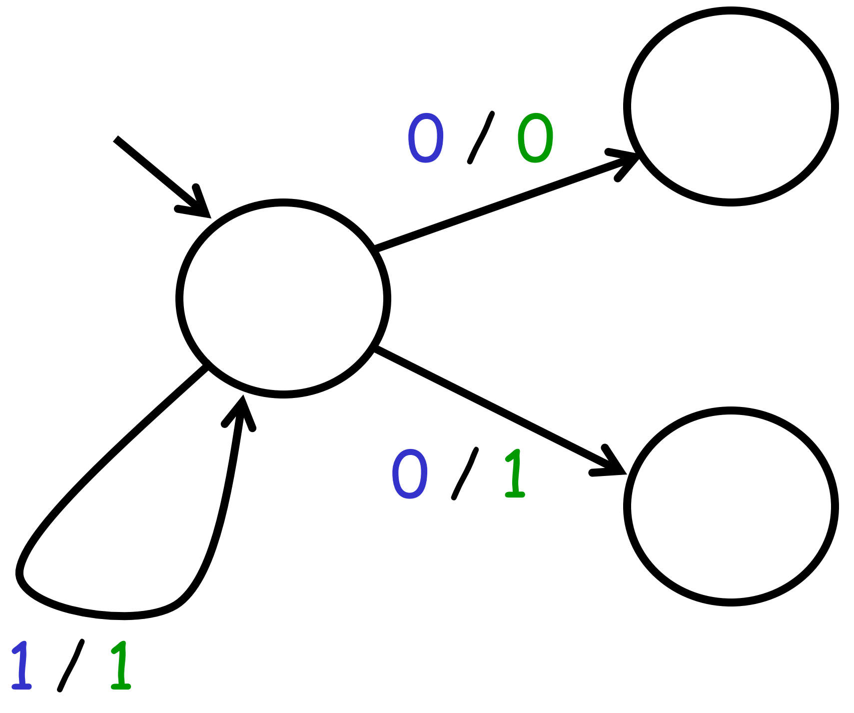
EECS 20

Lecture 14 (February 16, 2001)

Tom Henzinger

Revised by Matteo Zavatteri and Tiziano Villa, Fall 2020

Nondeterminism



Nondeterminism

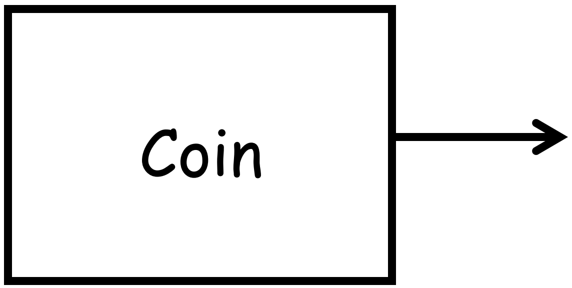
-modeling randomness

-modeling abstraction

-modeling uncertainty

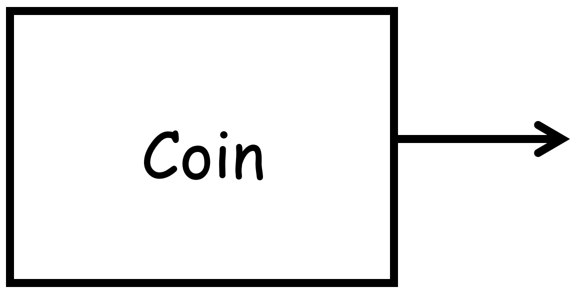
-modeling properties

Modeling Randomness: Coin Tossing

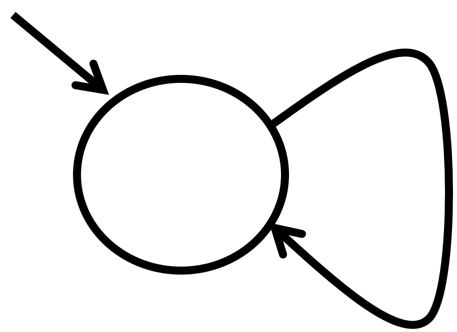


$\mathrm{Nats}_0 \to \{ \text{heads}, \text{tails} \}$

Modeling Randomness: Coin Tossing



$Nats_0 \to \{\ heads, tails\ \}$



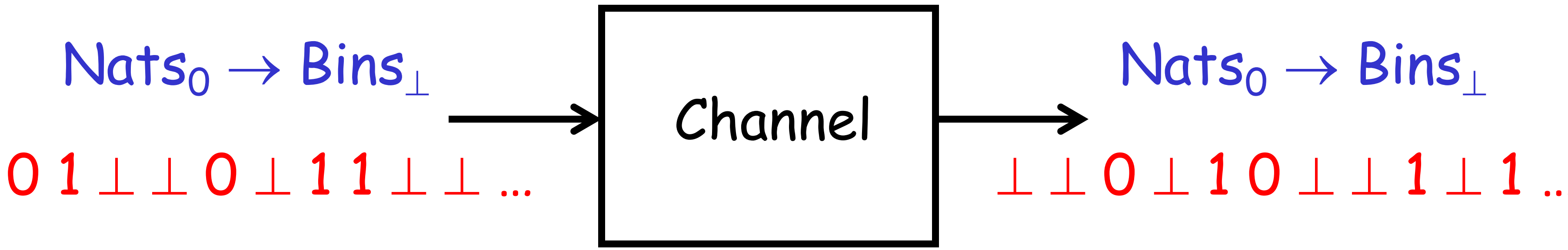
One possible behavior

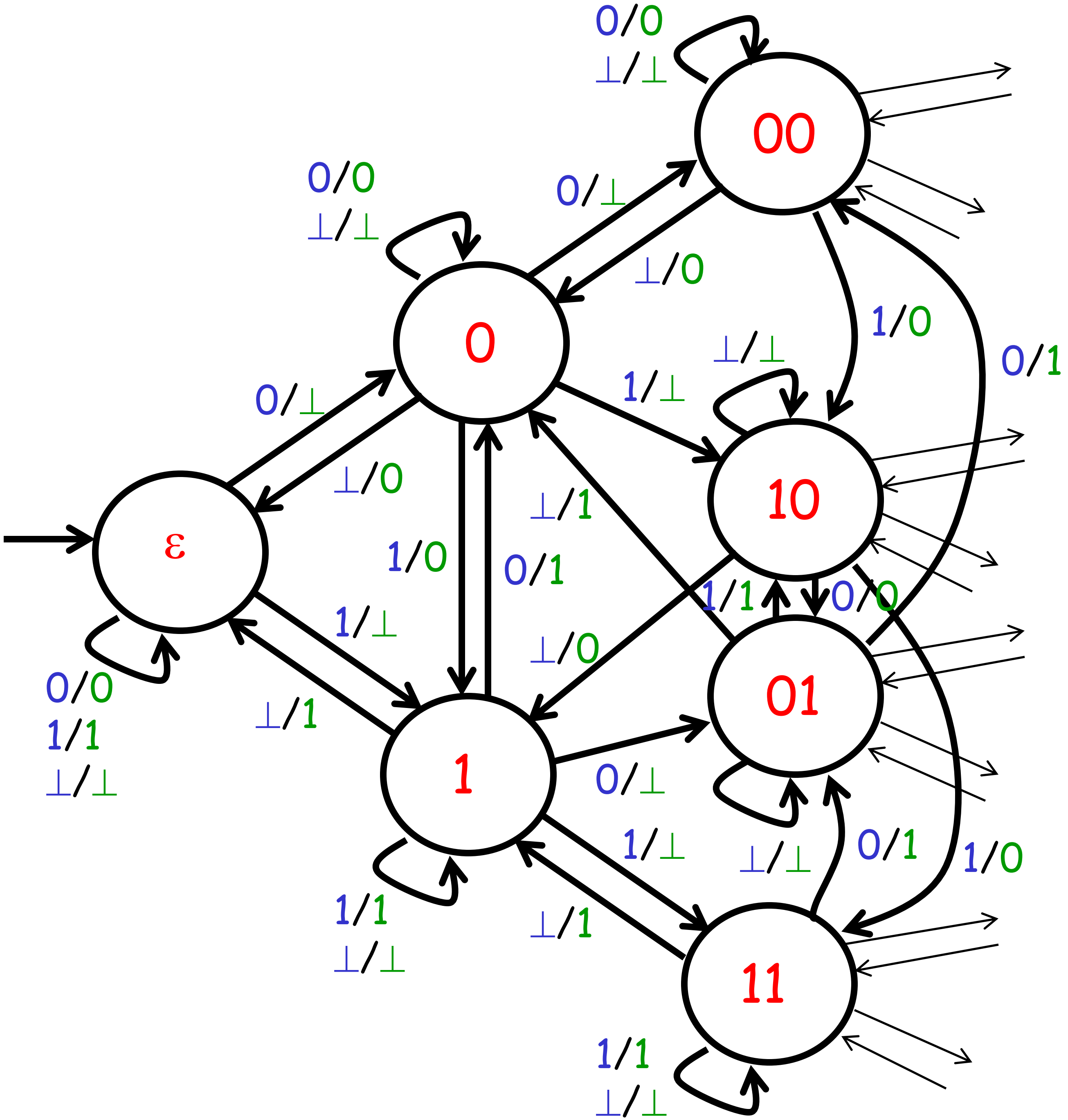
Time	0	1	2	3	4
Input	•	•	•	•	•
Output	heads	heads	tails	heads	tails

Another possible behavior

Time	0	1	2	3	4
Input	•	•	•	•	•
Output	tails	heads	tails	heads	heads

Modeling Abstraction: Channel Latency





State = channel contents

One possible run

Time		0		1		2		3		4		5	
Input		0		1		$\perp$		0		$\perp$		$\perp$	
Output		$\perp$		$\perp$		$\perp$		0		$\perp$		1	
State	ε		0		10		10		01		01		0

One possible run

Time		0		1		2		3		4		5	
Input		0		1		⊥		0		⊥		⊥	
Output		⊥		⊥		⊥		0		⊥		1	
State	ε		0		10		10		01		01		0

Corresponding behavior

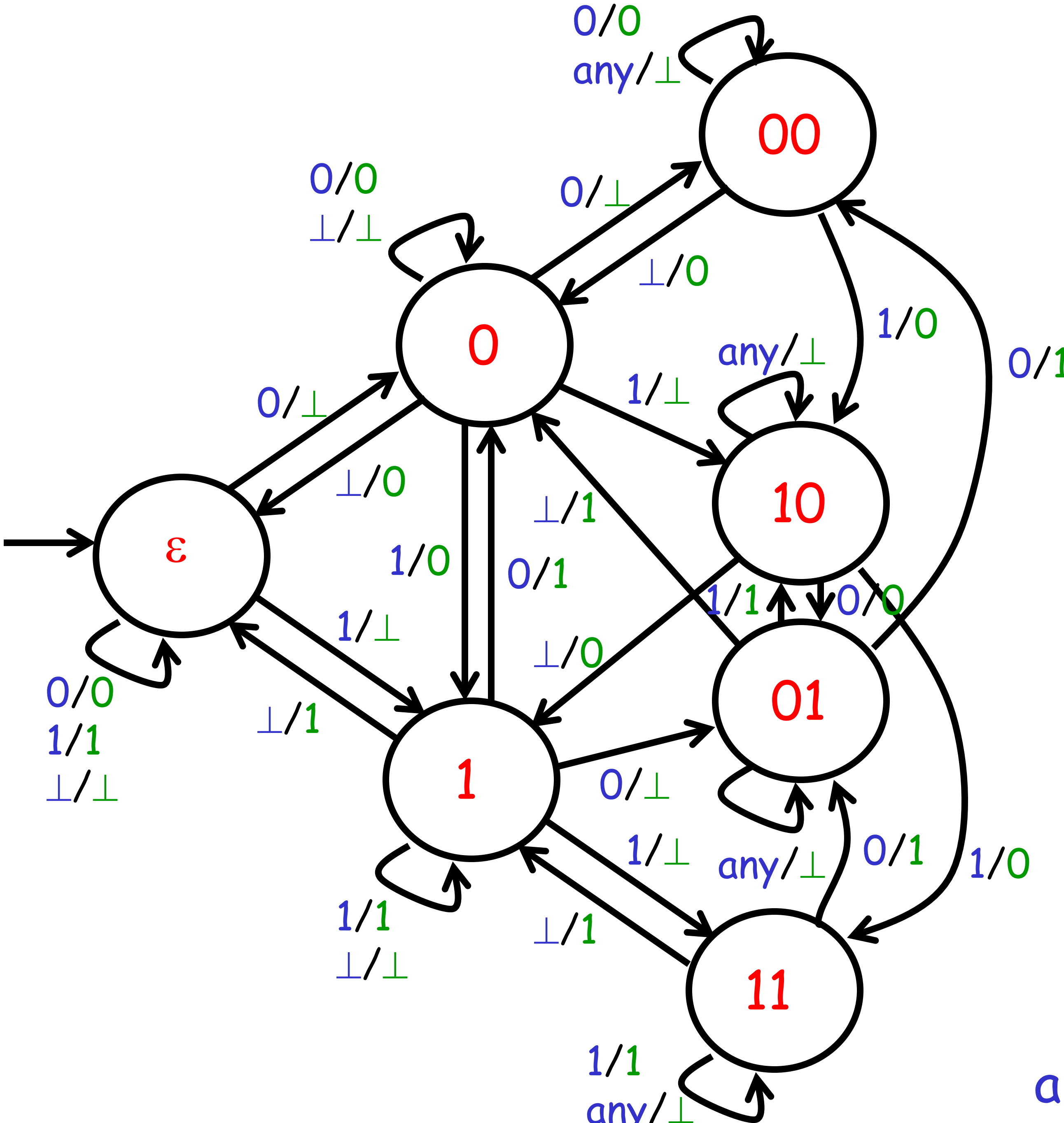
Time	0	1	2	3	4	5
Input	0	1	⊥	0	⊥	⊥
Output	⊥	⊥	⊥	0	⊥	1

One possible run

Time		0		1		2		3		4		5	
Input		0		1		$\perp$		0		$\perp$		$\perp$	
Output		$\perp$		$\perp$		$\perp$		0		$\perp$		1	
State	ε		0		10		10		01		01		0

Another possible run on the same input signal

Input		0		1		$\perp$		0		$\perp$		$\perp$	
Output		$\perp$		$\perp$		0		1		$\perp$		0	
State	ε		0		10		1		0		0		ε

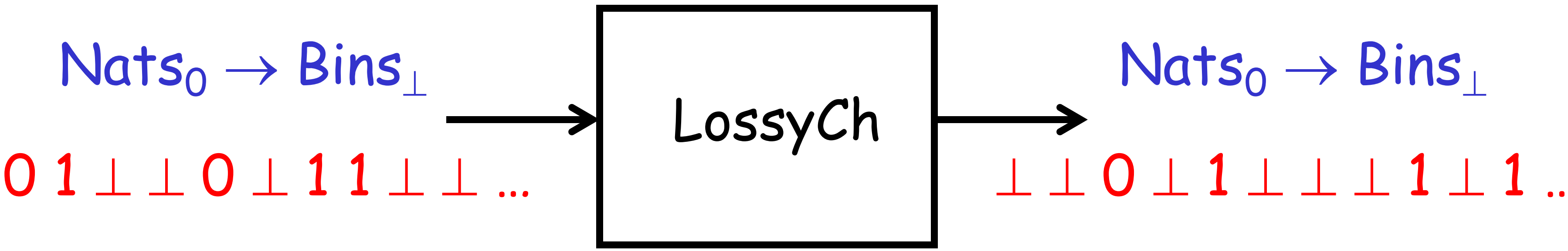


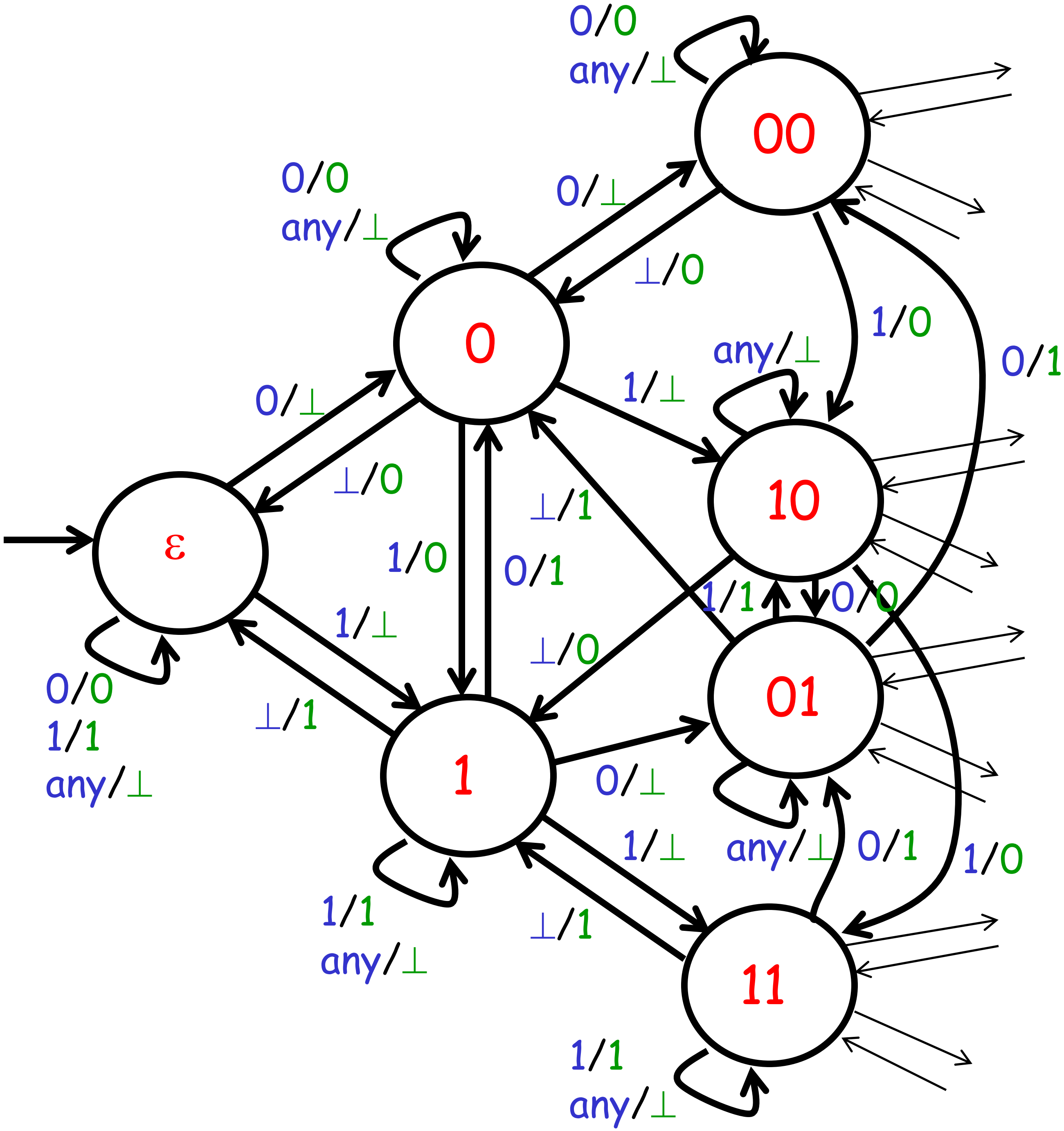
Finite state !

Buffer of size 2

any = { 0, 1, ⊥ }

Modeling Uncertainty: Lossy Channel





ε
0
1
00
10
01
11
0/0
any/⊥
0/⊥
⊥/0
1/⊥
⊥/1
1/0
0/1
1/1
0/0
1/1
any/⊥
0/0
1/1
any/⊥
0/1
1/0
any/⊥

One possible run

Time 0 1 2 3 4 5

Input 0 1 $\perp$ 0 $\perp$ $\perp$

Output $\perp$ $\perp$ $\perp$ 0 $\perp$ 0

State ε 0 0 0 0 0 ε

Another possible run on the same input signal

Input 0 1 $\perp$ 0 $\perp$ $\perp$

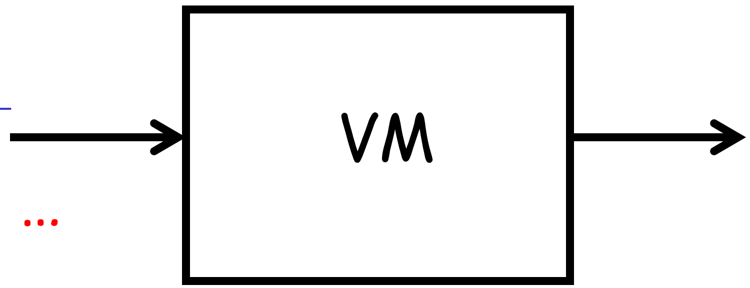
Output $\perp$ $\perp$ $\perp$ $\perp$ 0 $\perp$

State ε ε ε ε 0 ε ε

Modeling Properties: Vending Machine

$Nats_0 \rightarrow Select_\perp$

$C \perp \perp D \perp C \perp \perp \ldots$



$Nats_0 \rightarrow Dispense_\perp$

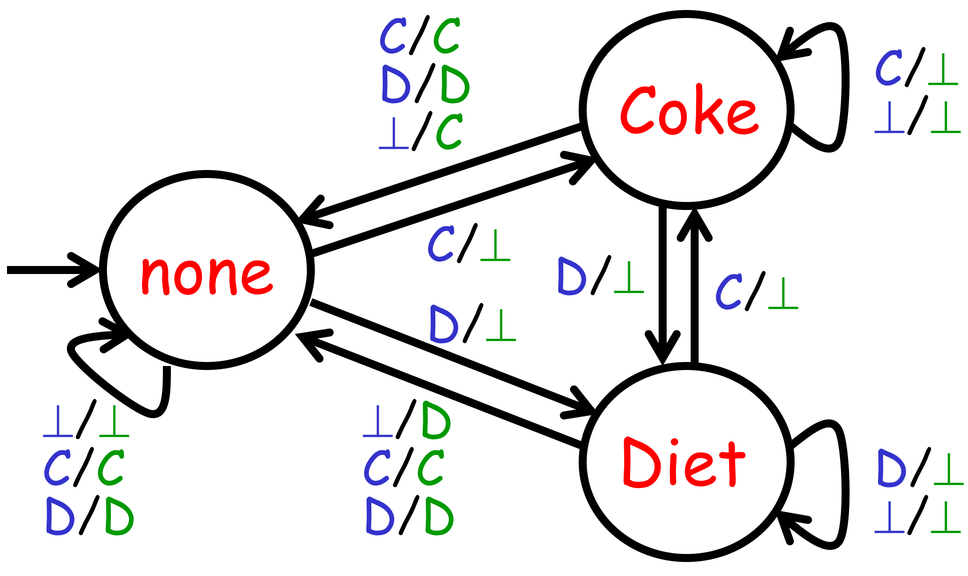
$\perp \perp C \perp \perp \perp \perp C \ldots$

Select = Dispense = { Coke, Diet }

“No-unrequested-soda” property:

Whenever the machine dispenses Coke, then the most recent request was for Coke;

whenever the machine dispenses Diet, then the most recent request was for Diet.



One possible run

Time		0		1		2		3		4		5	
Input		C		$\perp$		$\perp$		$\perp$		D		$\perp$	
Output		$\perp$		$\perp$		$\perp$		C		$\perp$		D	
State	n		C		C		C		n		D		n

Another possible run on the same input signal

Input		C		$\perp$		$\perp$		$\perp$		D		$\perp$	
Output		$\perp$		$\perp$		$\perp$		$\perp$		$\perp$		D	
State	n		C		C		C		C		D		n

Deterministic Reactive System:

for every input signal, there is exactly one output signal.

Nondeterministic Reactive System:

for every input signal, there is one or more output signals.

Deterministic Reactive System : function

$\text{DetSys} : [\,\text{Time} \to \text{Inputs}\,] \to [\,\text{Time} \to \text{Outputs}\,]$

Nondeterministic Reactive System : relation

$\text{NondetSys} \subseteq [\,\text{Time} \to \text{Inputs}\,] \times [\,\text{Time} \to \text{Outputs}\,]$

such that $\forall\, x \in [\,\text{Time} \to \text{Inputs}\,],$
$\exists\, y \in [\,\text{Time} \to \text{Outputs}\,],\ (x,y) \in \text{NondetSys}$

Every pair $(x,y) \in NondetSys$ is called a

behavior

of the nondeterministic reactive system NondetSys .

S1 is a more detailed description of S2;

S2 is an abstraction or property of S1.

System S1 refines system S2

iff

1. Time [S1] = Time [S2] ,
2. Inputs [S1] = Inputs [S2] ,
3. Outputs [S1] = Outputs [S2] ,
4. Behaviors [S1] $\subseteq$ Behaviors [S2] .

S1	refines	S2
Buffer of size 2		Arbitrary channel
Vending machine		No-unrequested-soda property
Fair coin		Nondeterministic coin

No output signal heads, heads, heads, heads, …
or tails, tails, tails, tails, …

Systems S1 and S2 are equivalent

iff

1. Time [S1] = Time [S2] ,
2. Inputs [S1] = Inputs [S2] ,
3. Outputs [S1] = Outputs [S2] ,
4. Behaviors [S1] = Behaviors [S2] .

Deterministic causal discrete-time reactive systems
can be implemented by
(deterministic) state machines.

Nondeterministic causal discrete-time reactive systems
can be implemented by
nondeterministic state machines.

Deterministic State Machine

Inputs

Outputs

States

$\text{initialState} \in \text{States}$

$\text{update} : \ \text{States} \times \text{Inputs} \ \to \ \text{States} \times \text{Outputs}$

Nondeterministic State Machine

Inputs

Outputs

States

possibleInitialStates $\subseteq$ States

possibleUpdates :

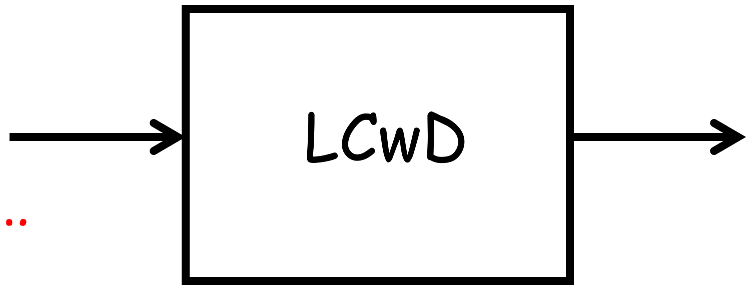
$$\text{States} \times \text{Inputs} \to P(\text{States} \times \text{Outputs}) \setminus \emptyset$$

receptiveness (i.e., machine cannot prohibit an input)

Lossy Channel without Delay

$\mathrm{Nats}_0 \to \mathrm{Bins}$

0 1 1 0 0 0 1 1 …



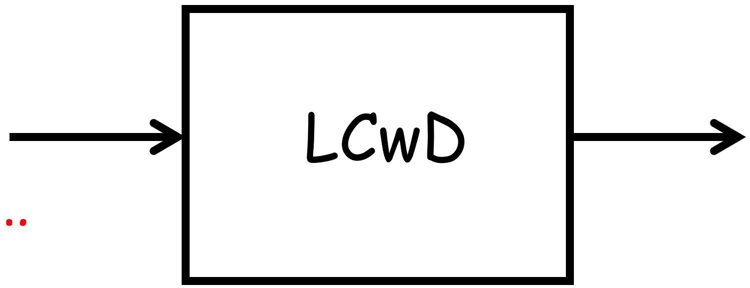
$\mathrm{Nats}_0 \to \mathrm{Bins}_\perp$

0 $\perp$ 1 0 $\perp$ $\perp$ 1 1 …

Lossy Channel without Delay

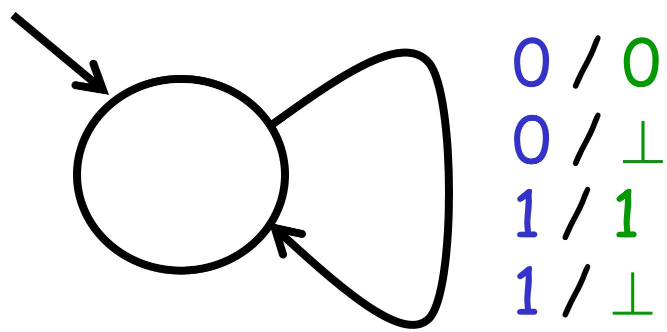
$Nats_0 \rightarrow Bins$

0 1 1 0 0 0 1 1 …



$Nats_0 \rightarrow Bins_\perp$

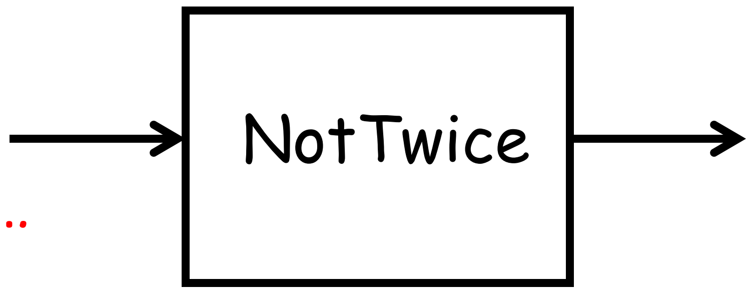
0 ⊥ 1 0 ⊥ ⊥ 1 1 …



Channel that never drops two in a row

$\mathrm{Nats}_0 \to \mathrm{Bins}$

$0\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ \ldots$



$\mathrm{Nats}_0 \to \mathrm{Bins}_\perp$

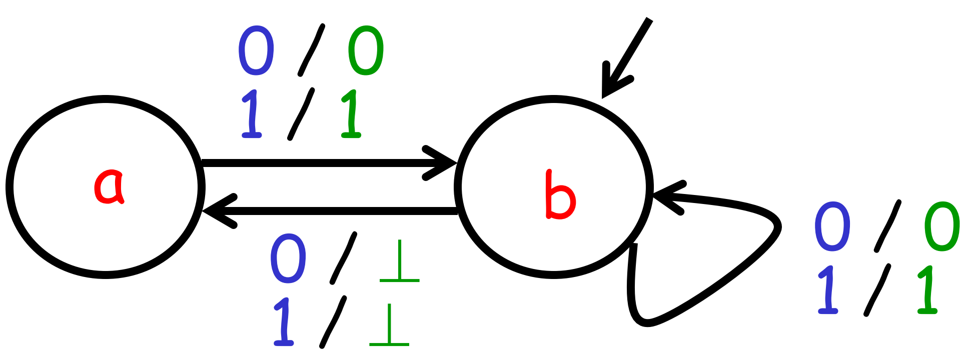
$0\ \perp\ 1\ 0\ \perp\ 0\ \perp\ 1\ \ldots$

Channel that never drops two in a row

State between time t-1 and time t :

a the input at time t-1 was dropped

b the input at time t-1 was not dropped, or t = 0



Channel that never drops two in a row

Inputs = $\{ 0, 1 \}$

Outputs = $\{ 0, 1, \perp \}$

States = $\{ a, b \}$

possibleInitialStates = $\{ b \}$

possibleUpdates $(a, 0) = \{ (b, 0) \}$
possibleUpdates $(a, 1) = \{ (b, 1) \}$
possibleUpdates $(b, 0) = \{ (b, 0), (a, \perp) \}$
possibleUpdates $(b, 1) = \{ (b, 1), (a, \perp) \}$

Deterministic state machine:

for every input stream, there is exactly one run.

Nondeterministic state machine:

for every input stream, there is one or more runs.

Every run generates an output stream, and therefore every run gives rise to a behavior.