CS 5: The Home Stretch

All of *my* eyes are on the lookout!

CS Practice

Final Projects

TextGame
Picobot
vPython
Life+1
TextID

CS5: Introduction to Computer Science
CS5 Web > WebHome
Submissions: CS submission site

CS 5: *Welcome!*

Administration | Using Python | Class Resources | Exams & Projects | Related Courses

- Gold Midterm Review
- Black Midterm Review
- Final Projects
- Final Review (Gold)
- Final Review (Black)

Homework Assignments

... and CS5's **<u>final</u>**es ...

Week 3
Week 8
Week 12

Want CS 60? Don't wait to PERM it!

The "spherical cow" strategy: for physics *and* CS!

And especially vPython!

Keenan Crane

Start with the

MOST BORING

version of your design

Ex: make your game with only spheres!

Once that works, you can add detail *iteratively*

Make it "work", then make it better!

What's left to do…

Wednesday, **4/17**: Final project "starter" (see page)

`starter.txt` – team member names, project goals

`starter.py` – initial progress with initial steps complete (check out the website for what to do)

Tuesday, **4/23**: hw12pr0 – reading hw12pr1 – jsFLAP

Final project "milestone" (see page)

`milestone.txt` – reflection on how work has gone so far

`milestone.py` – more progress with 6-7 functions complete (check out the website for what to do)

Want CS 60? Don't wait to PERM it!

What's left to do…

Wednesday, **4/17**: Final project "starter" (see page)

Tuesday, **4/23**: hw12pr0 – reading hw12pr1– jsFLAP

Final project "milestone" (see page)

Friday, **4/26**: Final project (*final version*)

Thus. **5/9** @ 2pm Final **exam** ~ similar to midt.

Remaining Labs are *optional*

no "signing in" - no lab problems

work on **projects**:
start/milestone/final

and/or work on hw 12's
finite-state machines

we **won't** be able to get you graded feedback on the milestone
before the final project is due – ***so join us for lab!***

AI Wrap Up

Two kinds…

Classical (search)

- Explore a search space
- Human written heuristics
- Explicitly programmed for task
- Understandable

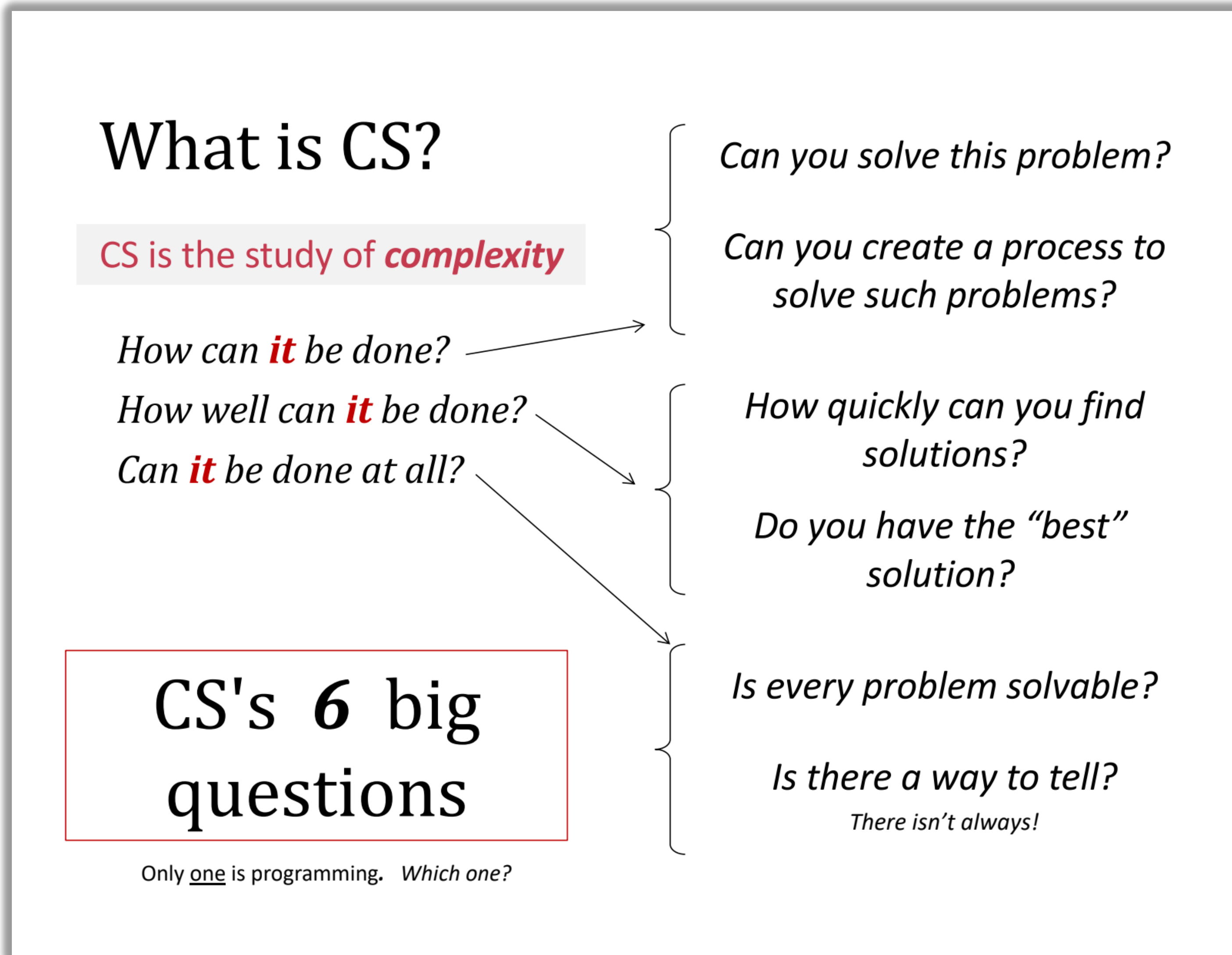
Connect 4

Machine Learning

- Learns from training data
- Simple parts (e.g., neurons in network)
- Less understandable

ChatGPT

Back on January 15…



CS != Programming

CS5's broad view:

Final Projects

TextGame
Life +1 Picobot
TextID vPython

CS Applications

- biological analysis + algorithms
- graphics media games ...
- search and the web
- robots and computer vision

CS 5

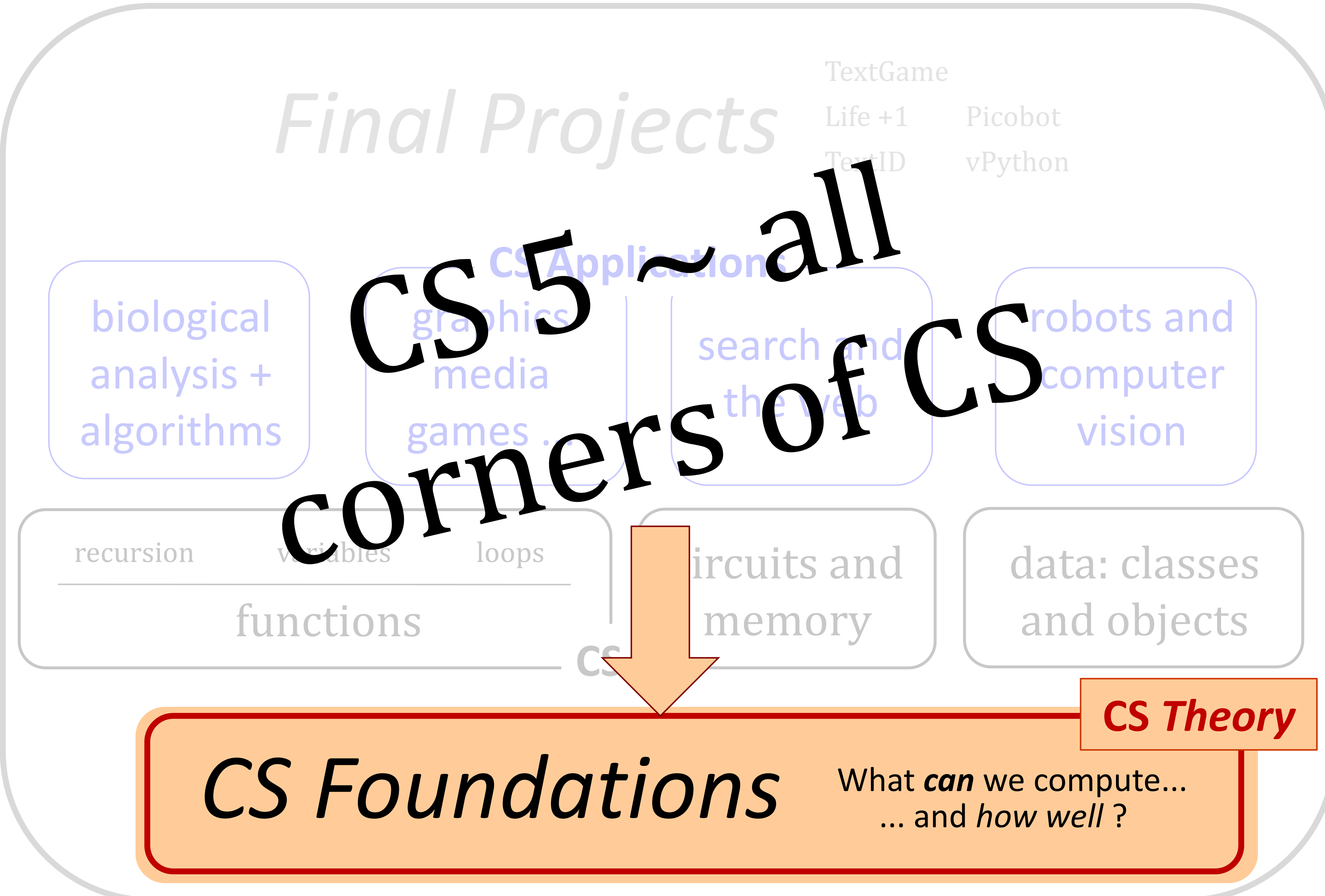
- recursion variables loops — functions
- circuits and memory
- data: classes and objects

CS Theory

CS Foundations

What ***can*** we compute...
... and *how well* ?

CS5's broad view:



Theory of Computation

(1) How do we define a "computer"?

(2) How do we "compute" with (1)?

(3) Given (1) and (2), what can we **provably** do (or not do?)

CS Foundations

What ***can*** we compute...
... and *how well* ?

A kind of computation?

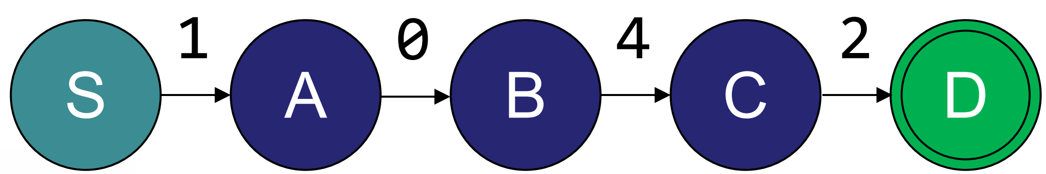
Takes an input:
a sequence of numbers + #

Only "accepts" some inputs:
either opens lock (correct)
or blinks red light (incorrect)

A kind of computation?

Code: 1042

Input: 1042



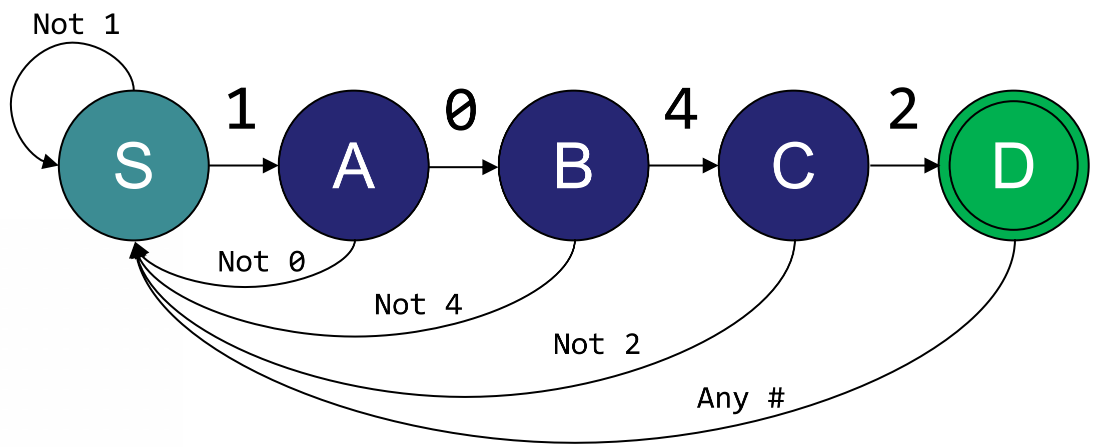
Takes an input:
a sequence of numbers + #

Only "accepts" some inputs:
either opens lock (correct)
or blinks red light (incorrect)

A kind of computation?

Code: 1042

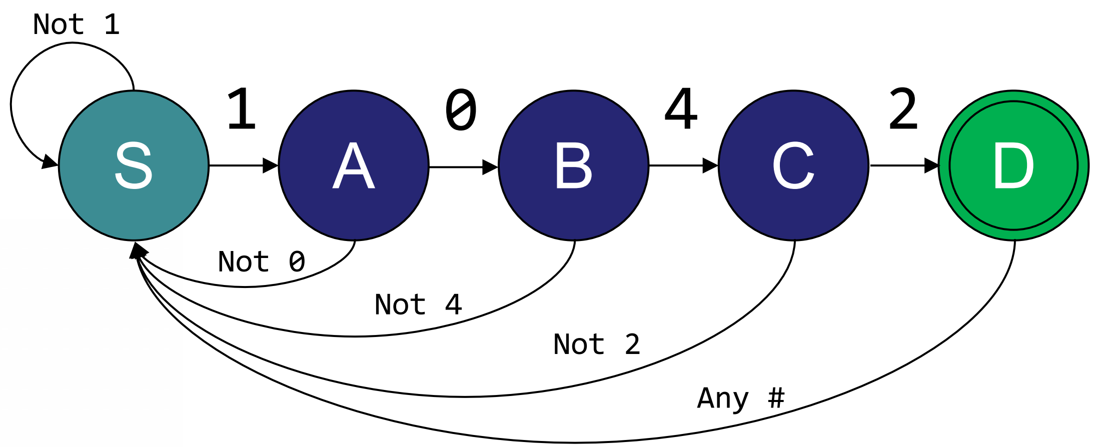
Input: 1038



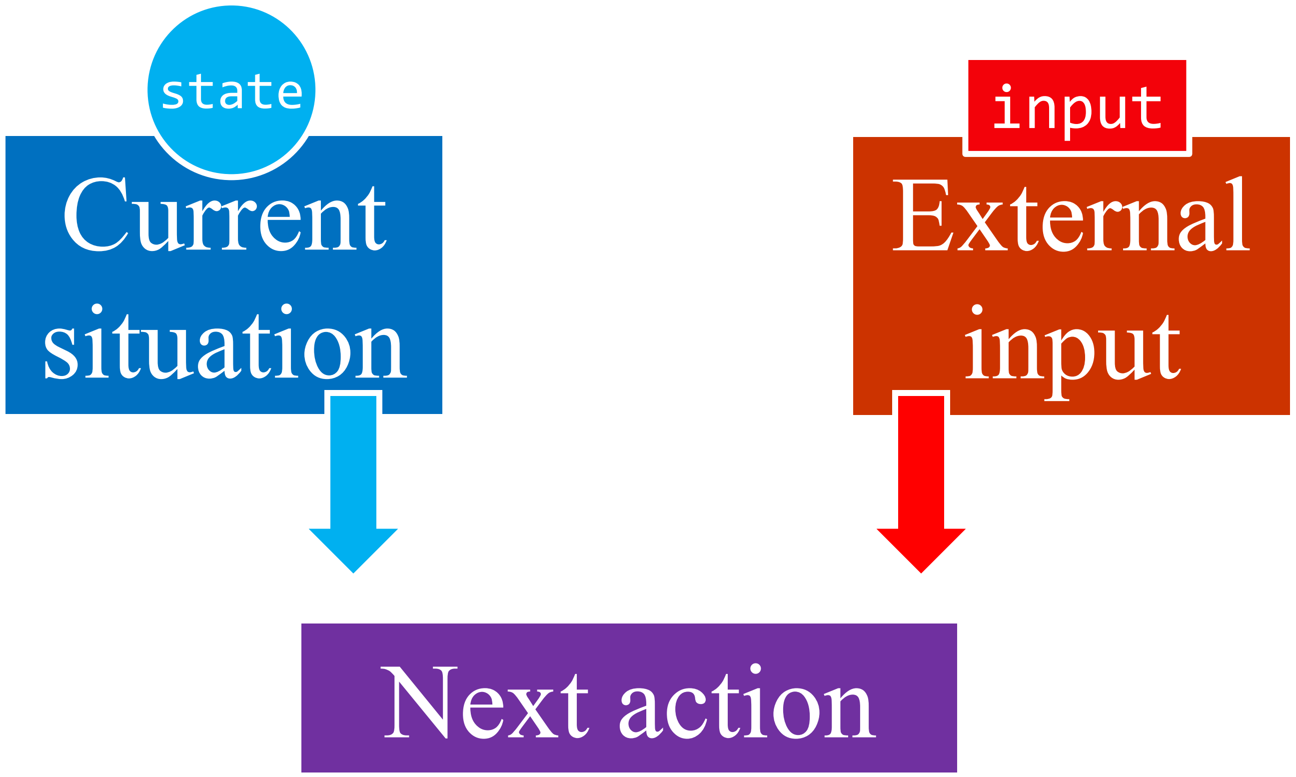
A kind of computation?

Input: 1031042

Code: 1042

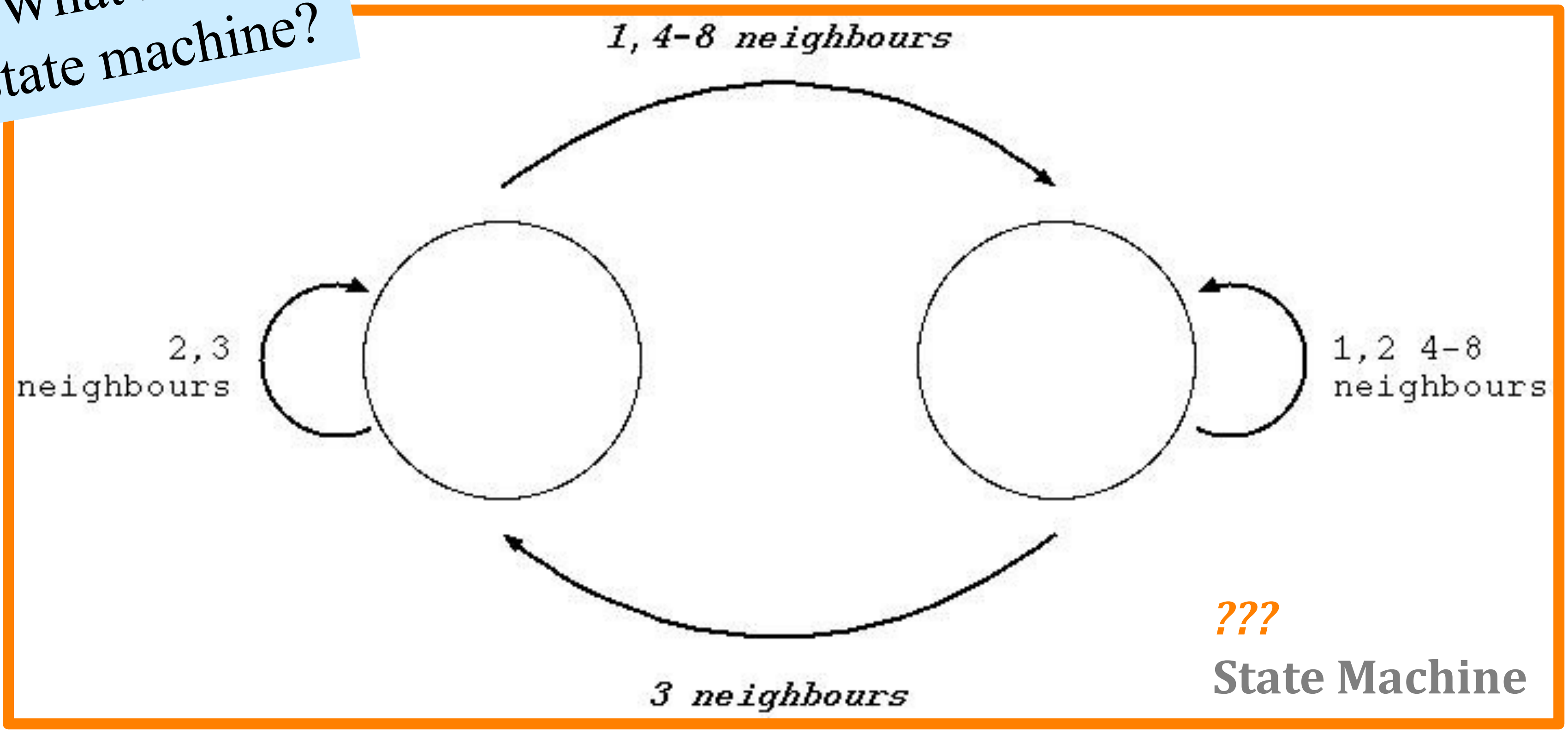


computers ~ ***state machines***



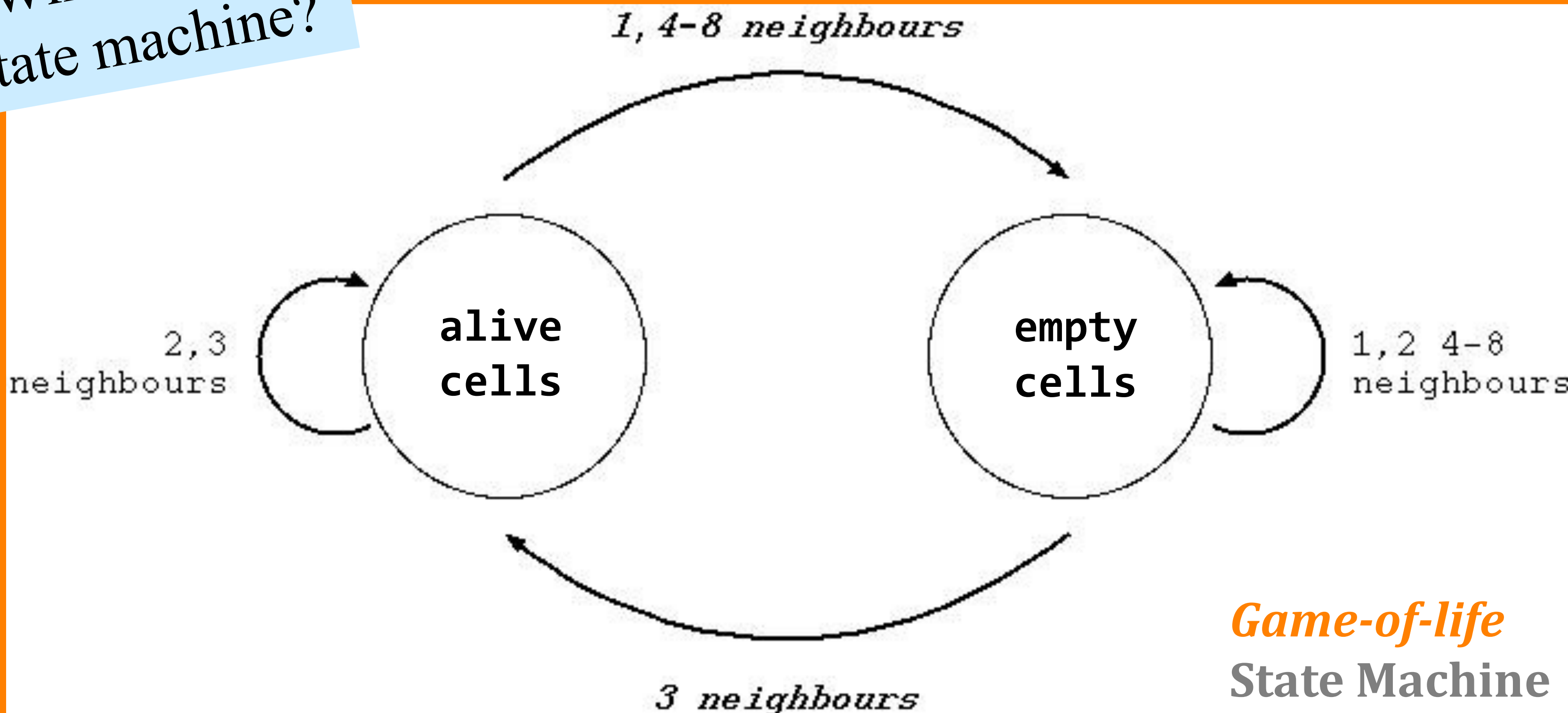
computers ~ ***state machines***

What is ***this*** state machine?



computers ~ ***state machines***

What is ***this*** state machine?



Game-of-life
State Machine

Unifying idea: **State**

```
0    NEWx   ->   S    1
```

previous state — external input — next step — next state

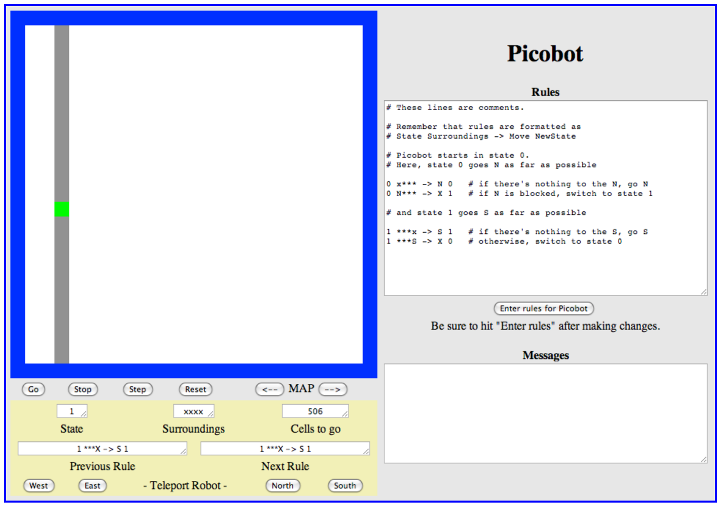
The ***state*** of a computation is

all the internal information

needed to take the next step

Picobot takes *"next step"* literally!

states help specify ***subtasks***



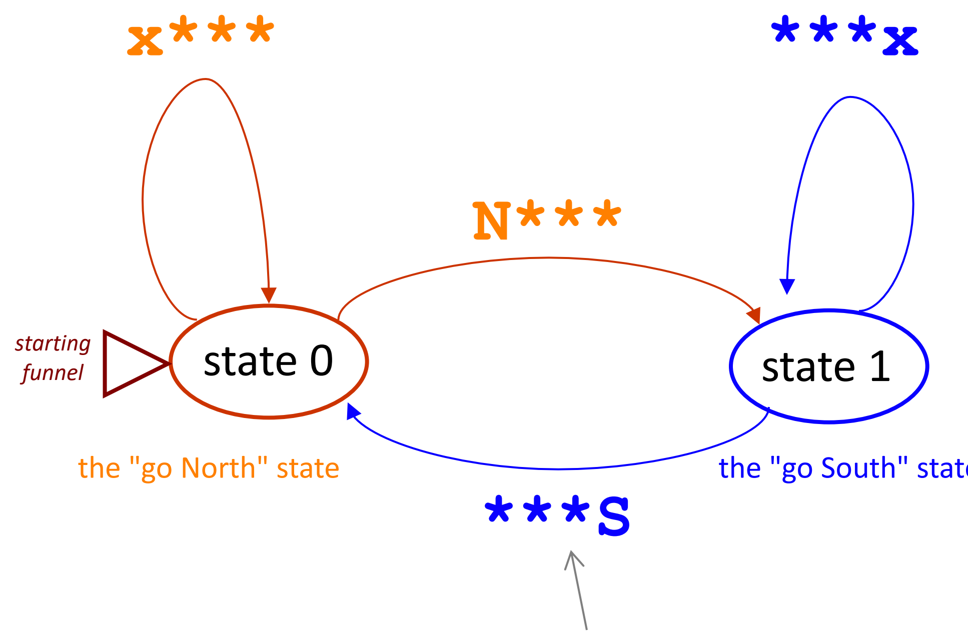
surroundings

state   pattern   ->   move   new state

```
0  x***  ->  N  0
0  N***  ->  X  1
1  ***x  ->  S  1
1  ***S  ->  X  0
```

State Machine:

each oval represents a different Picobot state



transitions move from state to state

Computation is a deliberate sequence of state changes

this doesn't seem very meaningful

```
| | | | | | | |
| | | | | | | |
|X| | | | | | |
|X| | | | | | |
|X| | | | | | |
|X|O|O|O| |O| |
---------------
 0 1 2 3 4 5 6
X's move: 4
```

```
| | | | | | | |
| | | | | | | |
|X| | | | | | |
|X| | | | | | |
|X| | | | | | |
|X|O|O|O|X|O| |
---------------
 0 1 2 3 4 5 6
O's move:
```

```
00 read   r1
01 setn   r2 1
02 jeqzn  r1 08
03 mul    r2 r2 r1
04 addn   r1 -1
05 jumpn  02
06 nop
07 nop
08 write  r2
09 halt
```

```
10101001011        10101001011
00000000000        00000001110
```

Computation is a deliberate sequence of state changes

this doesn't seem very *meaningful*

Computers ~ **F**inite **S**tate **M**achines

```
| | | | | | | |
| | | | | | | |
|X| | | | | | |
|X|
|X|
|X|
---
 0 1
X's
```

```
| | | | | | | |
| | | | | | | |
|X|
```

```
05 jumpn  02
06 nop
07 nop
08 write  r2
09 halt
```

```
10101001011  →  10101001011
00000000000     00000001110
```

Computers ~ Finite State Machines

What if we just assume we have binary strings as inputs?

OK! *All data can be written in binary!*

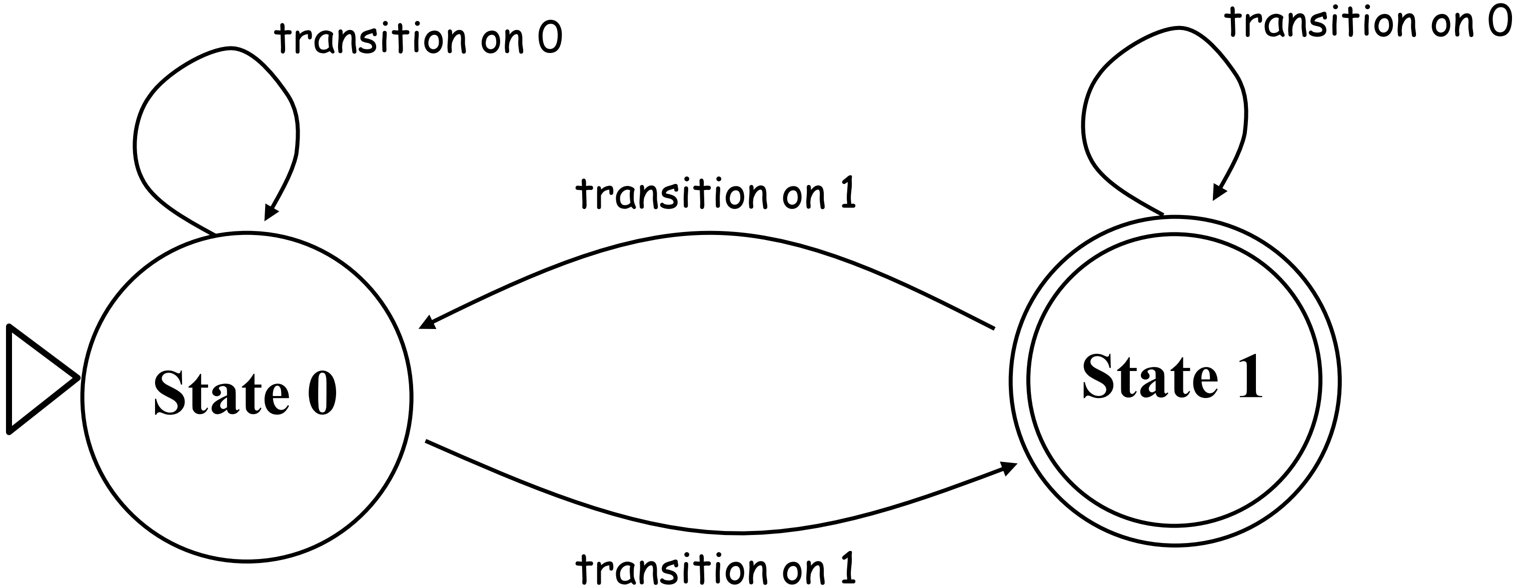
What if we just assume we only get one input?

OK! *We can concatenate our inputs into one!*

What if we're only allowed to have one Boolean output?

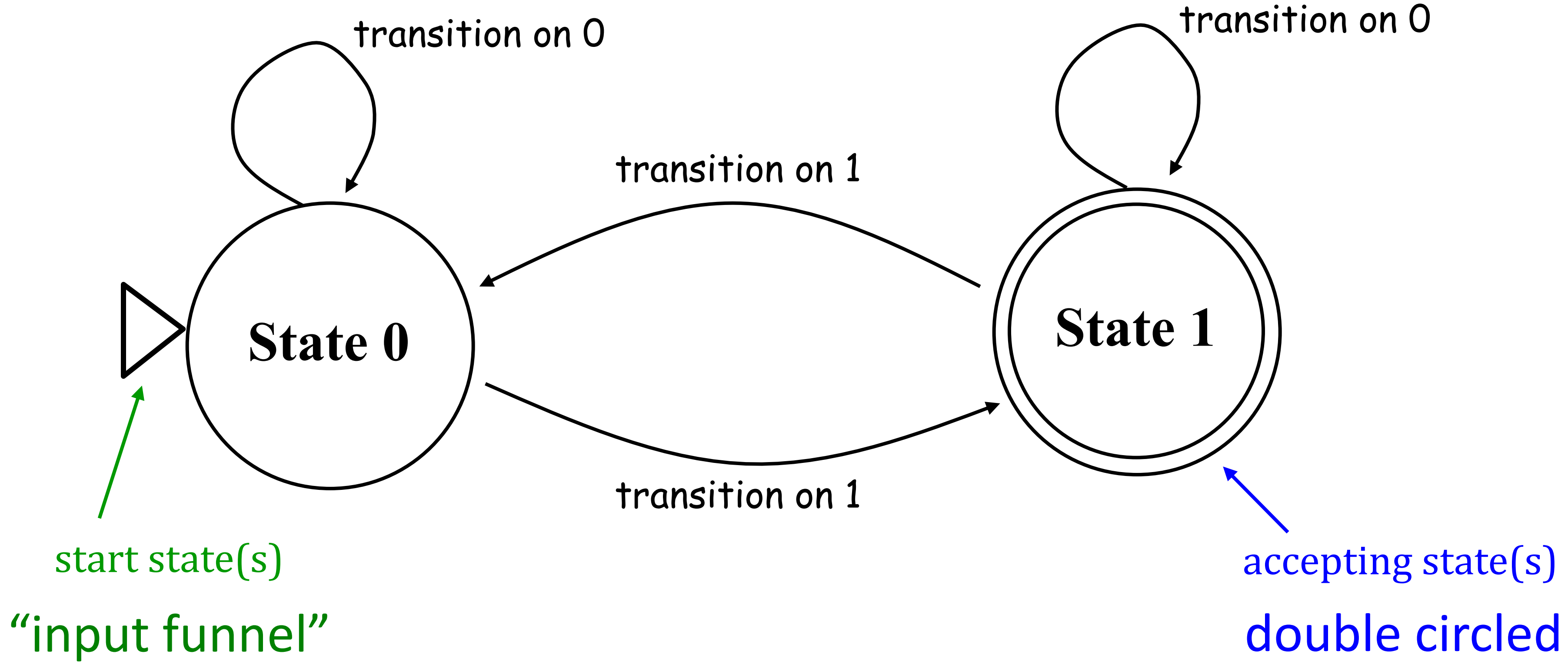
OK! *Like with circuits, we can use multiple FSMs to handle bigger outputs.*

Finite state machine

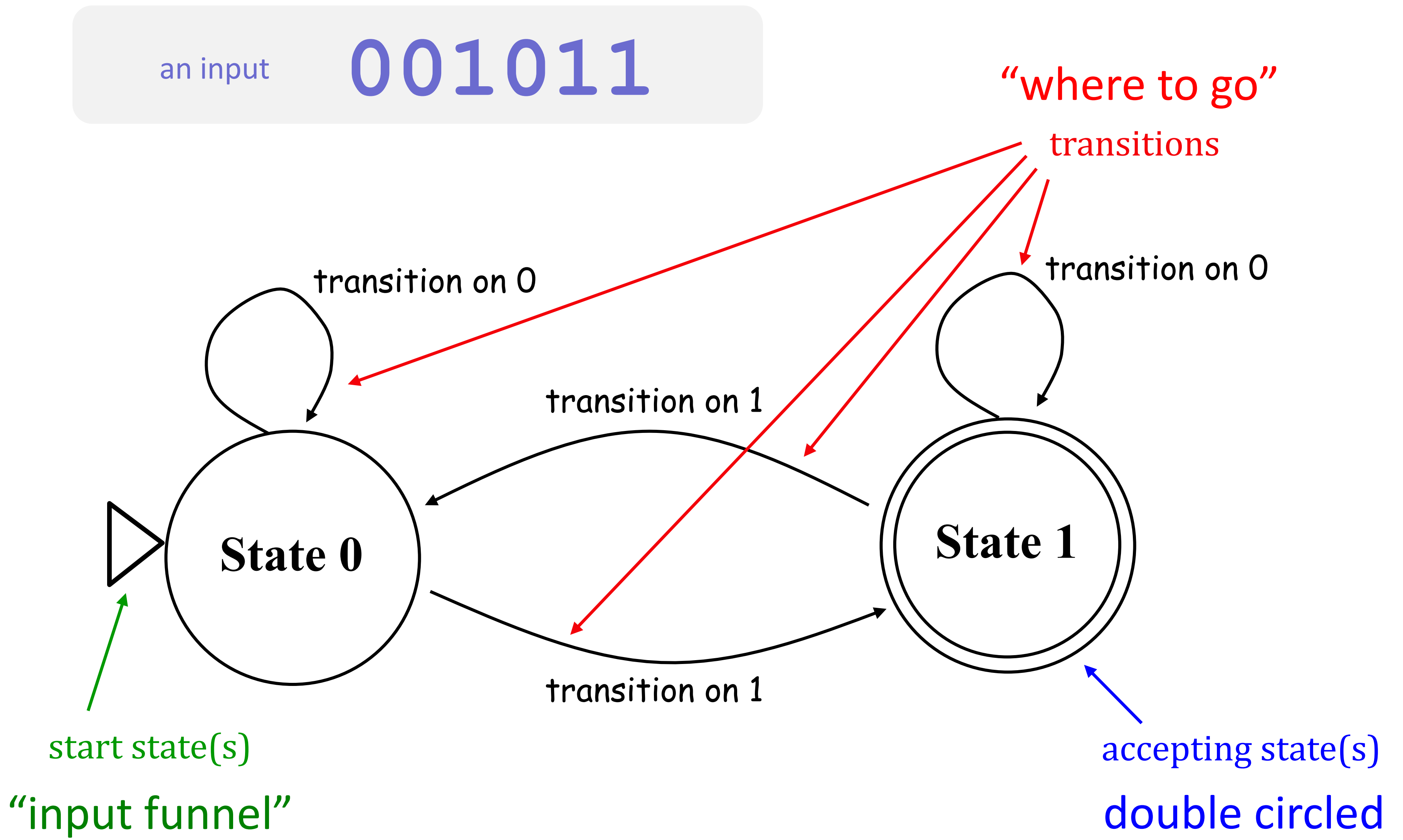


Finite state machine

an input 001011



Finite state machine



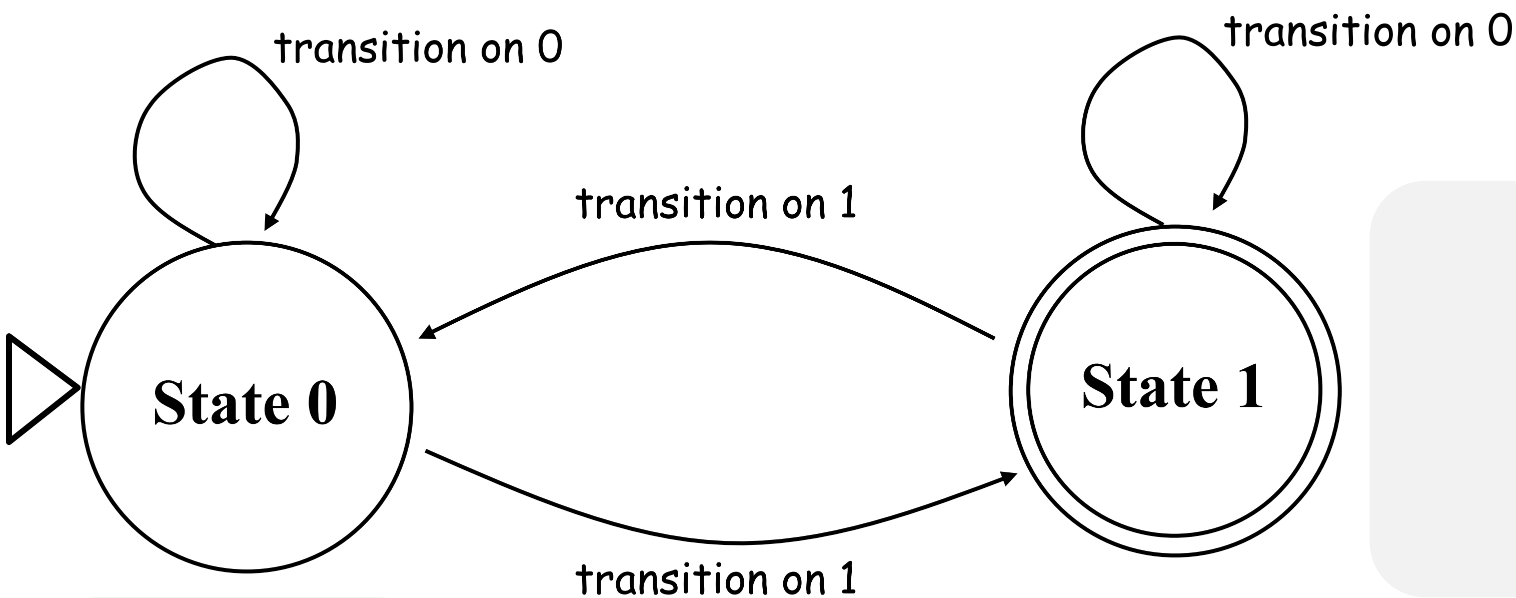
FSM: *Finite state machine*

an input sequence

always left-to-right

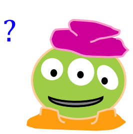
001011

output for this input



What does each state MEAN ?

What does this FSM do overall?



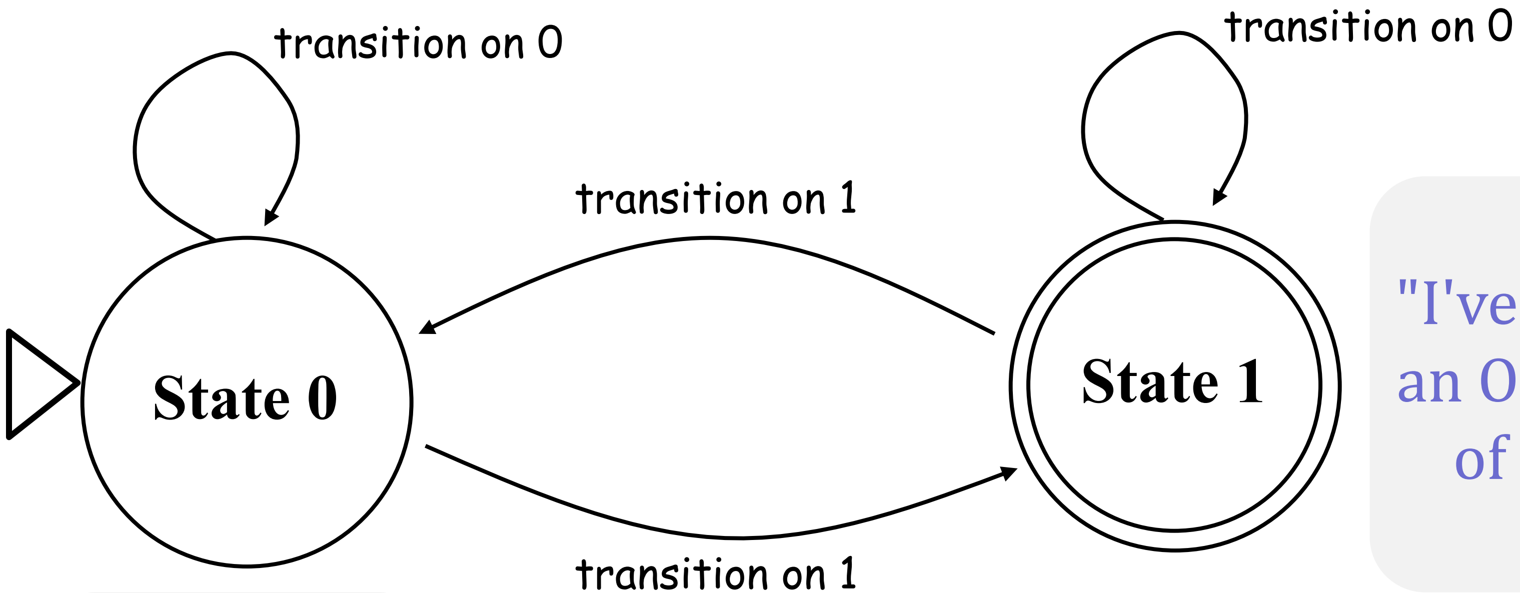
FSM: *Finite state machine*

an input sequence
always left-to-right

001011

output for this input

accepted!

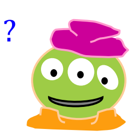


"I've seen an ODD # of **1**'s

"I've seen an EVEN # of **1**'s

What does each state MEAN ?

What does this FSM do overall?



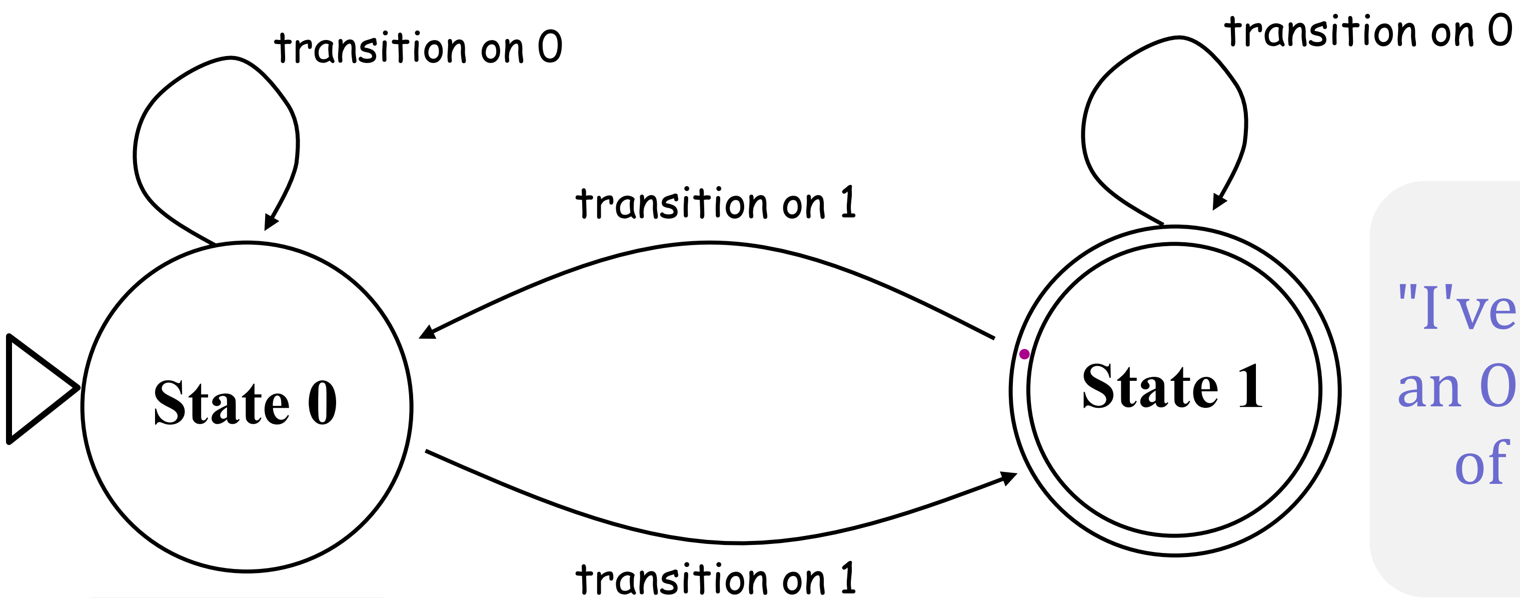
FSM: *Finite state machine*

an input sequence
always left-to-right

0010111

output for this input

rejected!

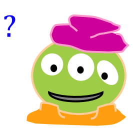


"I've seen an ODD # of **1**'s

"I've seen an EVEN # of **1**'s

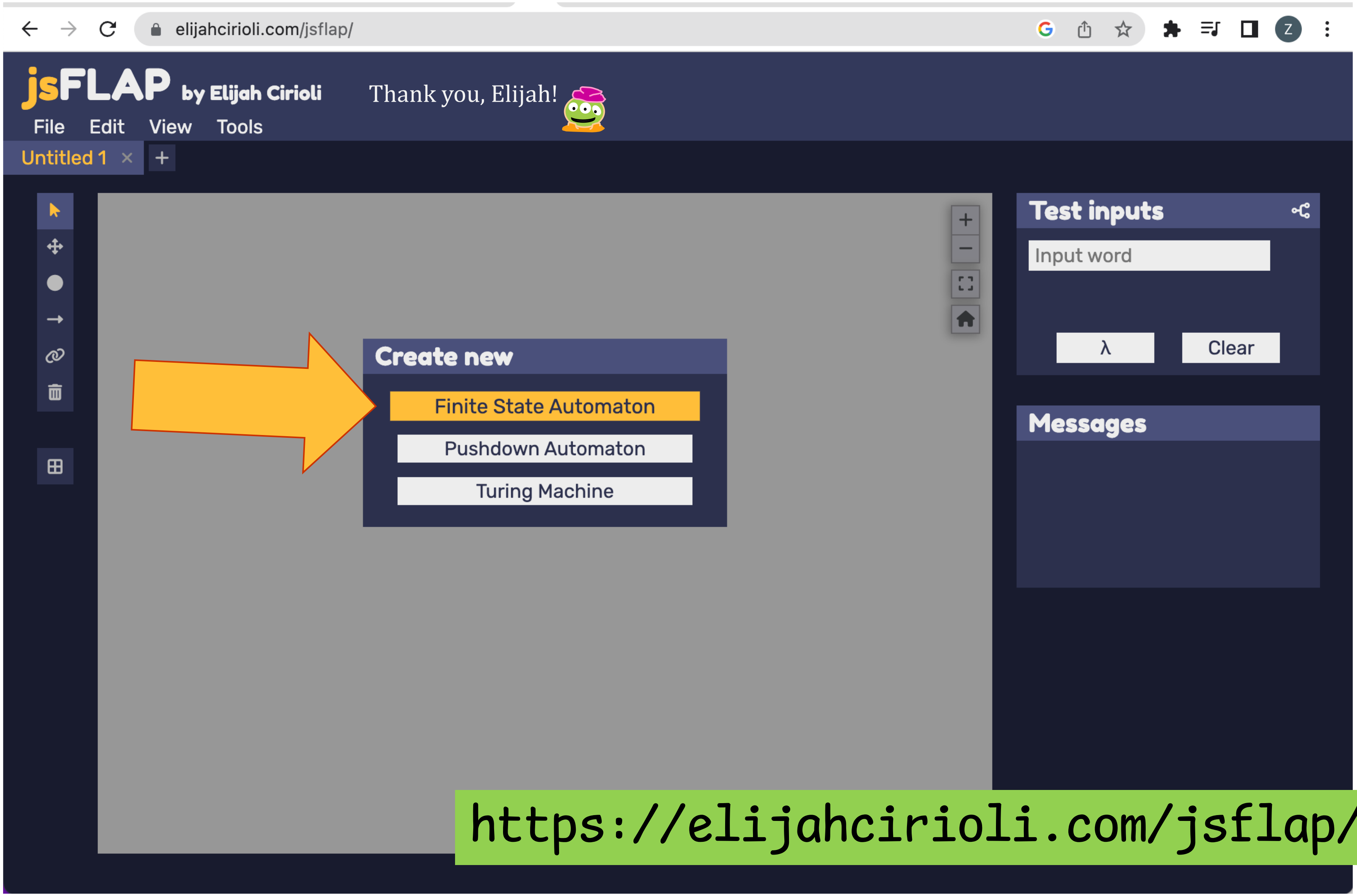
What does each state MEAN ?

What does this FSM do overall?



JSFLAP !

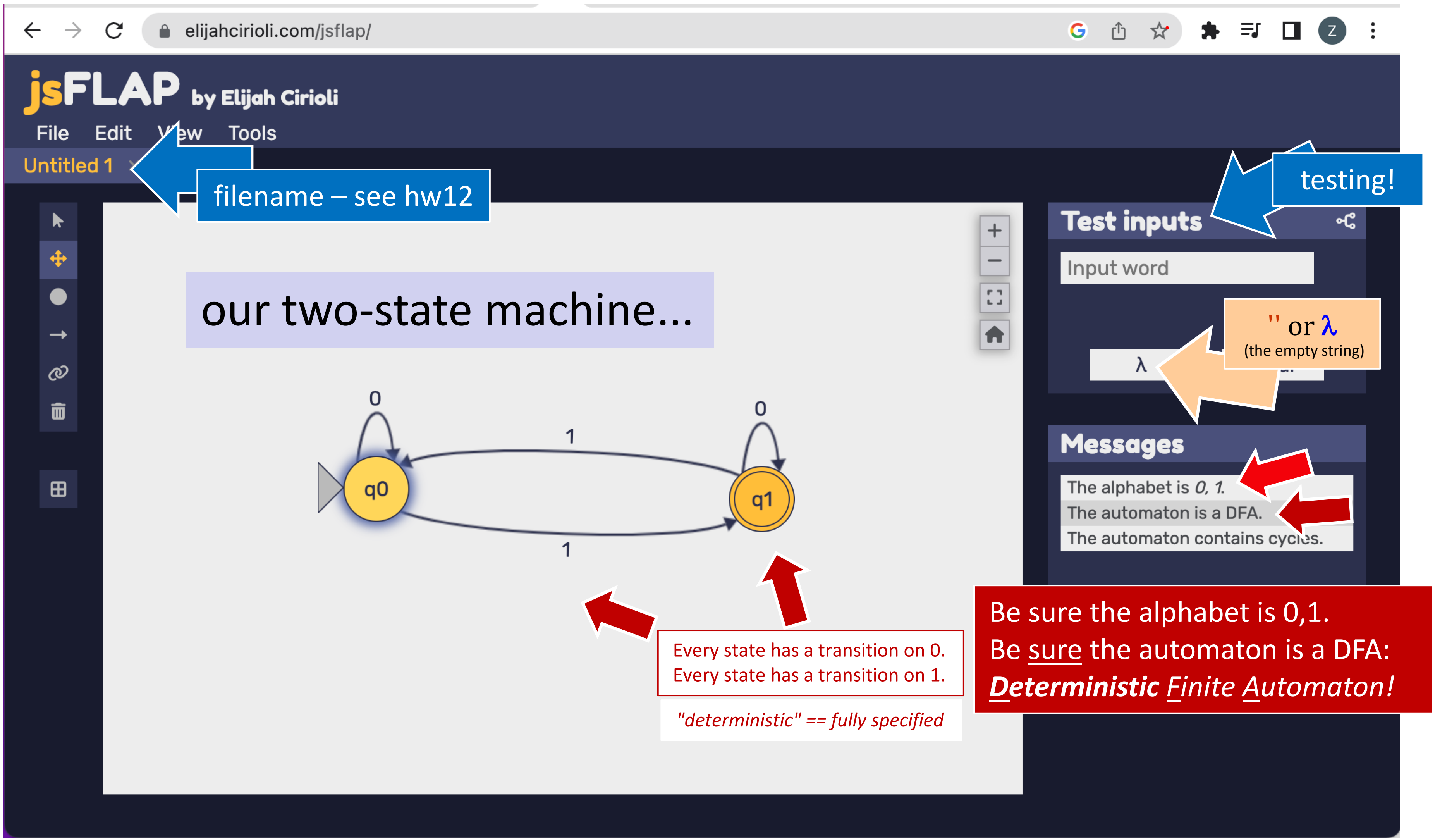
graphical state-machine **builder** for hw12



https://elijahcirioli.com/jsflap/

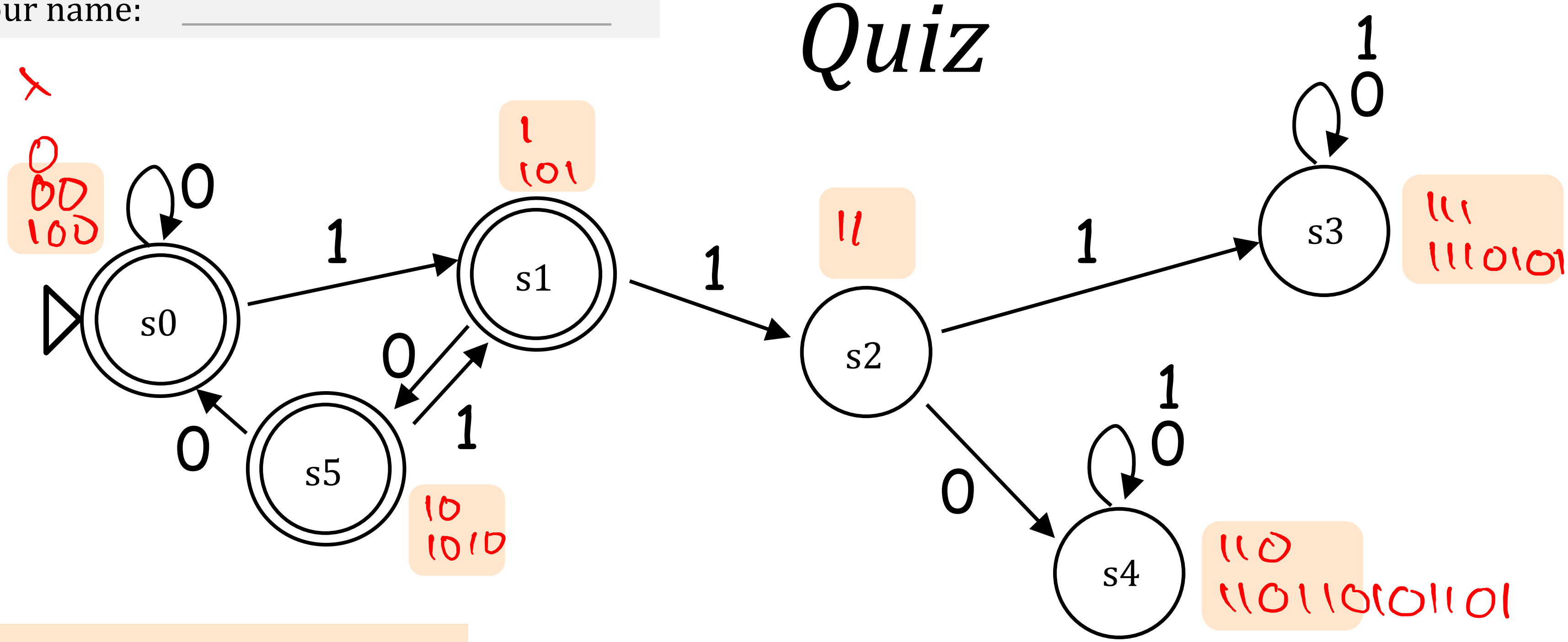
JSFLAP !

graphical state-machine **builder** for hw12



State your name: ______________________

Quiz



Label each state with 1-2 inputs that "land" there...

In general, what English phrase describes the ***rejected inputs***?

This machine rejects strings with ...

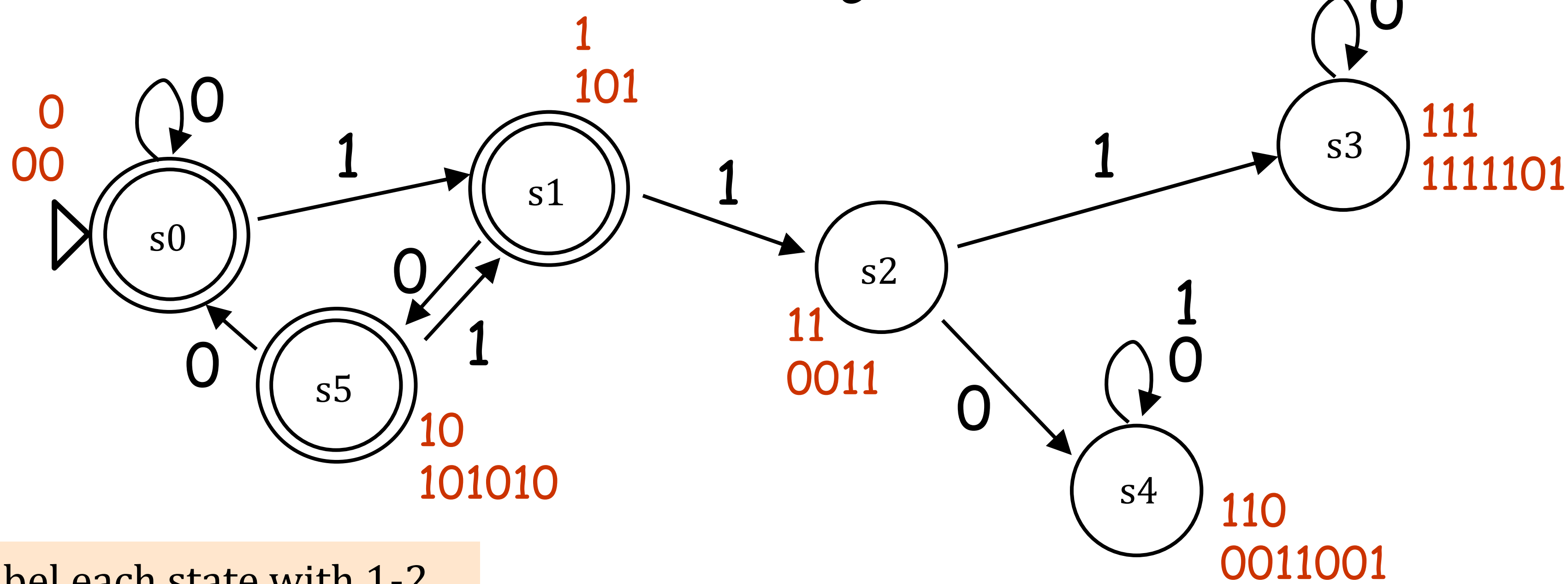
...11...

Extra Could *fewer* states produce the same accept-and-reject behavior here? What's the ***minimum*** #?

Hint: which strings ***have*** to be in separate states?

answers...

Quiz



Label each state with 1-2 inputs that "land" there...

In general, what English phrase describes the ***rejected inputs***?

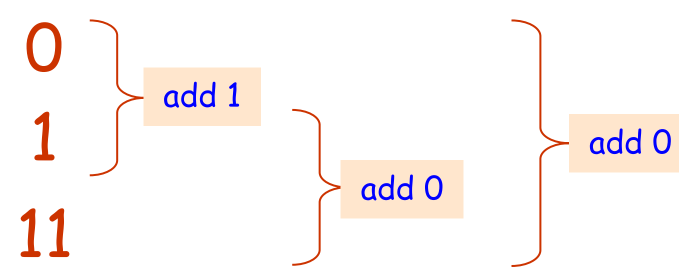
This machine rejects strings with ...

two 1's in a row (anywhere in the string)

Extra Could *fewer* states produce the same accept-and-reject behavior here? What's the ***minimum*** #?

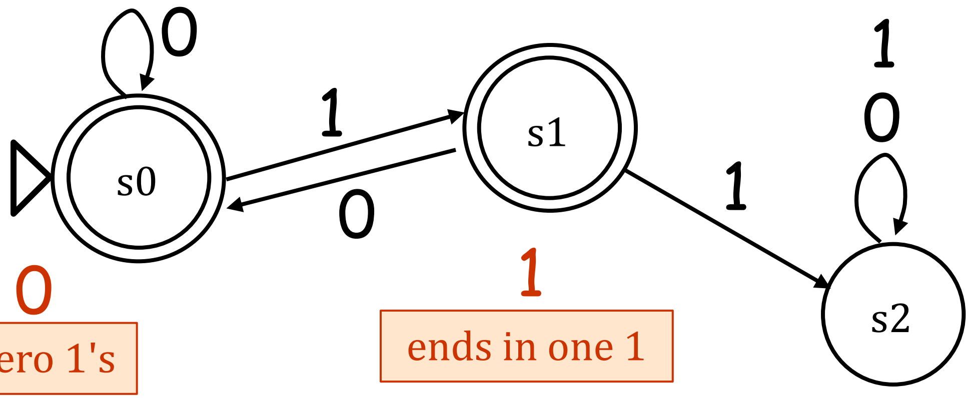
Hint: which strings ***have*** to be in separate states?

3 states min.



answers...

Quiz



0

ends in zero 1's

1

ends in one 1

"graveyard" state: be sure to have TWO transitions!

11

contains two (or more) 1's in a row

Label each state with 1-2 inputs that "land" there...

In general, what English phrase describes the ***rejected inputs***?

This machine rejects strings with ...

two 1's in a row (anywhere in the string)

Extra Could *fewer* states produce the same accept-and-reject behavior here? What's the ***minimum*** #?

3 states min.

Hint: which strings ***have*** to be in separate states?

0
1
11

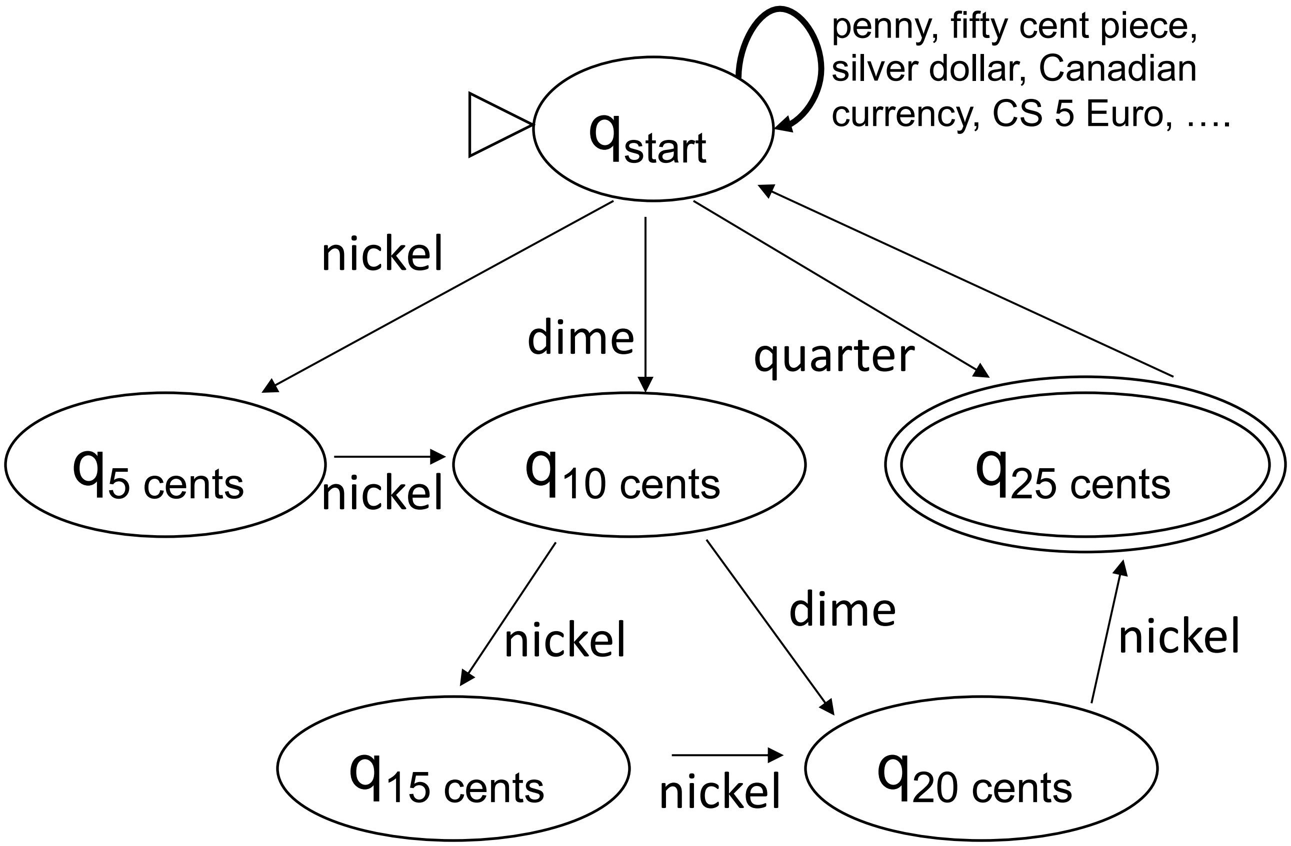
add 1

add 0

add 0

FSMs are everywhere!

mechanical vending machine

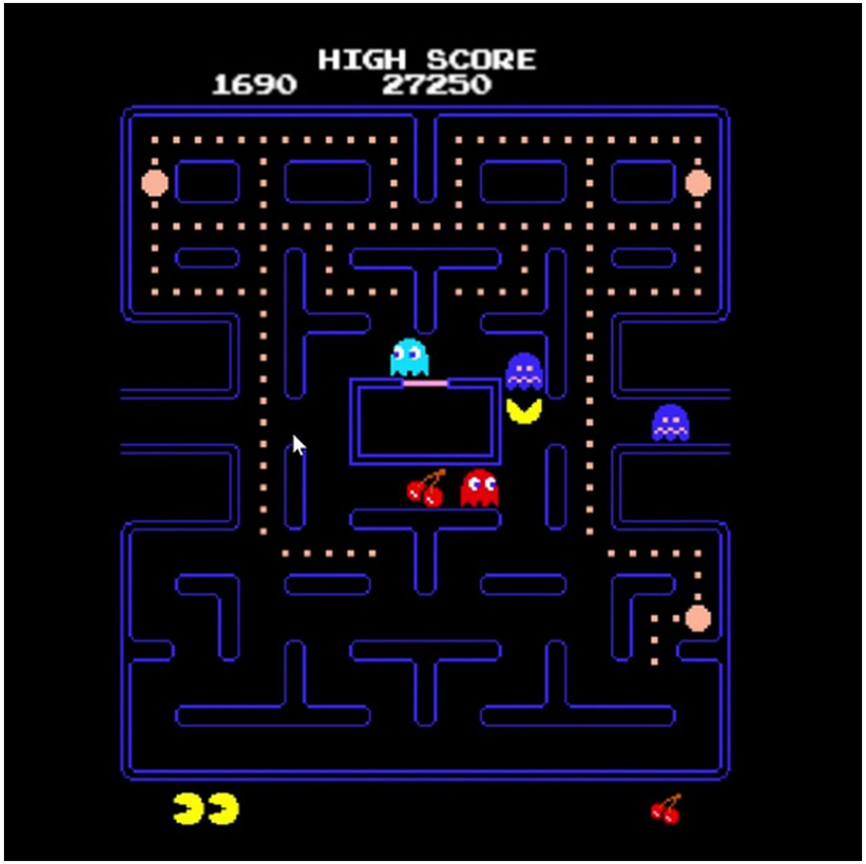


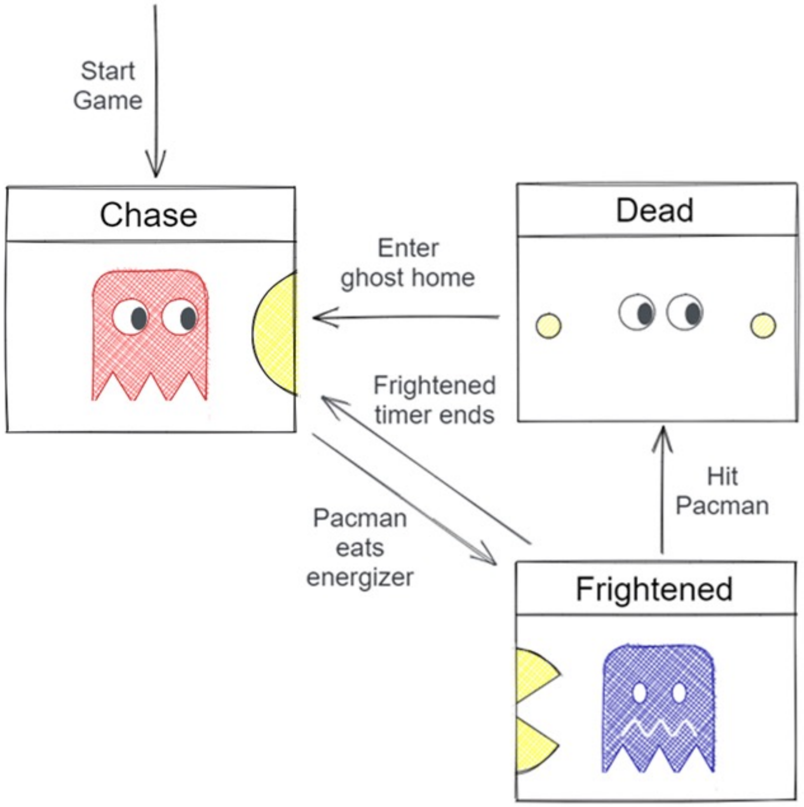
(some transitions not shown)

Mechanical state machines seem hard to change…

www.youtube.com/watch?v=85C4eh0mEJg @ 1:42 !

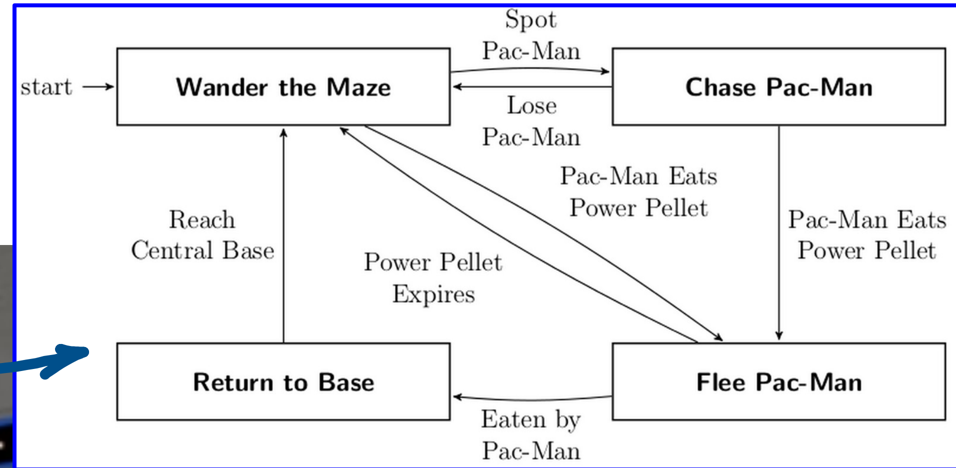
FSM ~ *Game AI*



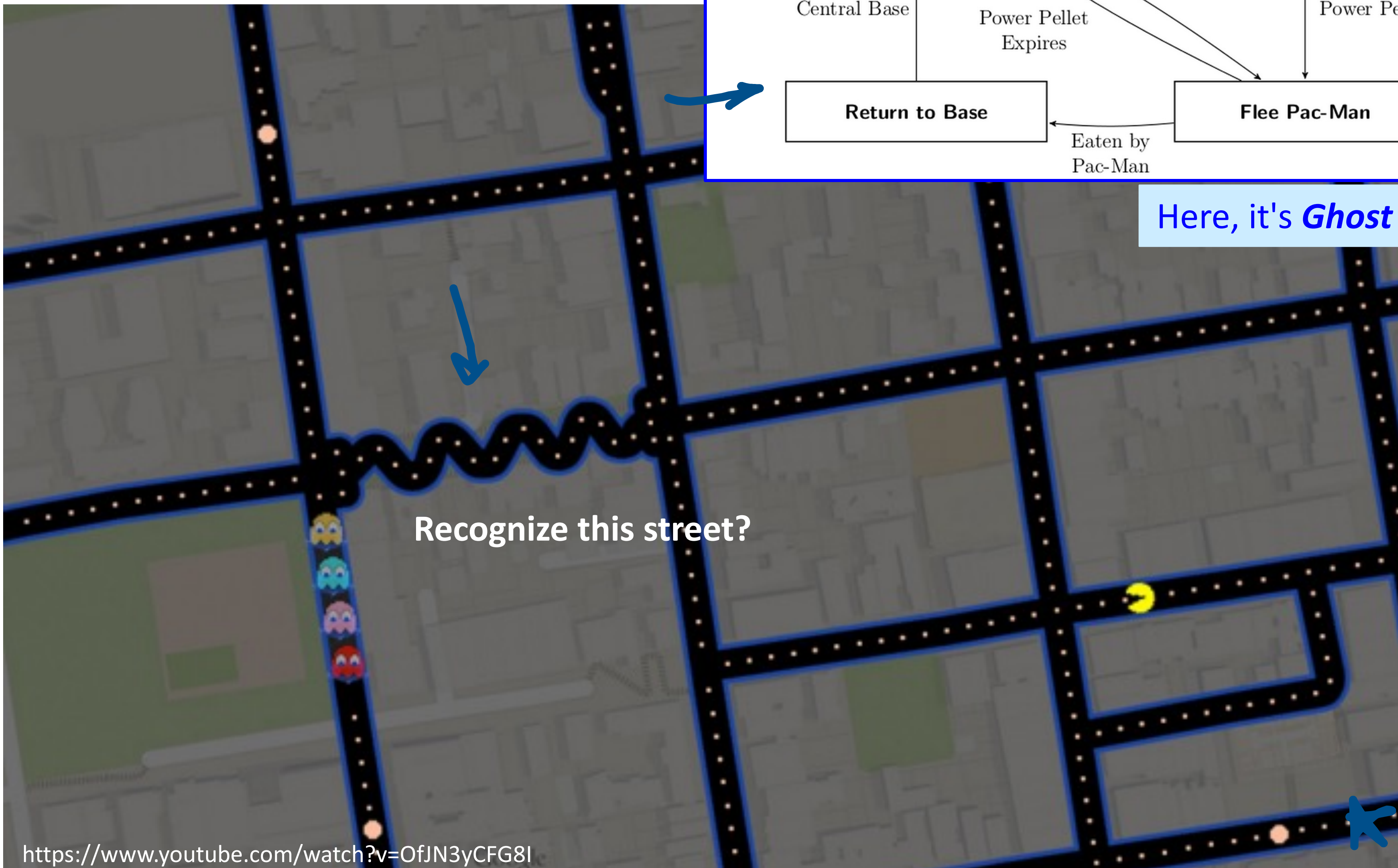


Matt Stobbs @ ScottLogic

FSM ~ *Game AI*



Here, it's ***Ghost AI***



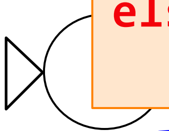
Build-your-own FSMs

2/10 Draw a FSM accepting strings with at least two **1**s (anywhere). Others are rejected.

Accepted examples: 0101, 00010110, 111011, 11
Rejected examples: 0100, 1000, 000000, 1, 0

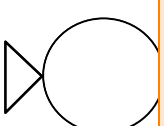
Hint - m
labels, t

```
if count('1',s) >= 2:
    return True
else:
    return False
```

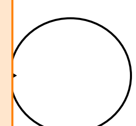


6/10 Draw a F ... don't contain the pattern **110** anywhere.

Accepted: 1010001, 011 **Rejected:** 10100<u>110</u>0, 0<u>110</u>1



```
if '110' not in s:
    return True
else:
    return False
```



Hint - there are FIVE more transitions – but no more states - needed here

4/10 Draw a FSM accepting strings in which the number of zeros (**0**s) is a multiple of 3, so there are 0, 3, 6, ... zeros. ***1**s don't matter!*

Accepted: 110101110, 11, 0000010
Rejected: 101, 0000, 111011101111

Hint: 1s never change the state!

Another hint: make a triangle!

```
if count('0',s)%3 == 0:
    return True
else:
    return False
```

needed?

FSMs ~ "software circuits"

8/10 ... in which the third digit (3d from the <u>left</u>) is a **1**.

Accepted: 10<u>1</u>0001, 01<u>1</u> **Rejected:** 11<u>0</u>00100, 11, 0

```
if s[2] == '1':
    return True
else:
    return False
```

What's the minim

```
if s[-3] == '1':
    return True
else:
    return False
```

10/10 ***Extra!*** Draw a FSM accepting strings whose third-to-last digit (3d from the **<u>right</u>**) is a **1**.

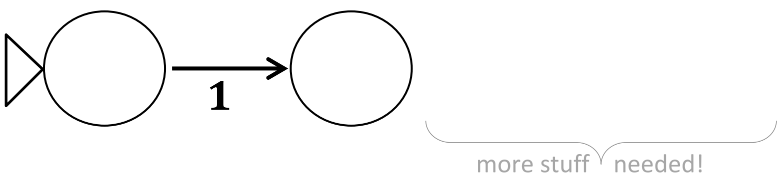
Acc: 0<u>1</u>00 and 01<u>1</u>01
Rej: 101<u>0</u>01 and 11

Build-your-own FSMs

2/10 Draw a FSM accepting strings with at least two **1**s (anywhere). Others are rejected.

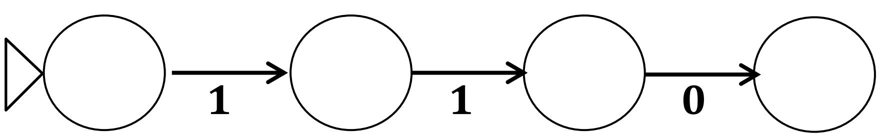
Accepted examples:	0101, 00010110, 111011, 11
Rejected examples:	0100, 1000, 000000, 1, 0

Hint - modify this starter FSM by adding labels, transitions, and one more state:



6/10 Draw a FSM that accepts strings that ***don't*** contain the pattern **110** anywhere.

Accepted: 1010001, 011 **Rejected:** 10100**110**0, 0**110**1



Hint - there are FIVE more transitions – but no more states - needed here

Draw a FSM accepting strings in which the number of zeros (**0**s) is a multiple of 3, so there are 0, 3, 6, ... zeros. ***1**s don't matter!* 4/10

Accepted:	110101110, 11, 0000010
Rejected:	101, 0000, 111011101111

Hint: 1s never change the state!

Another hint: make a triangle!

What's the minimum number of states needed?

Draw a FSM accepting strings in which the third digit (3d from the left) is a **1**. 8/10

Accepted: 10**1**0001, 01**1** **Rejected:** 11**0**00100, 11, 0

What's the minimum number of states needed?

10/10 ***Extra!*** Draw a FSM accepting strings whose third-to-last digit (3d from the **right**) is a **1**.

Acc: 0**1**00 and 01**1**01

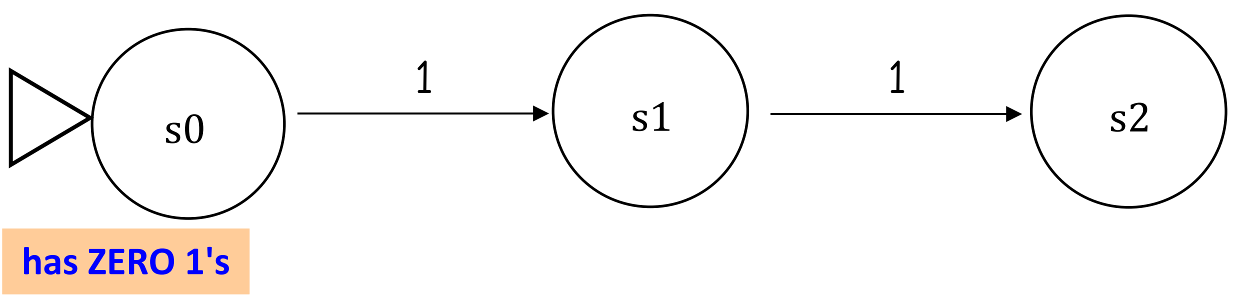
Rej: 101**0**01 and 11

Has at least two **1**s… ?

Draw a FSM accepting strings with at least two **1**s (anywhere). Others are rejected.

Accepted: 0101, 00010110, 111011, 11

Rejected: 0100, 1000, 000000, 1, 0



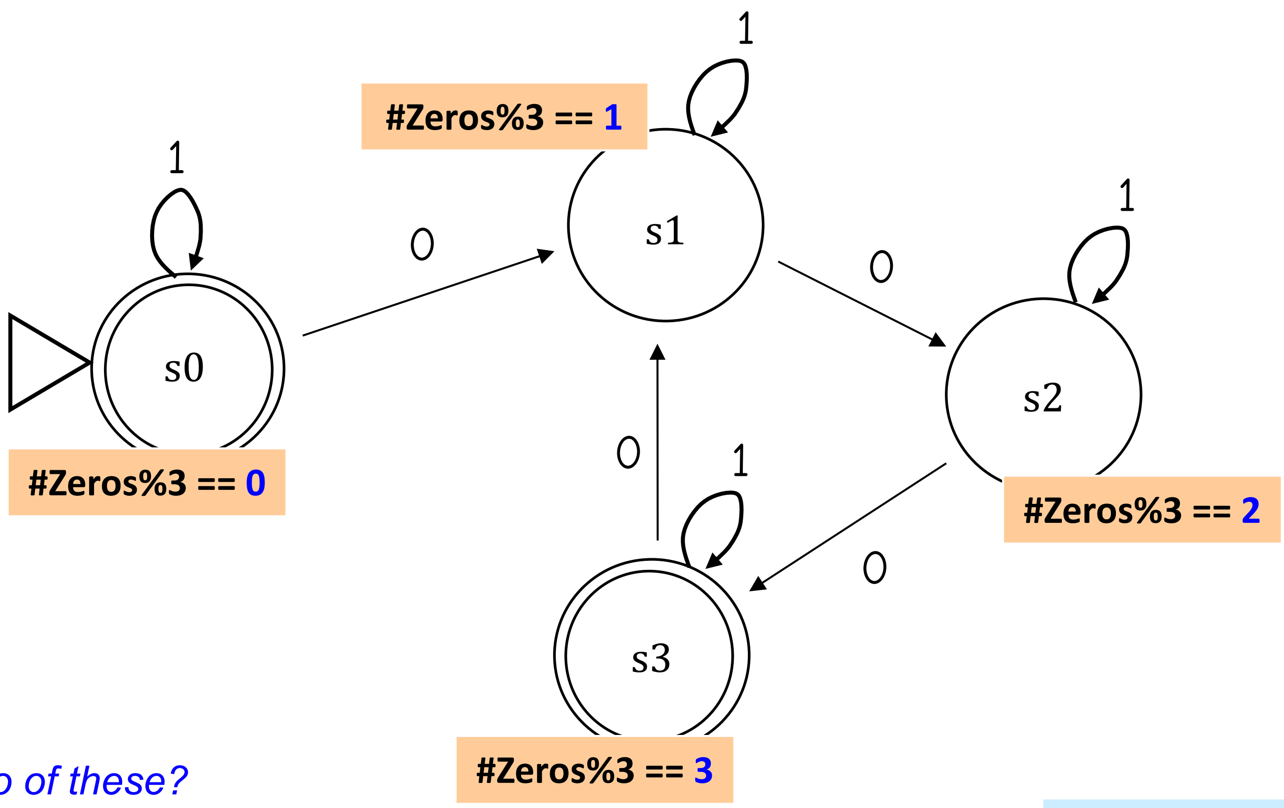
What do we need to complete this machine?

Number of **0**s is div. by 3

Draw a FSM accepting strings in which the number of zeros (**0**s) is a multiple of 3, so there are 0, 3, 6, ... zeros. **1**s don't matter.

Accepted: 110101110, 11, 0000010

Rejected: 101, 0000, 111011101111



Combine two of these?

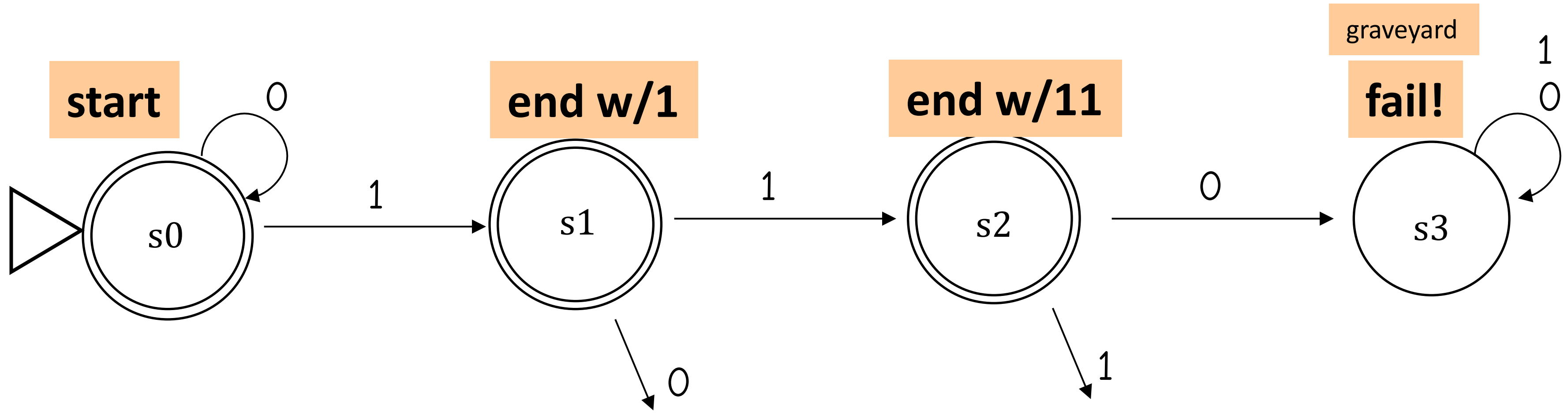
Minimum number of states?

No occurrences of **`110`**?

Draw a FSM accepting strings that do ***NOT*** anywhere contain the pattern **`110`**

Accepted: 1010001, 0001011

Rejected: 101001100, 011001



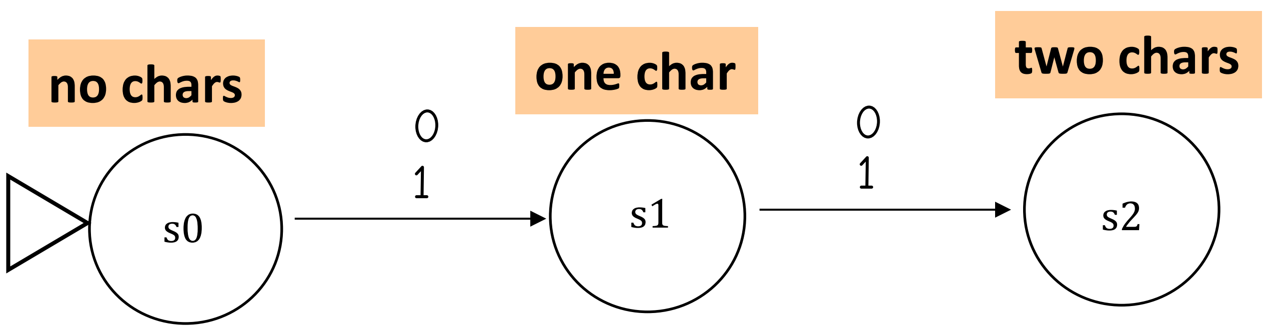
*Which **transitions** are still needed here?*

Third character is a **1**

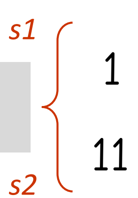
Draw a FSM accepting strings in which the third digit (from the left) is a **1**.

Accepted: 10<u>1</u>0001 and 01<u>1</u>0

Rejected: 11<u>0</u>00100 and 11



Why must s1 and s2 be separate states?



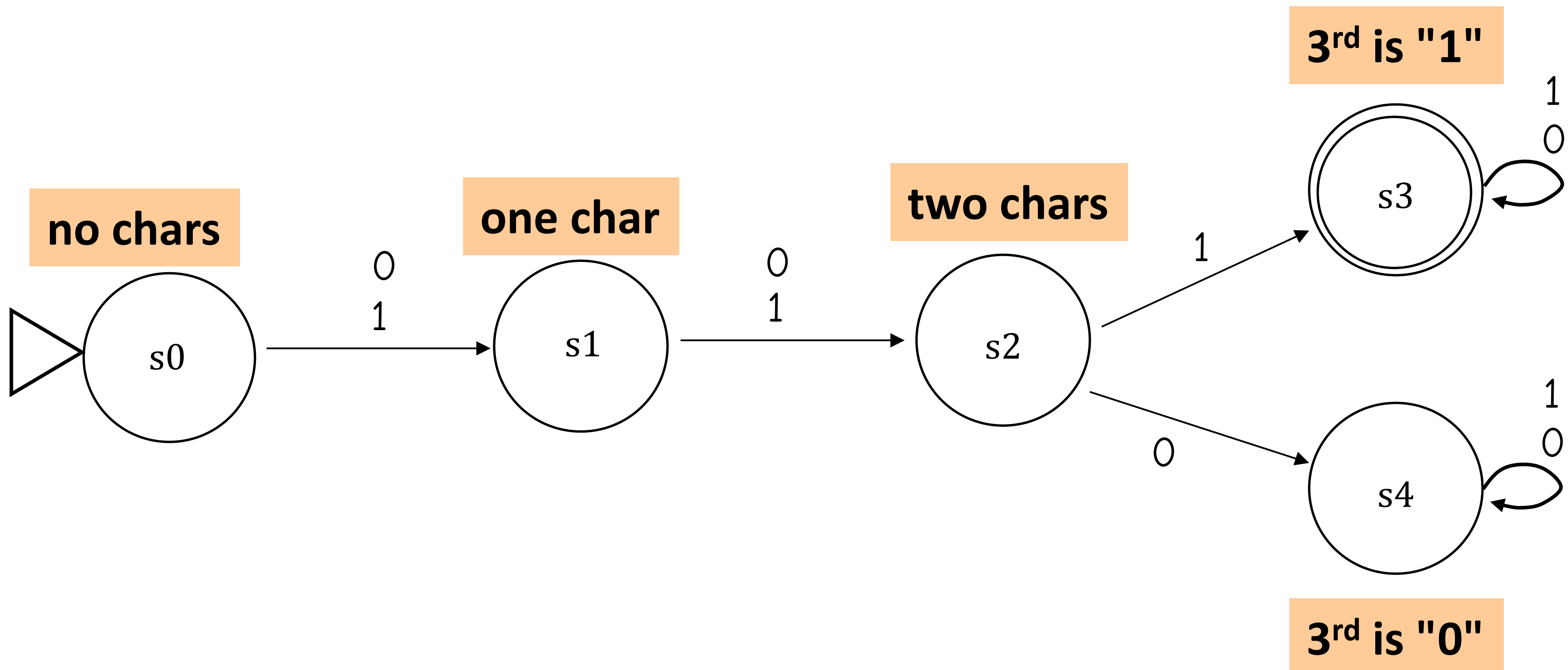
Minimum number of states?

Third character is a **1**

Draw a FSM accepting strings in which the third digit (from the left) is a **1**.

Accepted: 1010001 and 0110

Rejected: 11000100 and 11



Why must s1 and s2 be separate states?

s1: 1	*add 1*	11	*rejected*
s2: 11		111	*accepted*

5

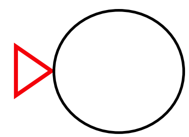
Minimum number of states?

Third-to-last character is a **1**?

Draw a FSM accepting strings whose third-to-last digit (from the right) is a **1**.

Accepted: 0**1**00 and 01**1**01

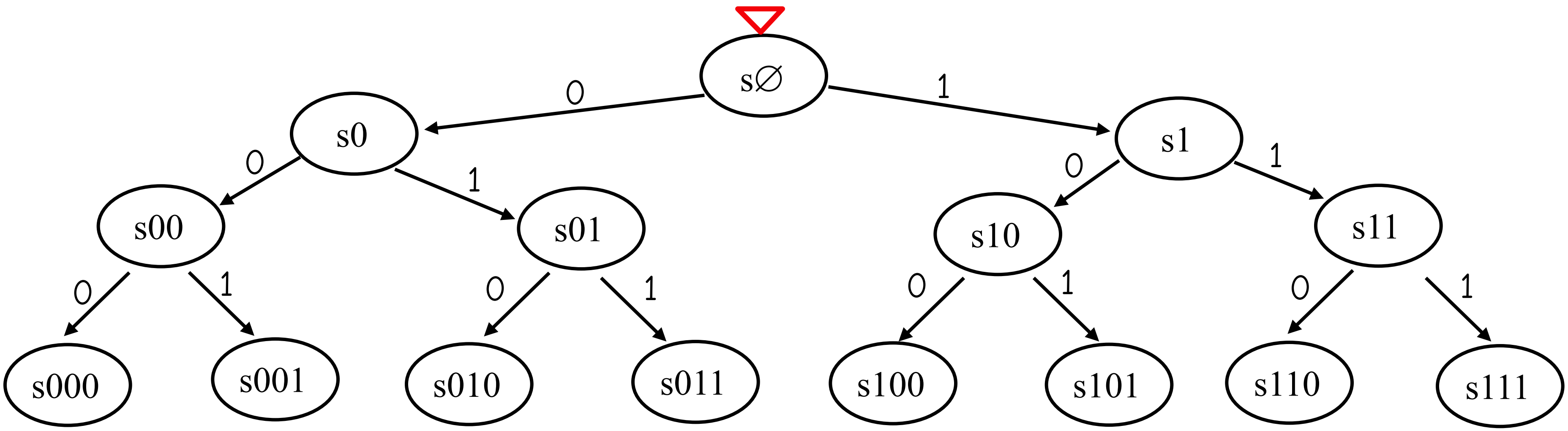
Rejected: 101**0**01 and 11



Gotta start somewhere!

Minimum number of states?

Third-to-last character is a `1`



I don't accept this solution!

Something's not right here: it's down-right h**arrow**ing!

Do we ***need*** 15 states?

Third-to-last character is a **1**

8 states?

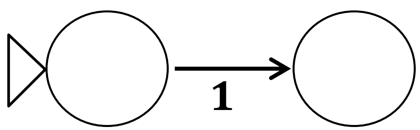
8 states <u>are required</u>!

Build-your-own FSMs

2/10 Draw a FSM accepting strings with at least two **1**s (anywhere). Others are rejected.

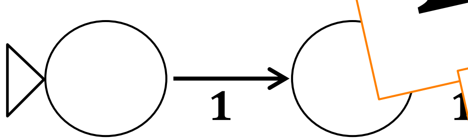
Accepted examples: 0101, 00010110, 111011, 11
Rejected examples: 0100, 1000, 000000, 1, 0

Hint - modify this starter FSM by adding labels, transitions, and one more state:



6/10 Draw a FSM that ac... contain t...

Accepted: 1010001...



Hint - there are FIVE more transitions – but no more states - needed here

4/10 Draw a FSM accepting strings in which the number of zeros (**0**s) is a multiple of 3, so there are 0, 3, 6, ... zeros. ***1**s don't matter!*

Accepted: 110101110, 11, 0000010
Rejected: 101, 0000, 111011101111

Hint: 1s never change the state!

Another hint: make a triangle!

...ber of states needed?

8/10 ...ich the ... **1**.

What's the minimum number of states needed?

10/10 ***Extra!*** Draw a FSM accepting strings whose third-to-last digit (3d from the **right**) is a **1**.

Acc: 0100 and 01101
Rej: 101001 and 11

Big picture

CS ~ building state machines

An autonomous vehicle's FSM

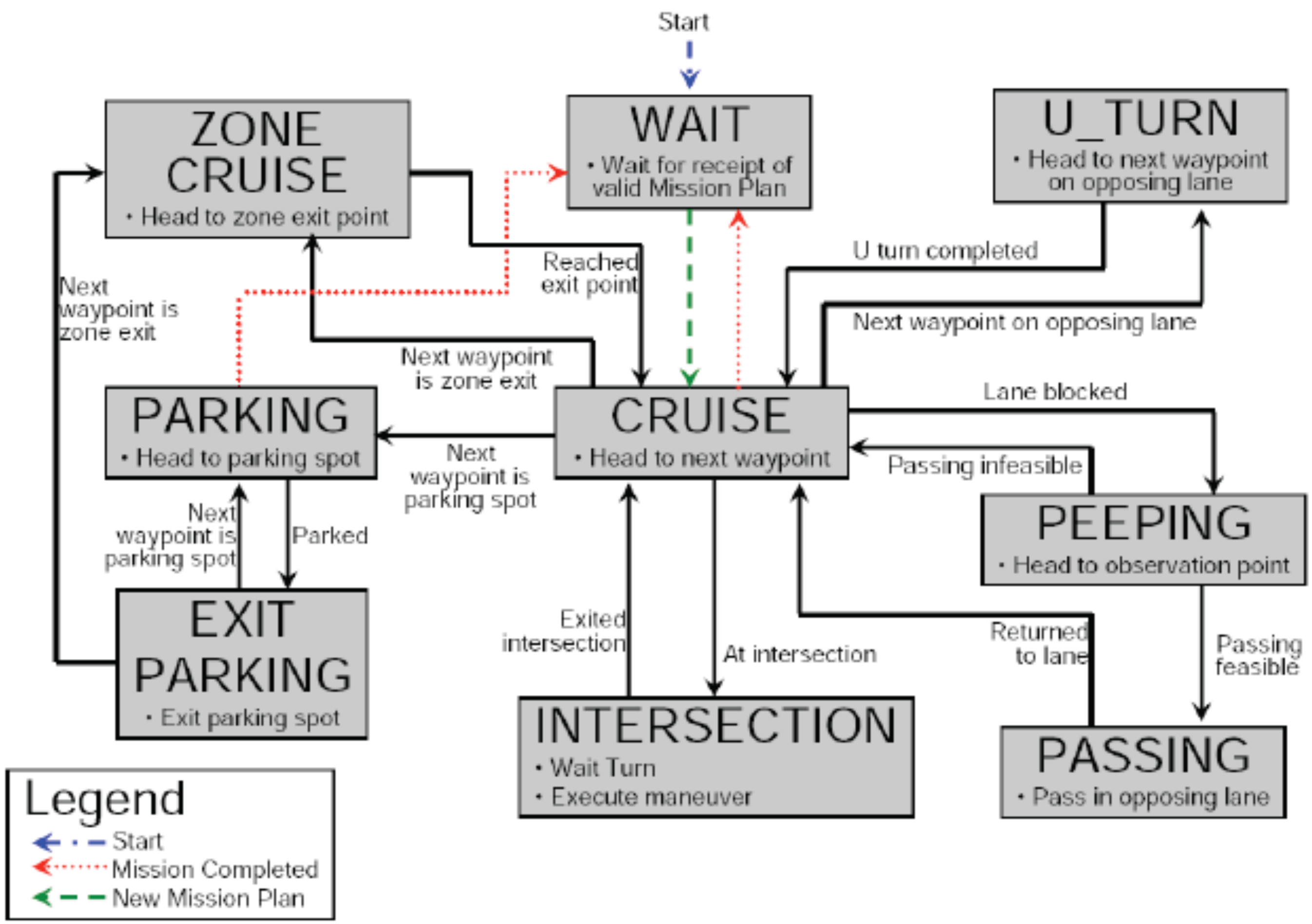
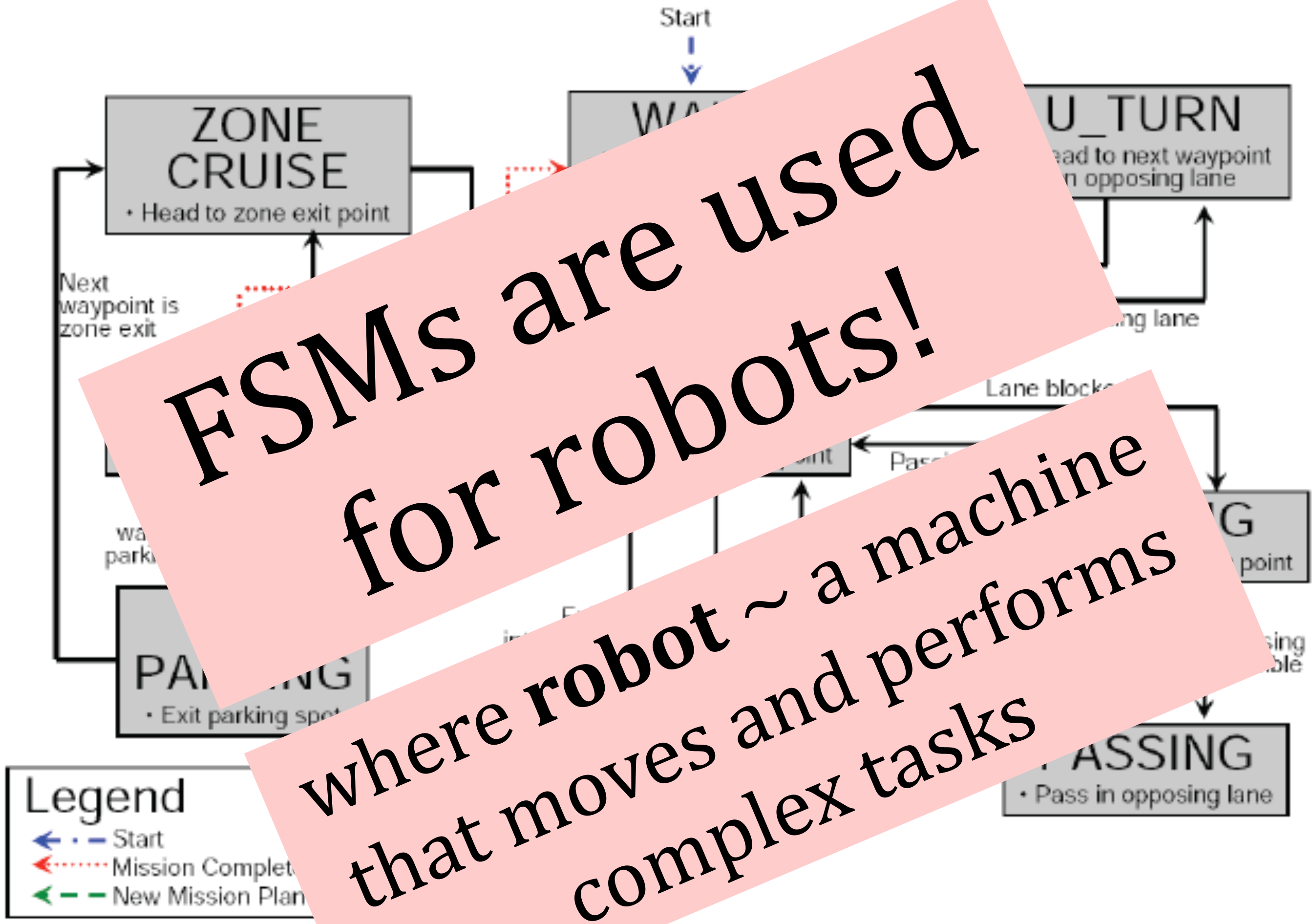


Fig. 9. Situational Interpreter State Transition Diagram. All modes are sub-modes of the system RUN mode (Fig 4(b)).

Why is an FSM a good design idea for an autonomous vehicle?

An autonomous vehicle's FSM



FSMs are used for robots!

where **robot** ~ a machine that moves and performs complex tasks

Fig. 9. Situational Interpreter State Tr gram. All modes are sub-modes of the system RUN mode (Fig 4(b)).

Why is an FSM a good design idea for an autonomous vehicle?

Robots use FSM control

towelFull.mp4

50x

What states can you "factor out" from watching this towel-folding?

Towel-folding states!

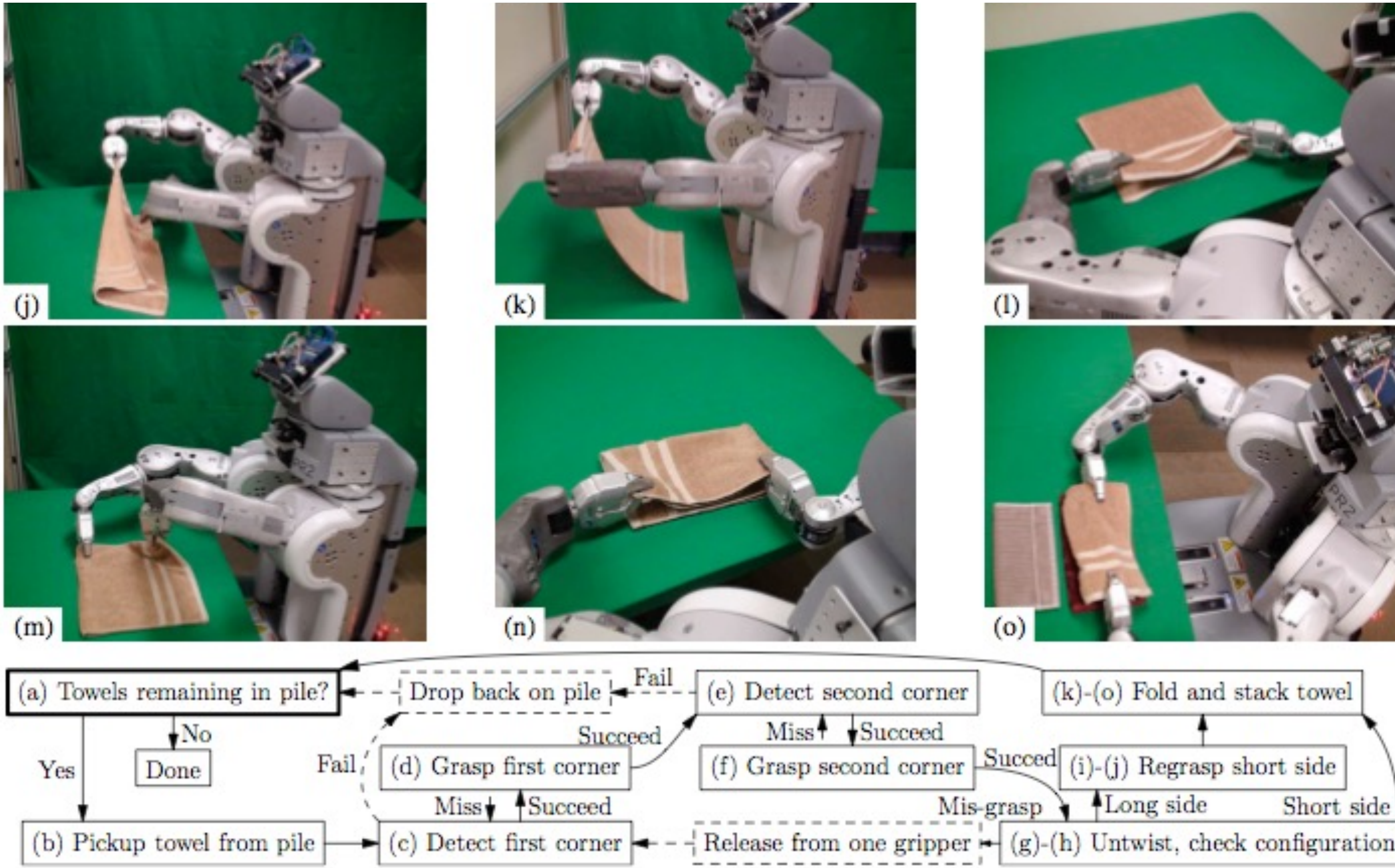


Fig. 2. The state machine model of the procedure: dashed lines indicate failure recovery cases. The images show an actual run.

ieee spectrum
INSIDE TECHNOLOGY

MAGAZINE MULTIM

AEROSPACE BIOMEDICAL COMPUTING CONSUMER ELECTRONICS ENERGY

AUTOMATON *The future of robots an*

BLOGS // AUTOMATON

U.S. Senator Calls Robot Projects Wasteful. Robots Call Senator Wasteful

POSTED BY: ERICO GUIZZO / TUE, JUNE 14, 2011

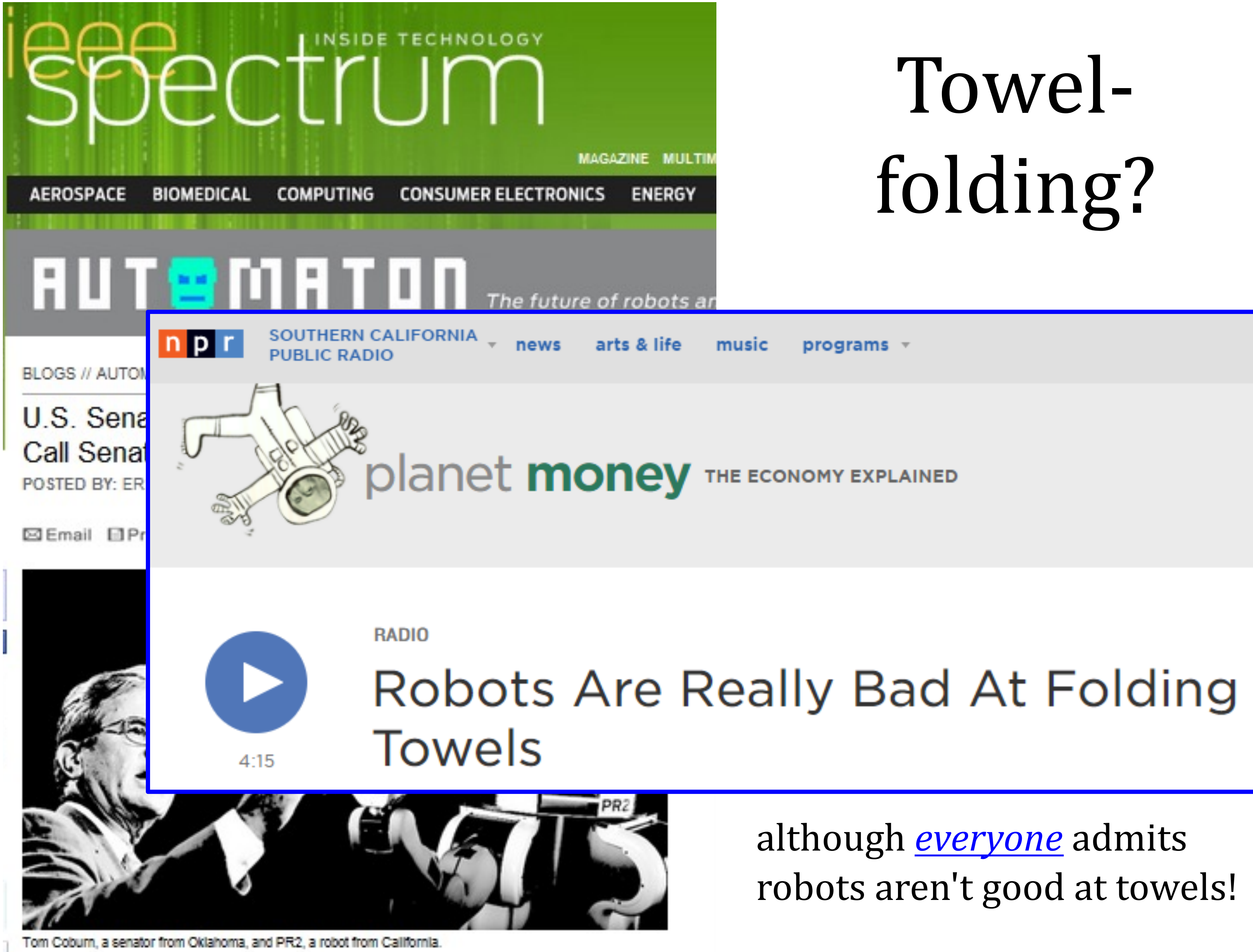
Email Print Share

Tom Coburn, a senator from Oklahoma, and PR2, a robot from California.

Towel-folding?

singled out as a questionable use of dollars...

Towel-folding?



although *everyone* admits robots aren't good at towels!

State-machine *limits*?

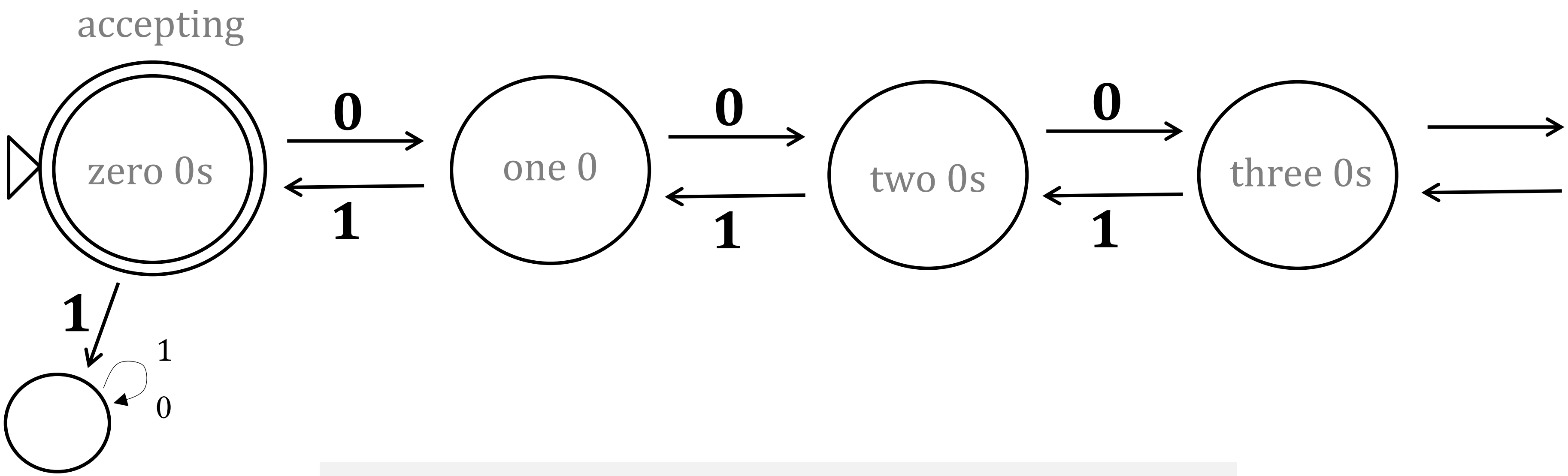
Are there *limits* to what FSMs can do?

they can't necessarily drive safely...

But are there any binary-string problems
that FSMs can't solve?

State-machine ***limits***?

You don't need three eyes to see some problems here!



How about a FSM that accepts strings with ***any*** **# of 0s** followed by the ***same*** **#of 1s**

rejected

011

001

11100

00110

accepted

000111

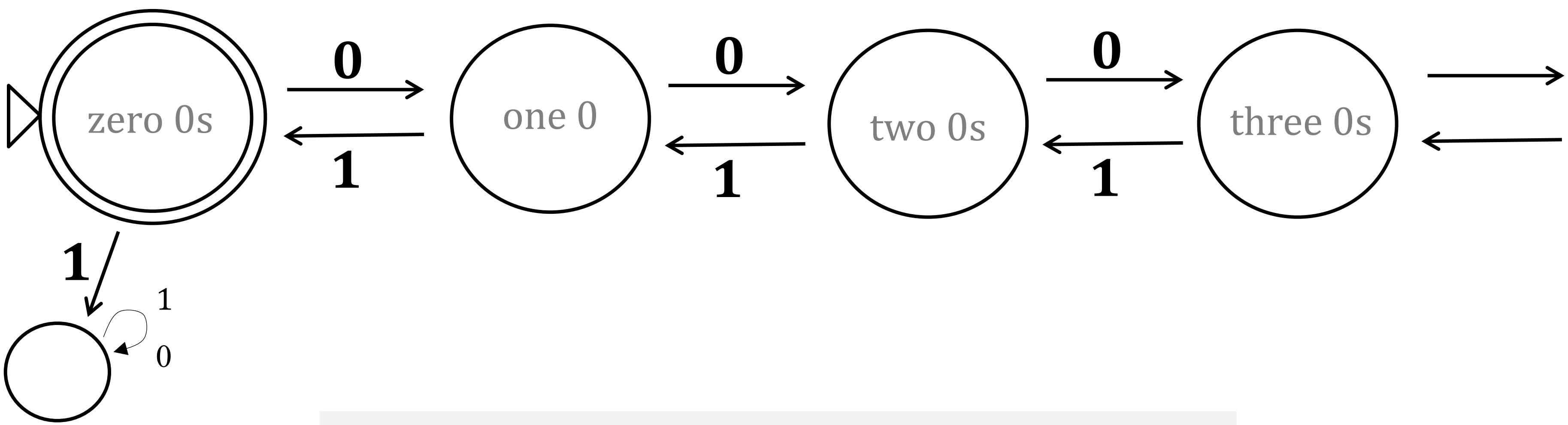
0011

01

λ

State-machine ***limits***?

You don't need three eyes to see some problems here!



How about a FSM that accepts strings with ***any*** **# of 0s** followed by the ***same*** **#of 1s**

rejected

011

001

11100

00110

FSMs "can't count"

at least, not arbitrarily high

accepted

000111

0011

01

λ

State-machines <u>are</u> limited.

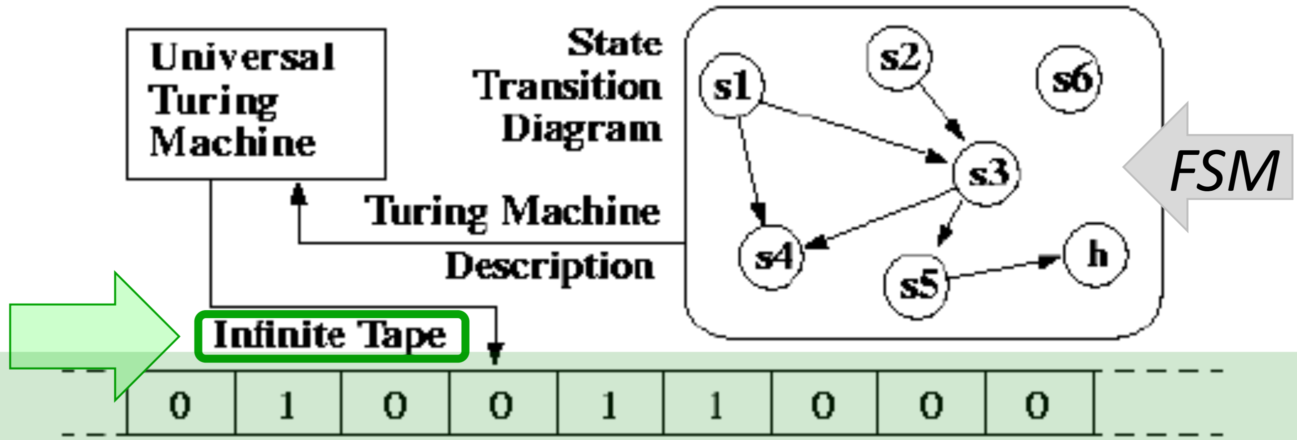
FSMs can't count

at least not arbitrarily high...

We need a **more powerful model** than FSMs...

What do we need to add?

Thursday: Turing Machines



*state machines w/ **memory!***

Lab sessions this week:

State machines + final projects...

Or is it state projects and final machines!?