COSC1107 Computing Theory

(We will commence soon. We are just allowing a few minutes for people to join and set up. *Please mute your microphone unless you are speaking.* You can raise your hand or use the chat at any time.)

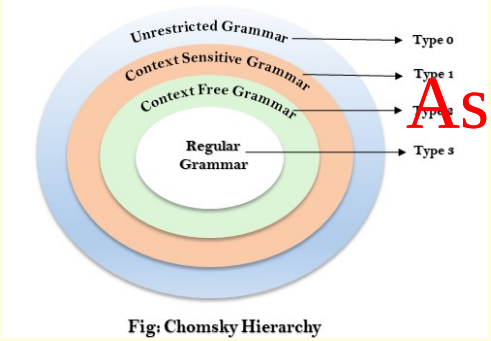
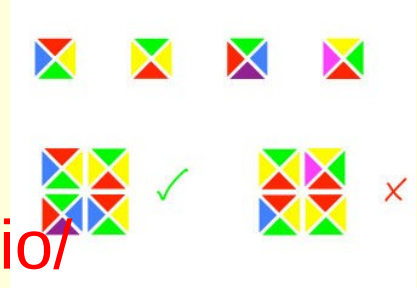
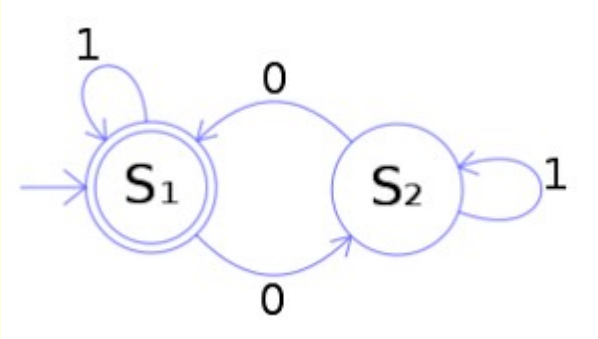
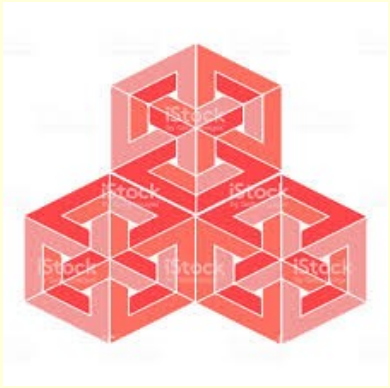
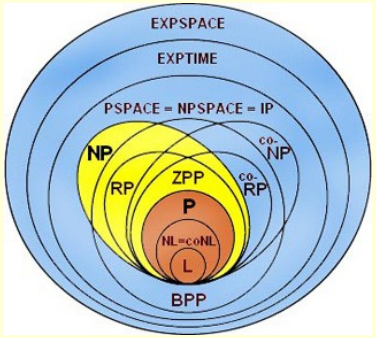


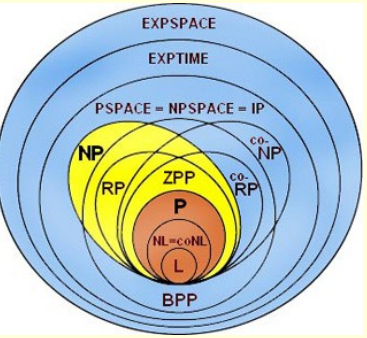
Fig: Chomsky Hierarchy

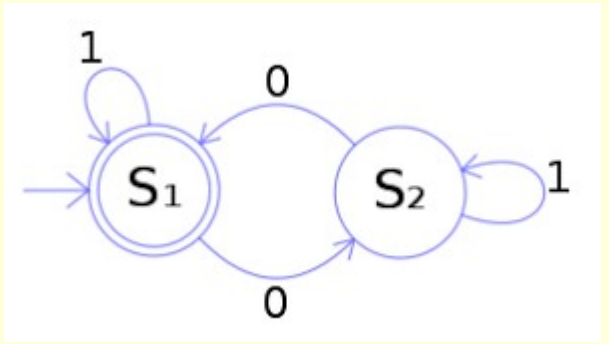












COSC1107

Week

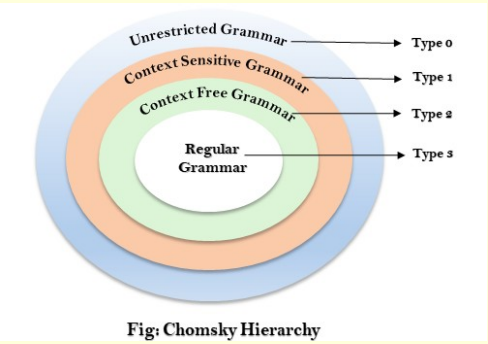


Fig: Chomsky Hierarchy

James Harland

james.harland@rmit.edu.au

* With thanks to Sebastian Sardina

Intro music 'Far Over' playing now …

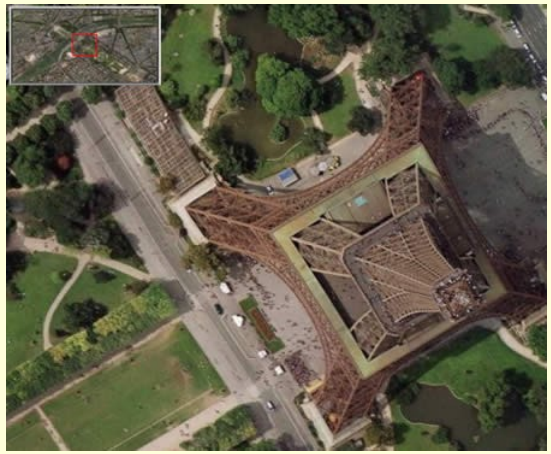
Acknowledgement

RMIT University acknowledges the people of the Woi wurrung and Boon wurrung language groups of the eastern Kulin Nations on whose unceded lands we conduct the business o University respectfully ac stors and Elders, past and present.

RMIT also acknowledges the Traditional Custodians and their Ancestors of the lands and waters across Australia where we conduct our business.

(add your name here to volunteer for this or email me)

Overview



- Questions?
- Universal Turing Machines
- Questions?
- Computability
- Questions?
- Platypus Game
- Questions?

What **can** be done

be done

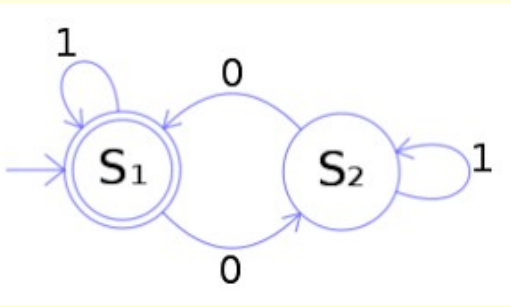
Both!

Questions?

Questions?

Questions?

Church-Turing thesis

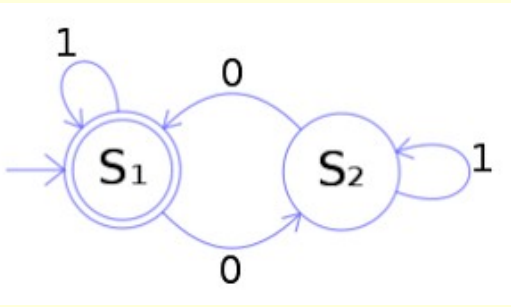


Church-Turing thesis (1936) *Any* computation can be translated to an equivalent computation on a Turing machine

so Turing machines are adequate for anything that can be computed

- No need to seek more powerful (there is nothing!)
- Enables rigorous analysis of c
- Can't be done by a TM means ne at all!
- Thesis, not a theorem
- Observed property of the universe, like scientific laws
- Consistent with observation, no counterexamples known

Church-Turing thesis



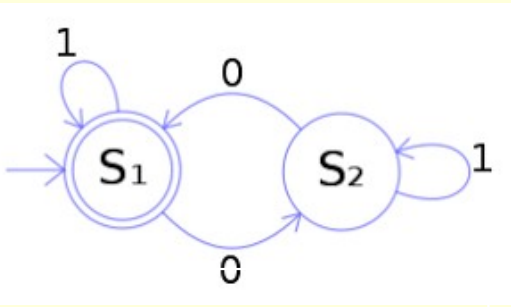
"A man* provided with paper, pencil, and rubber, and subject to strict discipline, is in effect a universal machine."
-- Alan Turing

"The idea behind digi ined by saying that these machines are in erations which could be done by a human c
-- Alan Turing

"It is possible to invent a single machine which can be used to compute any computable sequence. If this machine U is supplied with a tape on the beginning of which is written the [encoding] of some computing machine M ,then U will compute the same sequence as M."
-- Alan Turing

* This now sounds very dated. But that is what he said back in the 1930's …

Church-Turing thesis

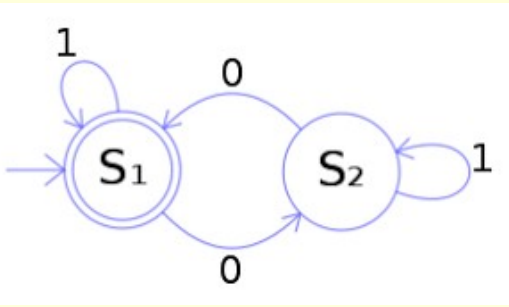


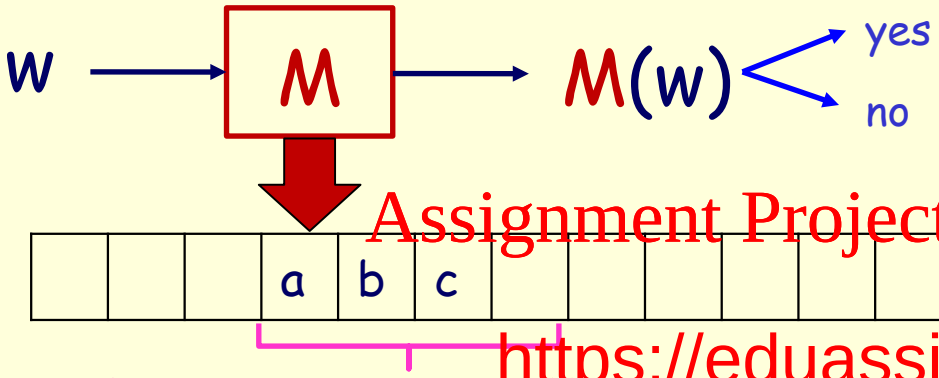
Model of computation	Description
enhanced Turing machines	multiple heads/tapes, 2D tape, nondeterminism
untyped lambda calculus	method to define and manipulate functions
recursive functions	functions dealing with computation on integers
unrestricted grammars	ement rules used by linguists
programming languages	sp, Prolog, …
random access machines	registers plus main Y, Pentium
cellular automata	automata cells whic ased on local interactions
quantum computer (??)	computer compute using superposition of quantum states
DNA computer	compute using biological operations on DNA
…	…

GOTCHA!

All equivalent. No counterexamples!

Universal Turing machine

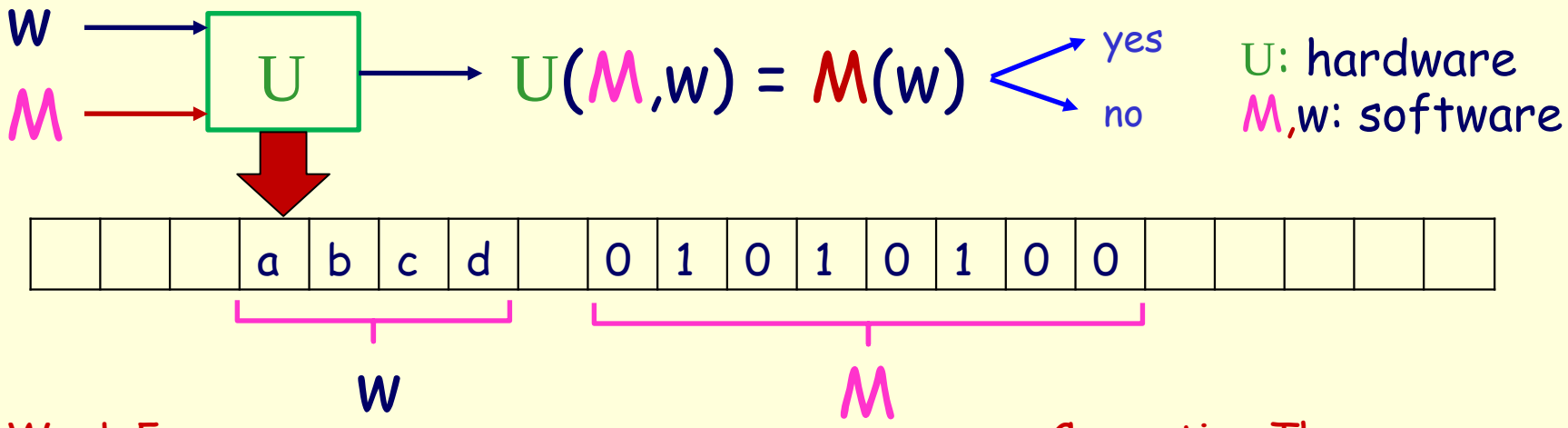




Solves one specific problem
New M for each new problem

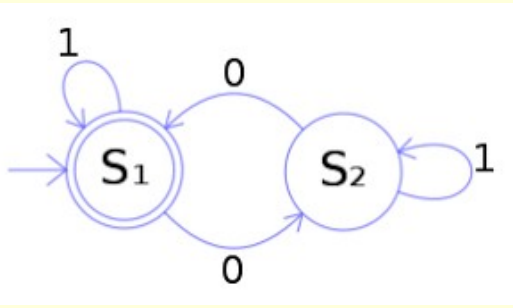
"U simulates the action of M on w"

ure that can solve
utable) problem!



U: hardware
M,w: software

Universal Turing machine



I'd like a Turing machine please!

Certainly Master Baggins! Which one? We have many!

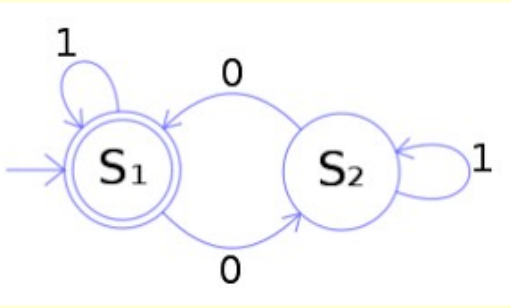
There are so many! It is confusing …

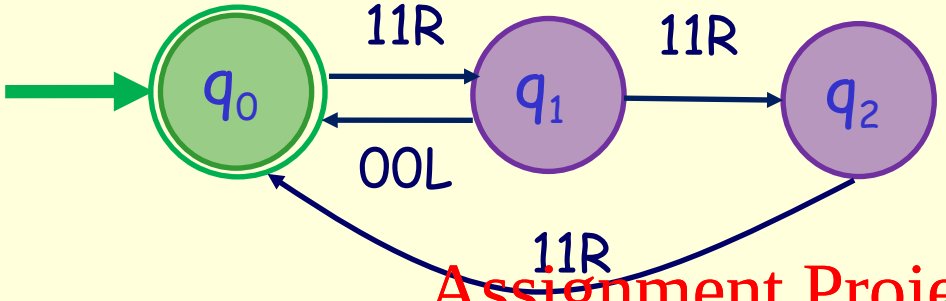
Frodo! Just use this one!
You will never need another!

U

"lean green Turing machine!"

Universal Turing machine





separate items by 0

T = (q,x,y,D,r) e)0en(r)

q_0	1	1	R	q_1	
q_1	1	1	R	q_2	110110110
q_1	0	0	L	q_0	1101010101
q_2	1	1	R	q_0	1110110110101

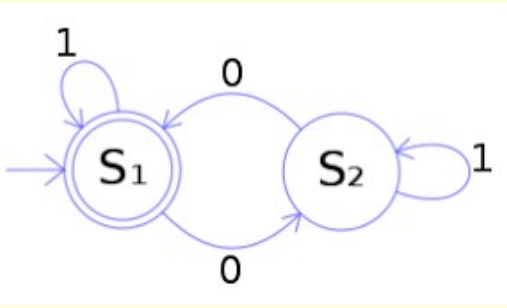
i	en(i)
q_0	1
q_1	11
q_2	111
0	1
1	11
	111
L	1
R	11

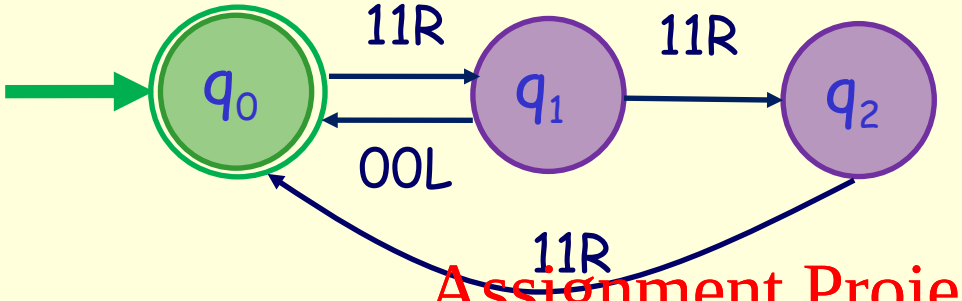
separate transitions by 00

code(M) = 000 en(T_1) 00 en(T_2) 00 en(T_3) 00 en(T_4) 000

start and end encoding with 000

Universal Turing machine





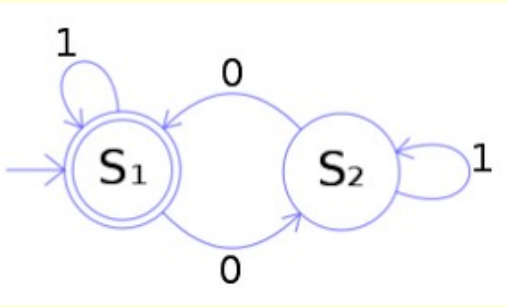
q_0	1	1	R	q_1	
q_1	1	1	R	q_2	
q_1	0	0	L	q_0	11010101
q_2	1	1	R	q_0	1110110110110 1

i	en(i)
q_0	1
q_1	11
q_2	111
0	1
1	11
	111
L	1
R	11

code(M) = 000 en(T_1) 00 en(T_2) 00 en(T_3) 00 en(T_4) 000

000101101101101100110110110110111001101010101001110110110110 1000

Universal Turing machine

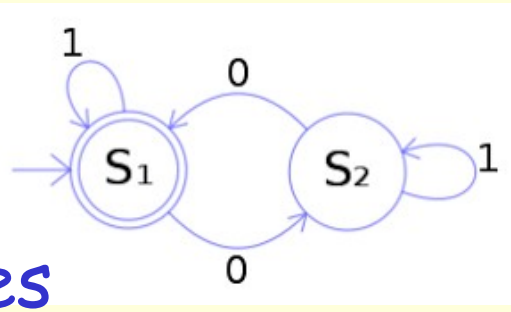


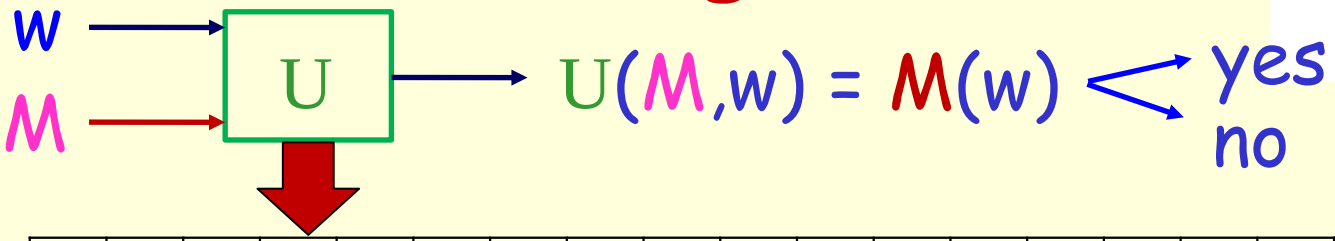
Turing machine string input to another TM

"Analyser" TMs can

- determine if (another) TM
  - has a 'halt' transition (ie no transition for a given pair of state & symbol)
  - is deterministic
  - contains a 'pla
  - simulates an F g the tape)
  - …
- Count the transitions, states, sym M
- Add a transition to a TM
- Change a transition in a TM
- "Join" two TMs (final state of one = initial state of s
- Simulate the action of the input TM
- 'Swap' the acceptance behaviour of the input TM
- Change the output of the input TM
- Be given their own definition as input (!!)

Universal Turing machine





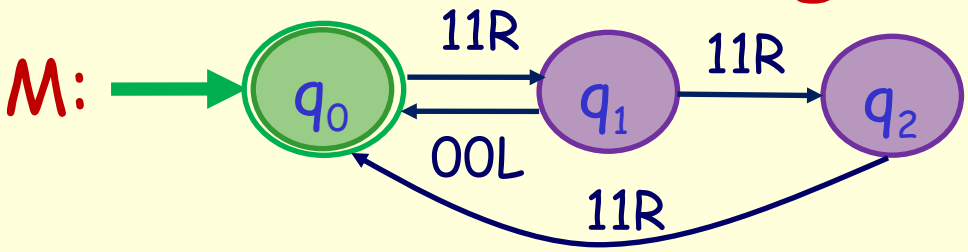
ne machine to rule them all;
ne machine to find them;
ne machine to …"

Build UTM using 3 tap

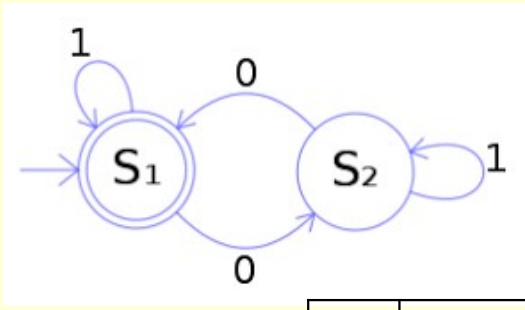
- One for input (encoded machine and inp
- One for the state of M
- One for the tape of M

Tape 1	code(M)code(w)
Tape 2	current state of M (encoded)
Tape 3	tape of M (encoded)

Universal Turing machine



w: 11001



i	en(i)
q_0	1
q_1	11
q_2	111
0	1
1	11
	111
L	1
R	11

M

U's alphabet may be different from M's ...

U:

code(M) code(w)

000101101101101100110110110110111001101010101001110110110110100011011010101100

U

code(M)code(w)

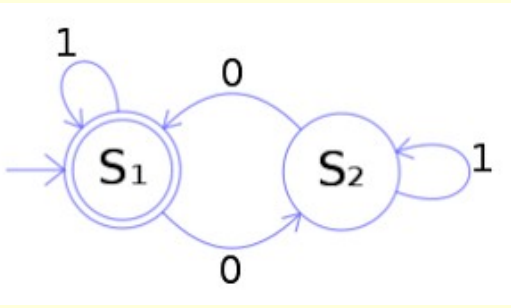
1

M state (encoded)

11011010101100

M tape (encoded)

Universal Turing machine

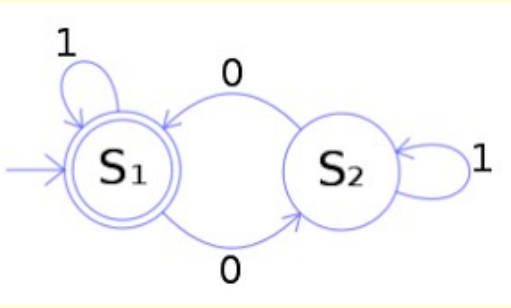


U:

1. Check tape 1. If format wrong, loop forever
2. Write code(w) on tape 3
3. Write en(q_0) on tape 1
4. Find transition matching q on tape 2 & x on tape 3. If none, halt.
5. Given en(q)0en(x)
   - Replace en(q)
   - Write en(y) ap
   - Move tape 3 according to D (la
6. Go to step 4

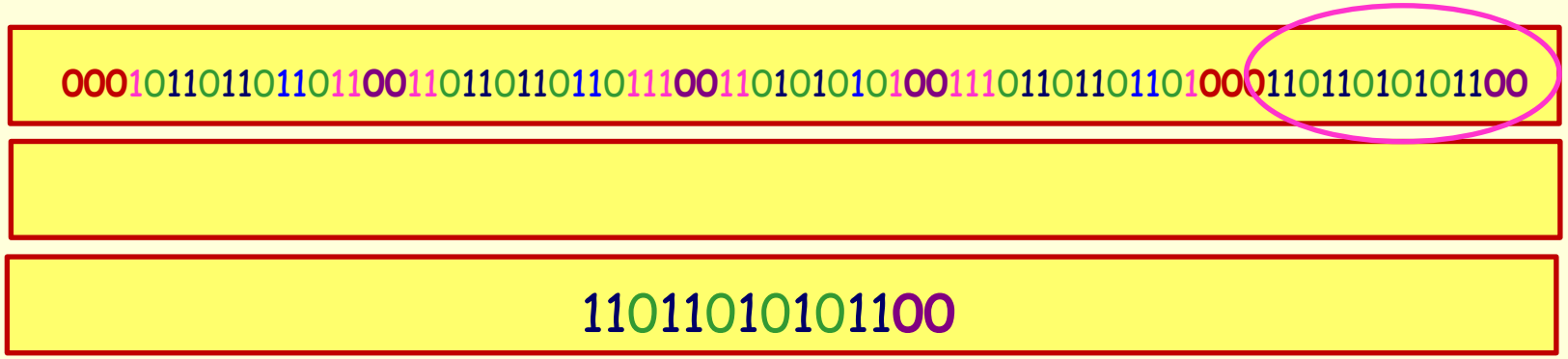
000101101101101100110110110110111001101010101001110110110110100011011010101100

Universal Turing machine

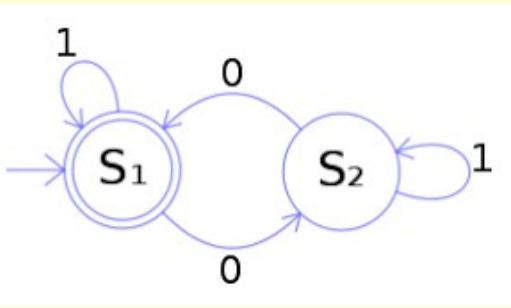


U:

1. Check tape 1. If format wrong, loop forever
2. **Write code(w) on tape 3**
3. Write en(q_0) on tape 1
4. Find transition matching q on tape 2 & x on tape 3. If none, halt.
5. Given en(q)0en(x)
   - Replace en(q)
   - Write en(y) ap
   - Move tape 3 according to D (left
6. Go to step 4



Universal Turing machine



U:

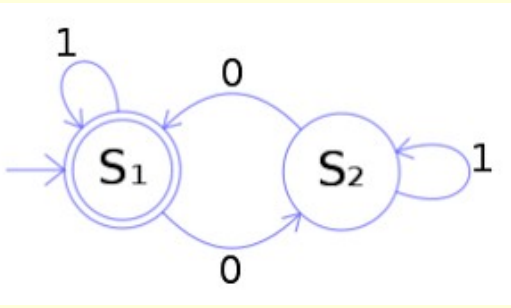
1. Check tape 1. If format wrong, loop forever
2. Write code(w) on tape 3
3. **Write en(q_0) on tape 1**
4. Find transition matching q on tape 2 & x on tape 3. If none, halt.
5. Given en(q)0en(x)
   - Replace en(q)
   - Write en(y) ap
   - Move tape 3 according to D (la
6. Go to step 4

000101101101101100110110110110111001101010101001110110110110100011011010101100

1

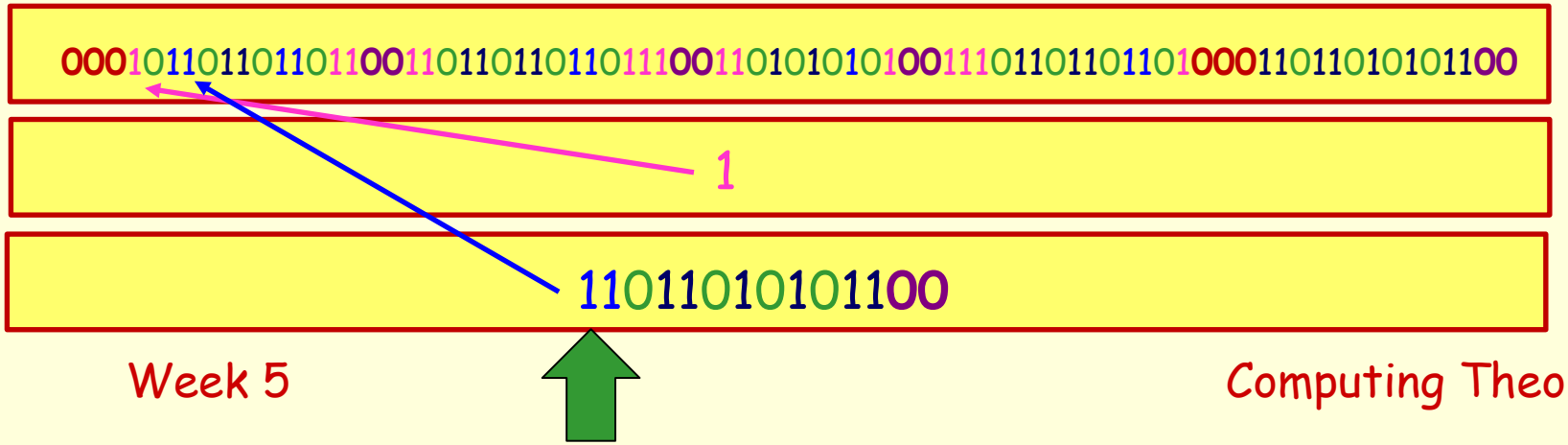
11011010101100

Universal Turing machine

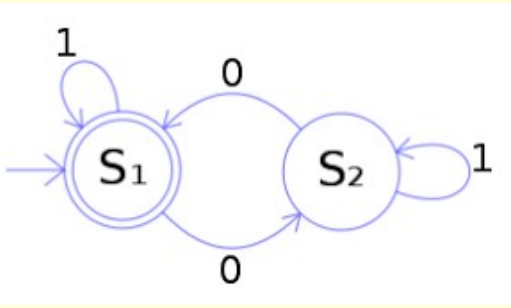


U:

1. Check tape 1. If format wrong, loop forever
2. Write code(w) on tape 3
3. Write en(q_0) on tape 1
4. Find transition matching q on tape 2 & x on tape 3. If none, halt.
5. Given en(q)0en(x)0
   - Replace en(q) w
   - Write en(y) app
   - Move tape 3 according to D
6. Go to step 4

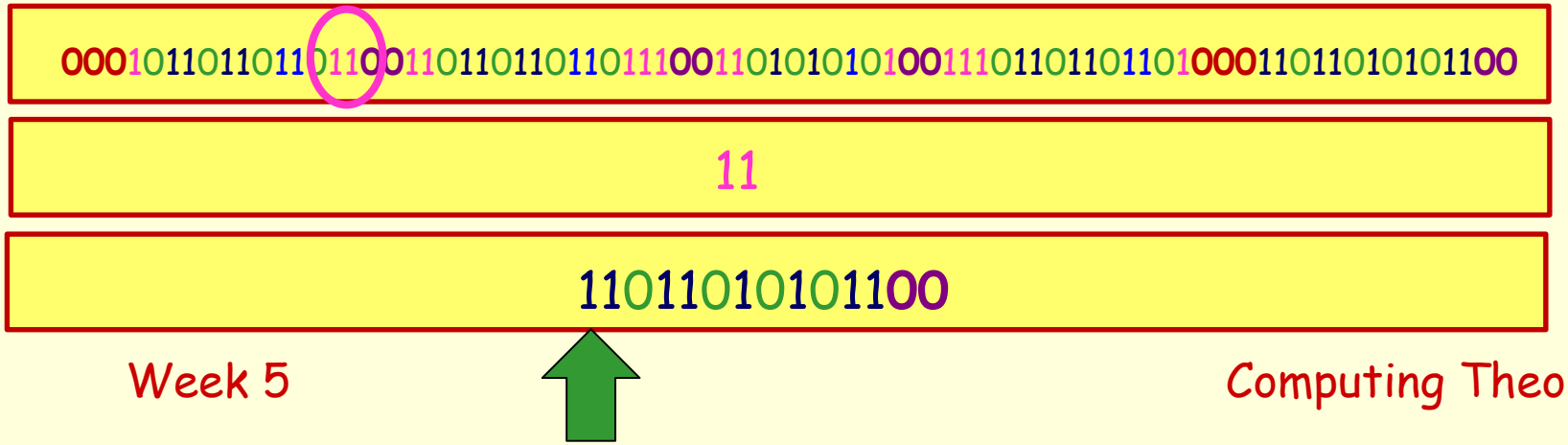


Universal Turing machine

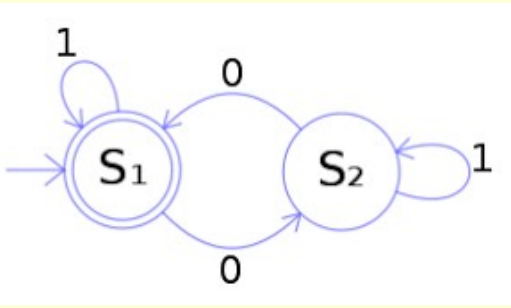


U:

1. Check tape 1. If format wrong, loop forever
2. Write code(w) on tape 3
3. Write en(q_0) on tape 1
4. Find transition matching q on tape 2 & x on tape 3. If none, halt.
5. Given en(q)0en(x)
   - **Replace** en(q)
   - Write en(y) ap
   - Move tape 3 according to D (la
6. Go to step 4

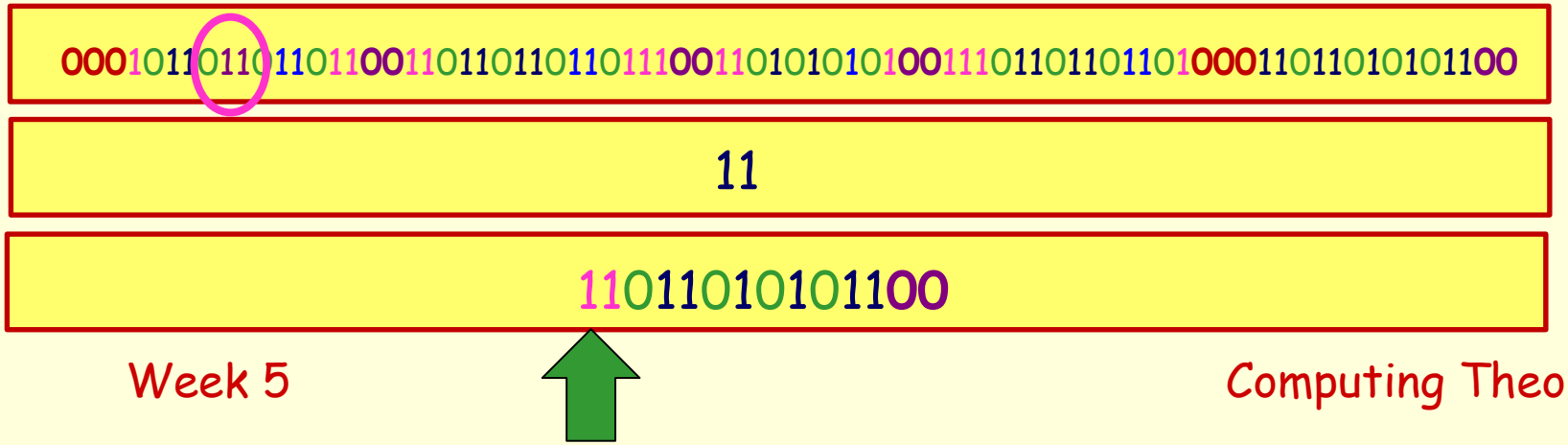


Universal Turing machine

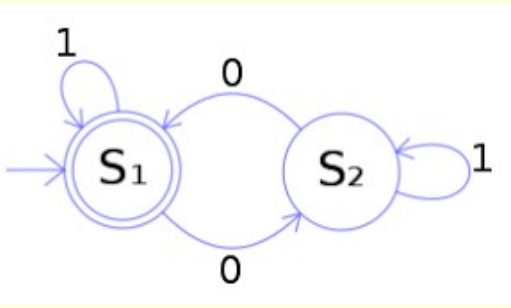


U:

1. Check tape 1. If format wrong, loop forever
2. Write code(w) on tape 3
3. Write en(q_0) on tape 1
4. Find transition matching q on tape 2 & x on tape 3. If none, halt.
5. Given en(q)0en(x)
   - Replace en(q)
   - **Write** en(y) **ap**
   - Move tape 3 according to D (la
6. Go to step 4

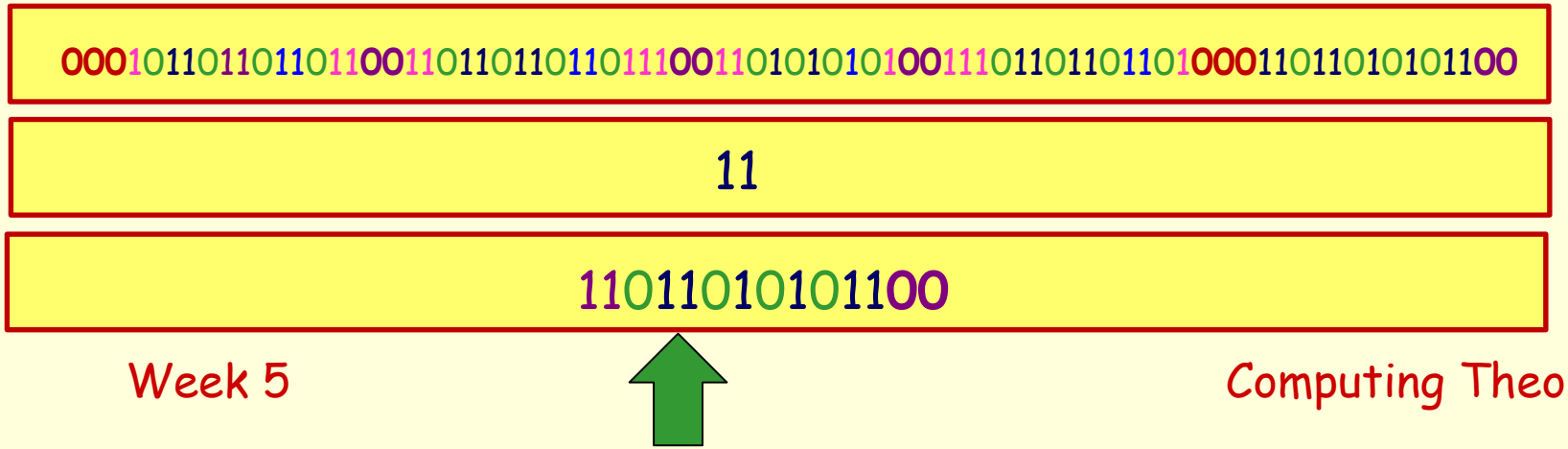


Universal Turing machine

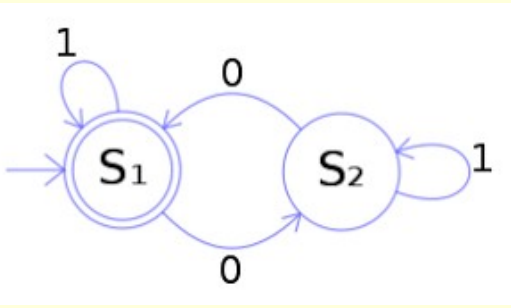


U:

1. Check tape 1. If format wrong, loop forever
2. Write code(w) on tape 3
3. Write en(q_0) on tape 1
4. Find transition matching q on tape 2 & x on tape 3. If none, halt.
5. Given en(q)0en(x)
   - Replace en(q)
   - Write en(y) ap
   - Move tape 3 according to R
6. Go to step 4



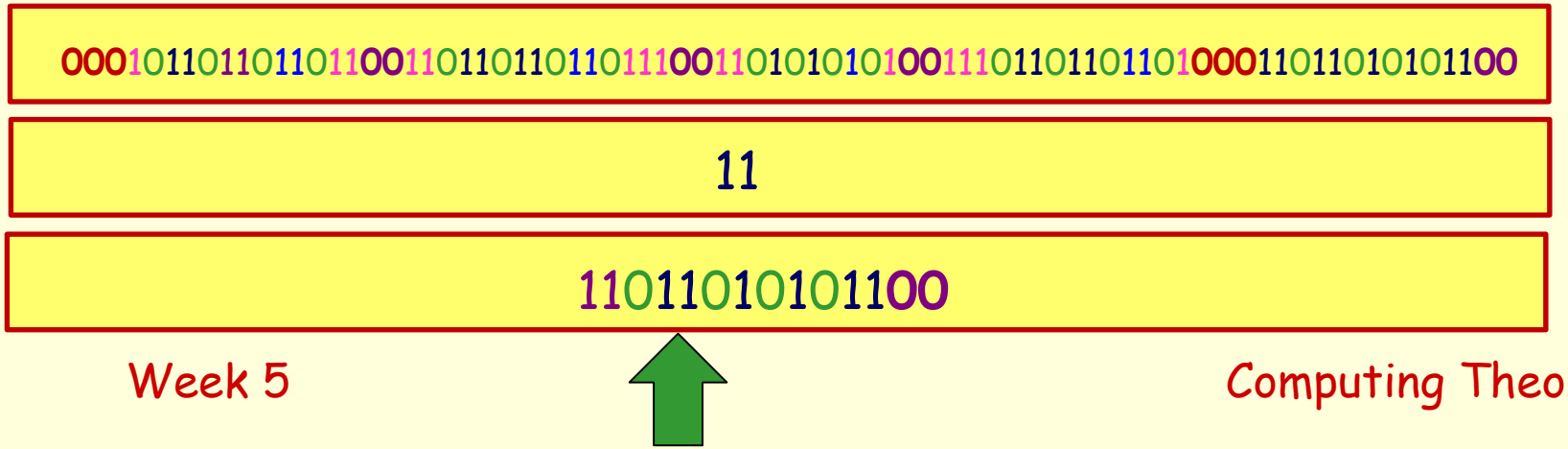
Universal Turing machine



U:

1. Check tape 1. If format wrong, loop forever
2. Write code(w) on tape 3
3. Write en(q_0) on tape 1
4. Find transition matching q on tape 2 & x on tape 3. If none, halt.
5. Given en(q)0en(x)
   - Replace en(q)
   - Write en(y) ap
   - Move tape 3 according to D (left)
6. Go to step 4

AND SO ON ...



Questions?

Questions?

Questions?

Impossible? Never?

Mary said Tom would never call her again, but I told her, "**Never say never**."
-- 'Pickwick Papers' by Charles Dickens

"Never is too long
--Treebeard in 'The Tolkien

~~"Believe the unbelievable.~~
~~Dream the impossible.~~
~~Never take 'no' for an answer!"~~

If there is convincing evidence that something is impossible, …

Impossible? Never?

When can you ever say 'never!'?

Solutions of polynomial equations

- 'closed' formulae exist for quadratic, cubic & quartic
- No such general f e ≥ 5 (!!)

Geometric const other shapes

- Only 'straightedge and compass' al
- Impossible constructions
  - 'Square the circle'
  - 'Double the cube'
  - Angle trisection

Impossible? Never?

When can you ever say 'never!'?

Clever people have got this wrong in the past!

Many predictions will never happen' have bee

So can you ever say 'never'?

Impossible? Never?

???

Known to be impossible

Riemann Hypothesis
P = NP
Computers pass Turing test
…

Status unknown

Computers outperform humans at chess
Self-driving cars
Text to speech and speech to text
Computer facial recognition
…

Known to be possible

Quiz time!

Go to Canvas and find the quiz Lectorial 5 Question set

- Not worth any marks
- You can consult other students if you wish
- Time limit will be 10 minutes

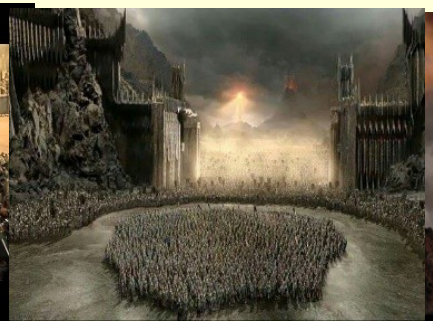
Are you ready?

Are you sure?

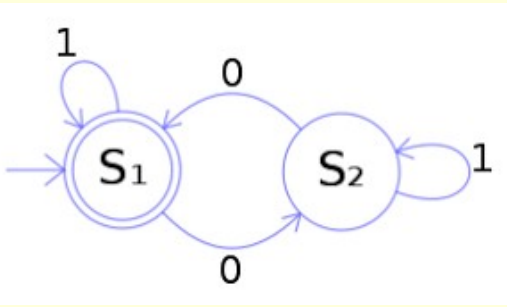
Go!

The pictures will take 10 minutes to disappear!

Thomas music means 1 minute left!



Decision problems



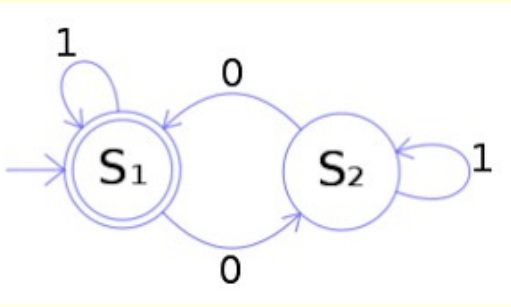
A decision problem is a **question** in a **formal system** with a **yes-no answer**, depending on some input parameters.

Sort: Is a list of integers sorted?
Hamiltonian circuit: I his graph that visits every node?
Primality: Is a given n
Python syntax: Is a given program syn rrect?
Password: Is the given password corr
Bank balance: Is the transaction approved?

Halting: Given a TM M and input w, does M halt on w?

Harder than it may seem …

Decision problems

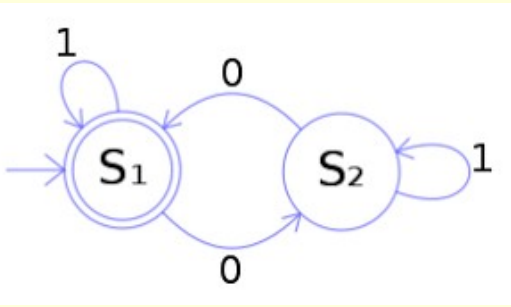



```
% n is an integer ≥ 1
while n > 1 do
        if n mod 2 == 0
                n = n/2
        else
                n = 3n+1
elihw
```


5 16 8 4 2 1
6 3 10 5 16 8 4 2 1
7 22 11 34 17 52 26 13 40 20 10 5 … 1
9 28 14 7 … 1
15 46 23 70 35 106 53 160 80 40 20 … 1

Does it always terminate? **No-one knows (!!)**

Decision problems

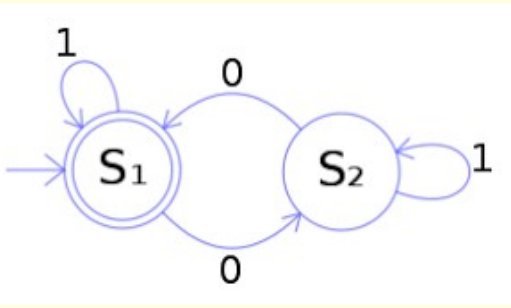



```
% n is an integer ≥ 1
while n > 1 do
        if n mod 2 == 0
                n = n/2
        else
                n = 3n+1
elihw
```


5 16 8 4 2 1
6 3 10 5 16 8 4 2 1
7 22 11 34 17 52 26 13 40 20 10 5 … 1
9 28 14 7 … 1
15 46 23 70 35 106 53 160 80 40 20 … 1

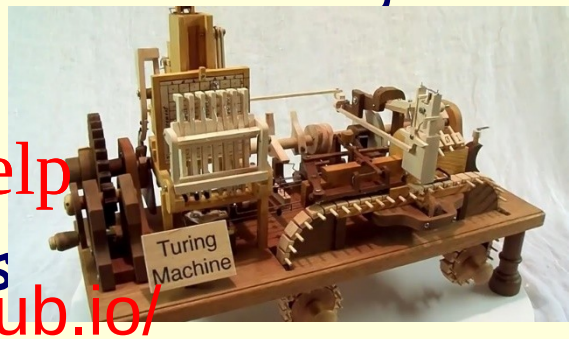
Does it always terminate? **No-one knows (!!)**

Decidable problems

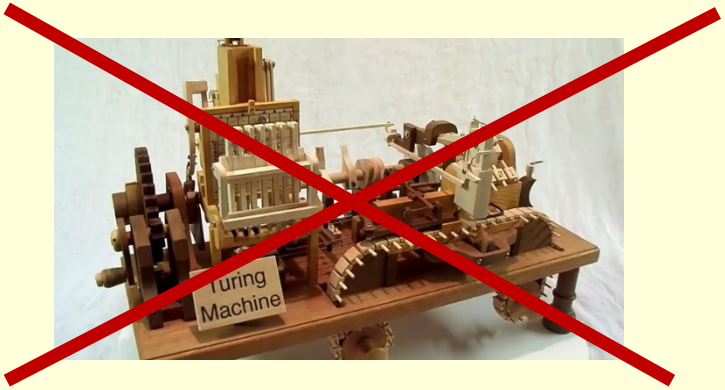


A decision problem is **decidable** if there **exists** a Turing machine M that solves it, ie

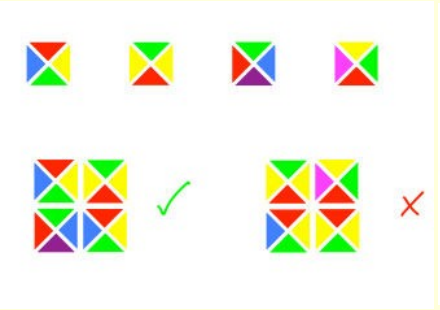
- M halts on all inputs w
- M outputs yes i yes
- M outputs no if o
- M produces no other outputs (so M recogniser)

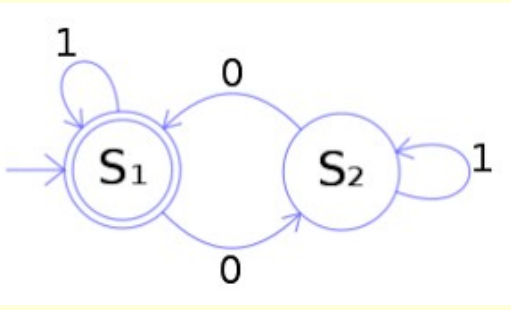


A decision problem is **undecidable** if there **does not exist** a Turing machine M that solves it.



Tile problem





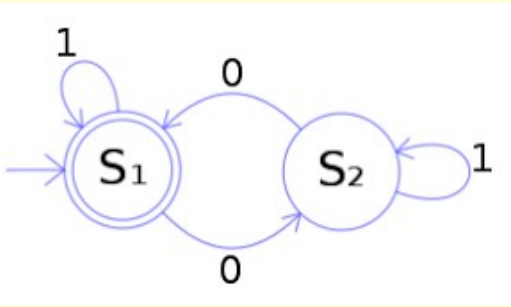
Can you cover any rectangular area with a set of tiles?

annot be rotated
olours must match

- Tiles can be copied

There is no Turing machine that solves this decision problem (!!!)

Halting problem



Halting problem: Given a Turing machine M and an input w, does M halt on w?

For every TM M and every input w, either

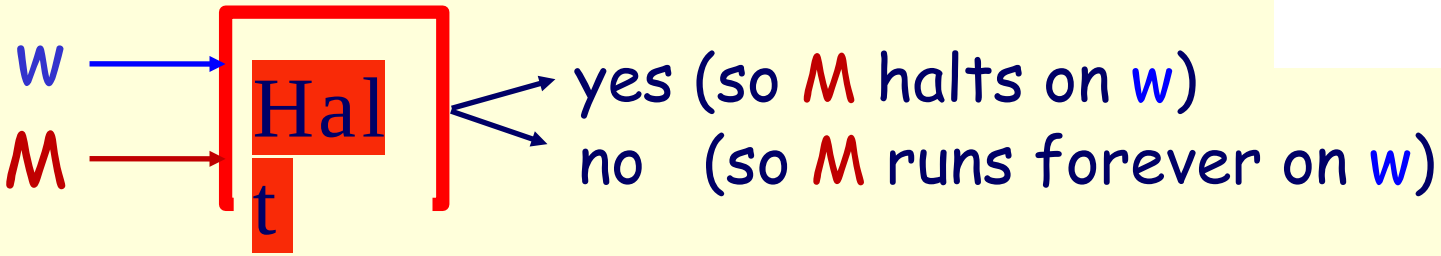
- M halts on w
- M doesn't halt

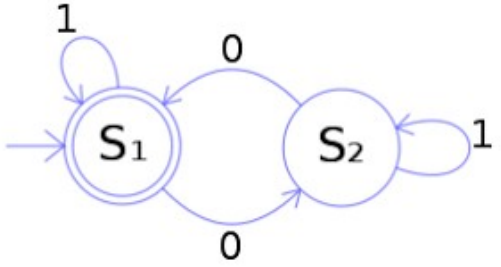
The Halting problem is a decisio _ about Turing machines ..

Question: Is there a Turing machine that can solve the Halting problem for Turing machines?

Is the Halting problem decidable?

Halting problem





Is there a TM Halt ?

- Halt terminate ry M and every w)
- Halt accepts code(M)code(w) ines on w
- Halt rejects code(M)code(w) if M does not terminate on w

Option 1:

Halt exists

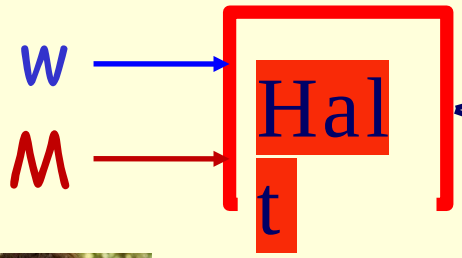
The Halting problem is decidable

Option 2:

Halt does not exist

The Halting problem is undecidable

Halting problem



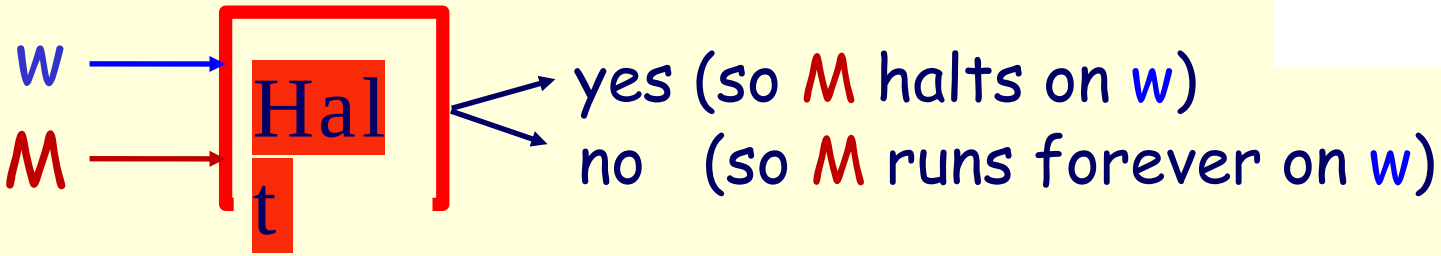
"Gandalf, does the machine Halt exist?"

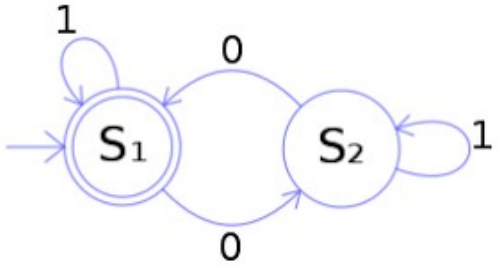
"No, it's a

Clues:

- "This statement is false" (Epiminedes)
- Paradox of the barber
- Self-reference

Halting problem





The Halting problem is undecidable

Proof: Assume th ie

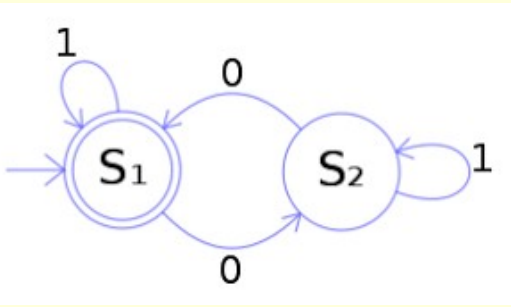
- Halt terminates on every input (ie. every w)
- Halt accepts code(M)code(w) if M t w
- Halt rejects code(M)code(w) if M does not terminate on w

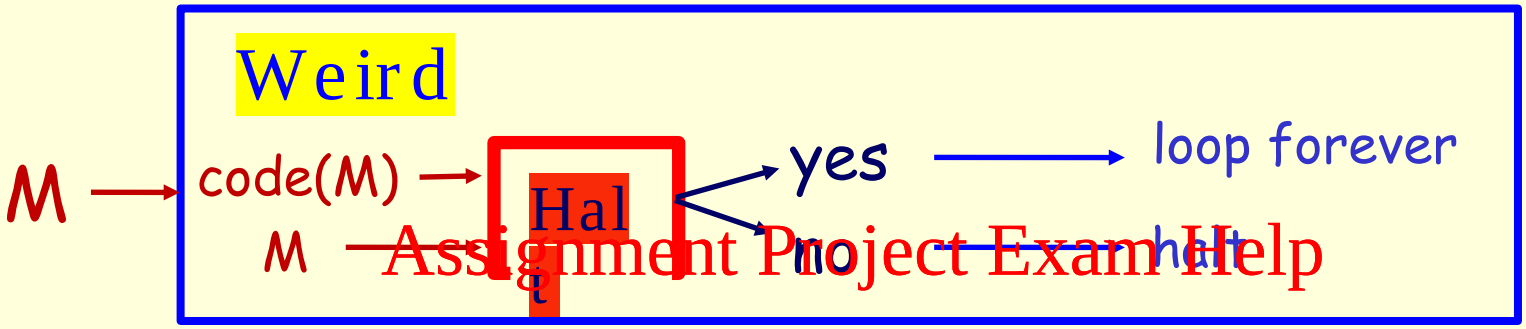
Use Halt to construct TM Weird as follows:

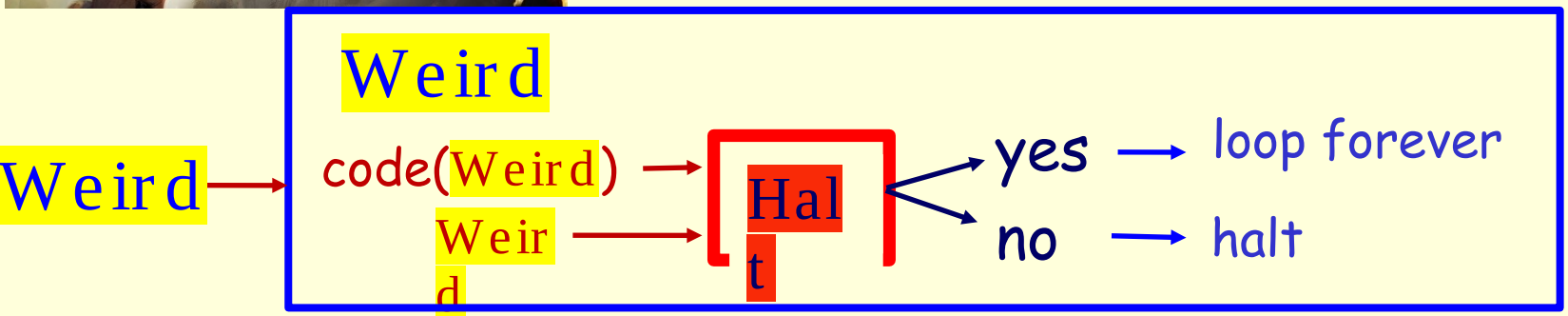
- If Halt accepts code(M)code(M)* then loop forever
- If Halt rejects code(M)code(M)* then halt

*encode M is as an input to itself

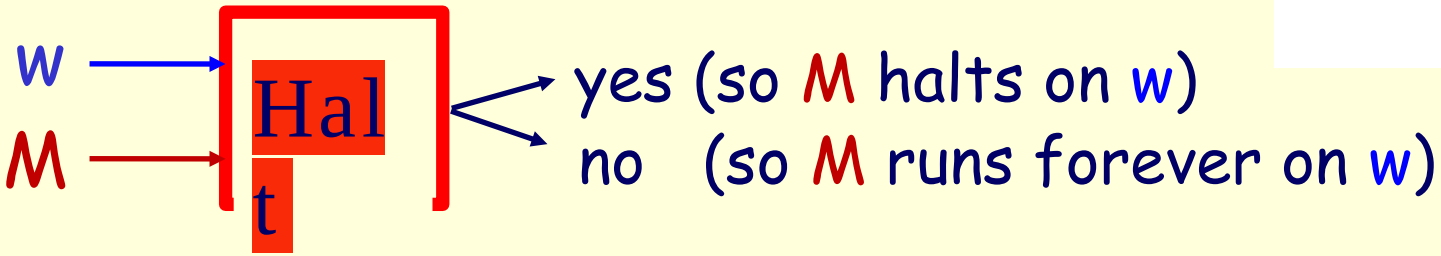
Halting problem

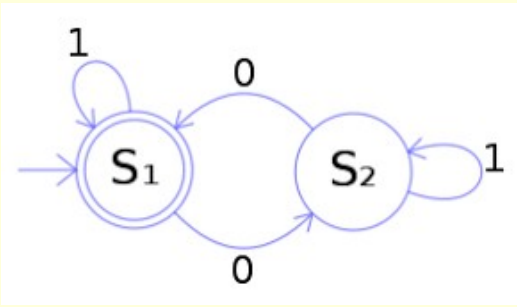






Halting problem





So if TM Halt exists, then TM Weird must also exist

Weird takes a TM M

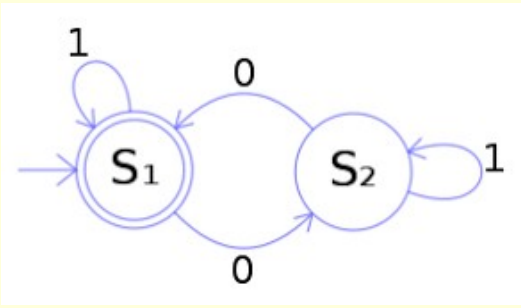
- If M halts on code(M), then Weird on code(M)
- If M does not halt on code(M), then s on code(M)

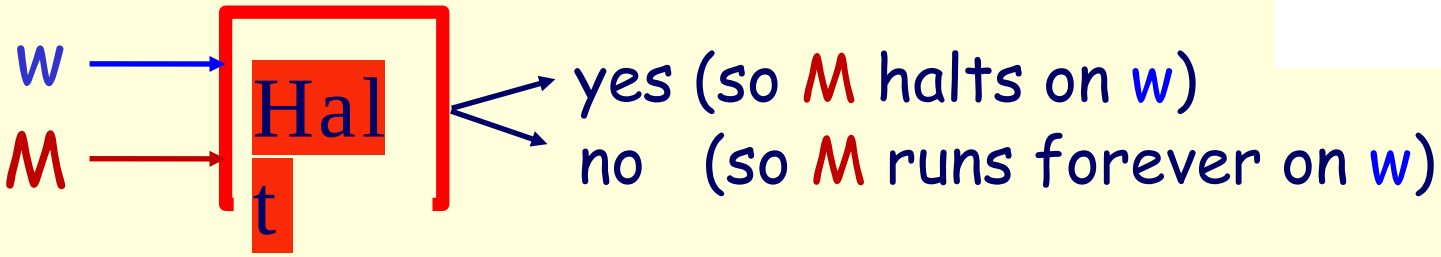
This must work for any TM M … Including when M = Weird (!!)

- If Weird halts on code(Weird), then Weird does not halt on code(Weird)
- If Weird does not halt on code(Weird), then Weird halts on code(Weird)

That's too weird! **CONTRADICTION!**

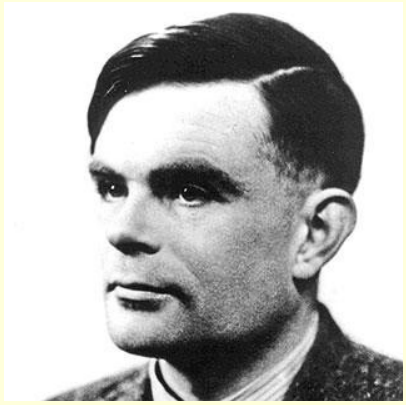
Halting problem





So our assumption is wrong, ie the TM Halt does not exist

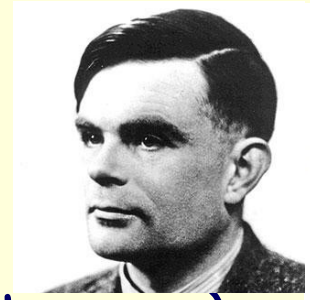
The Halting proble

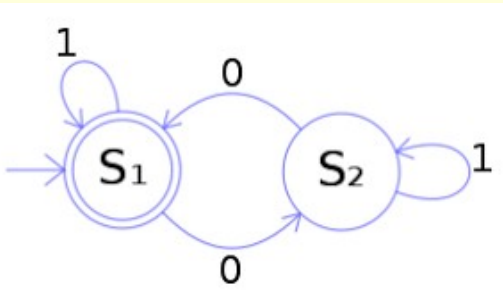


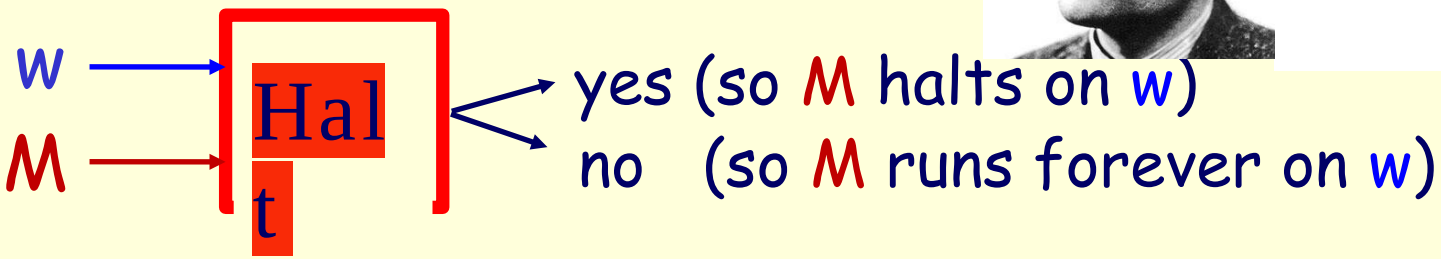
This means that there is no T

- Always halts for any M and w
- Outputs 'yes' if M halts on w
- Outputs 'no' if M does not halt on w
- (Does not produce any other output)

Halting problem







This means any attempt to solve the halting program must for some M and w either:

1. Not halt
2. Halt and output no if M halts on w
3. Halt and output yes if M doesn't h
4. Halt and produce some other output (e.g., "Dunno!")

1 is undesirable, 2 and 3 are insane!

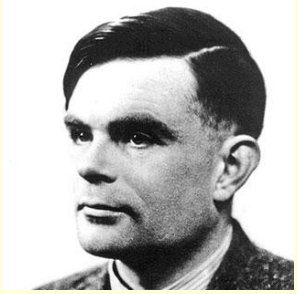
Essentially, we have to accept that an answer **"Don't know"** is sometimes unavoidable …

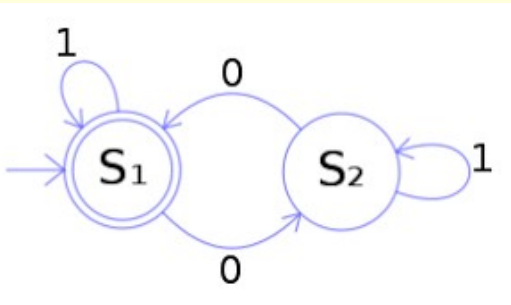
Questions?

Questions?

Questions?

Computability





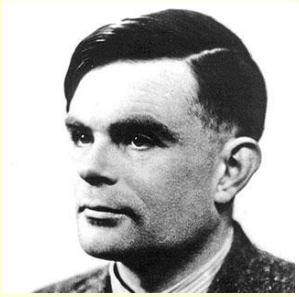
Decidable

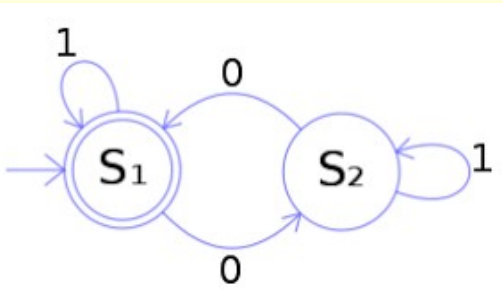
- Primality testing
- Factorisation
- Hamiltonian circuit
- 3-SAT satisfiab
- Sorting
- Travelling Salesperson
- …

Undecidable

- Does M halt on w? (halting problem)
- Does M halt on any w?
  lt on all w?
  lt on blank input?
- D halt on the same
  st
- Does M reach a particular state on input w?
- …
- Busy beaver problem
- Tile problem
- …

Computability





Let Problem *A* be a problem known to be undecidable, and Problem B with status unknown.

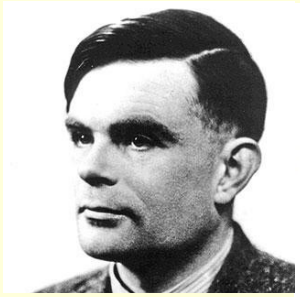
- Assume Problem B is decidable and derive a contradiction
- Reduce Problem

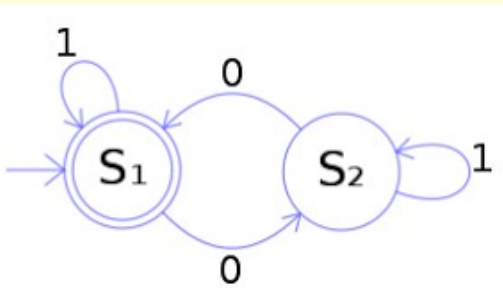
Problem reductio

1. Show that a TM for Problem B as part of a decision procedure for Problem
2. Shows Problem B decidable Problem A decidable
3. Problem A undecidable
4. So Problem B undecidable

Step 1 is the only necessary part …

Computability

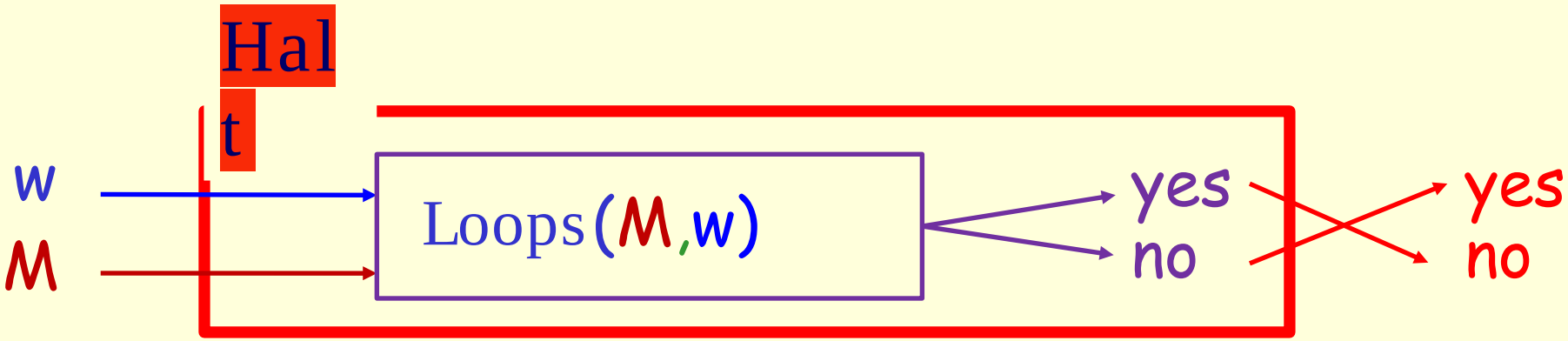




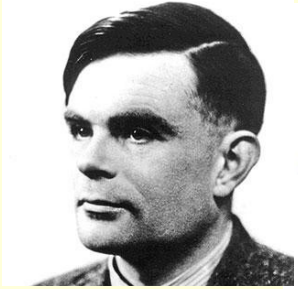
Loops problem: Does TM M run forever on input w?

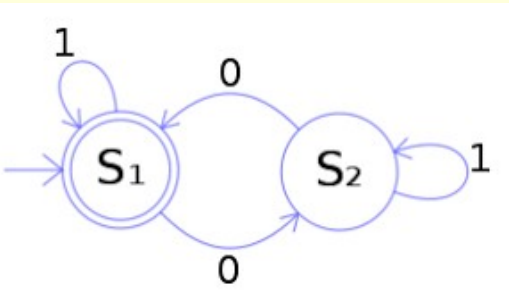
Assume there is a TM Loops which solves the Loops problem
Given M and w
1. Run Loops on M a
2. If Loops says yes
3. If Loops says no

So the Loops problem is undecidabl



Computability





Blank Tape problem: Does TM M halt on the blank tape?

Assume there is a TM Blank which solves the blank tape problem

Given M and w, comp with a blank tape

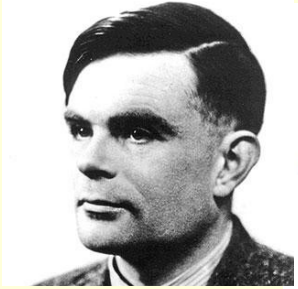
1. Writes w on the
2. Positions tape he
3. Goes to start state of M
4. Runs M on w

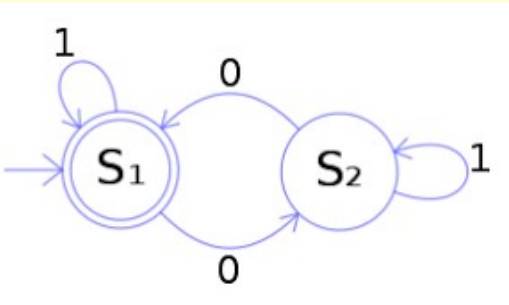
N halts on the blank tape iff M halts on w

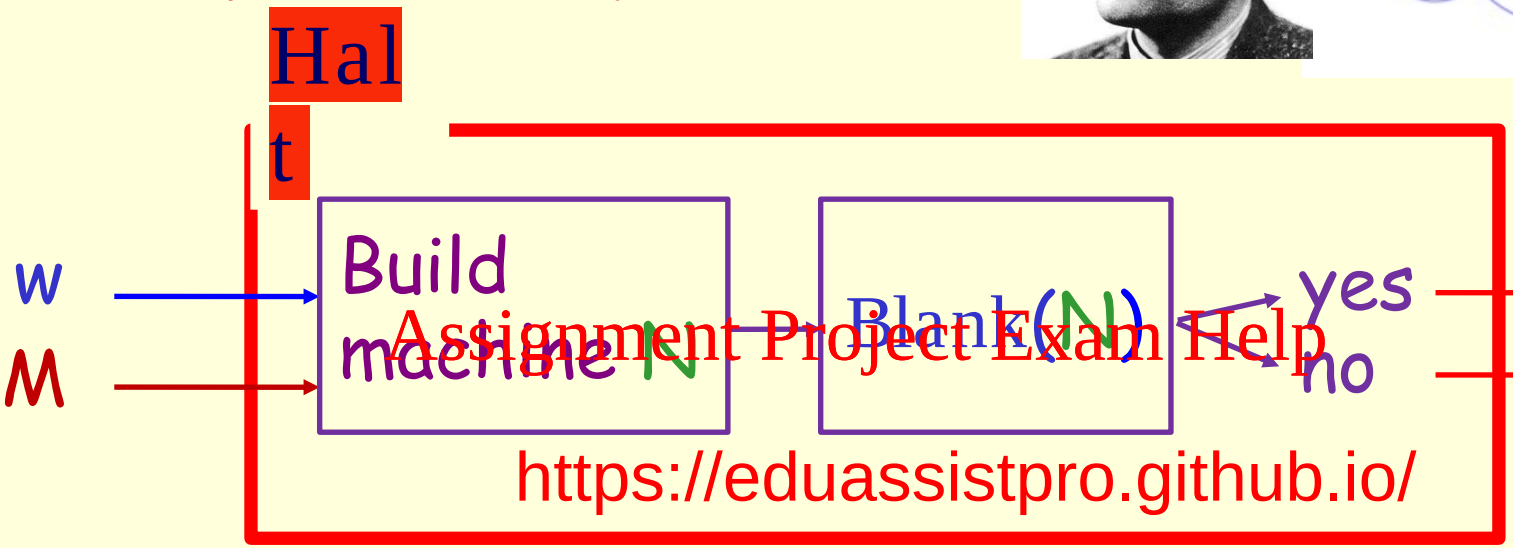
So Blank on input N will solve the Halting Problem for M on w

So the Blank Tape problem is undecidable

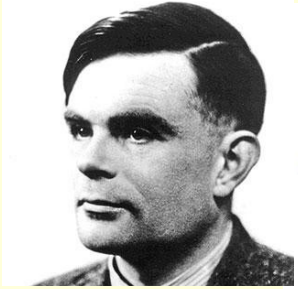
Computability

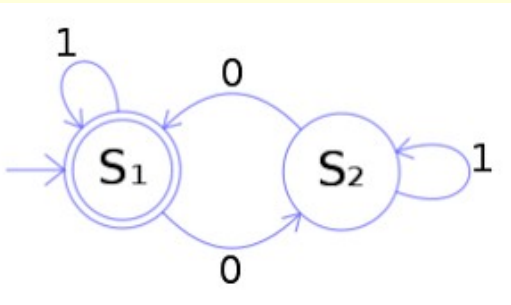






Computability





All Inputs problem: Does TM M halt on all inputs?

Assume there is a TM All which solves the All Inputs problem

Given M and w, comp

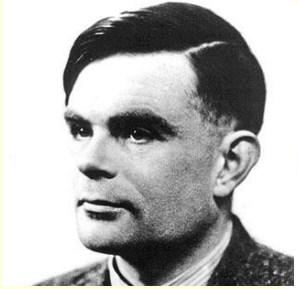
1. Erases its input
2. Writes w on the
3. Positions tape head to the first s
4. Goes to start state of M
5. Runs M on w

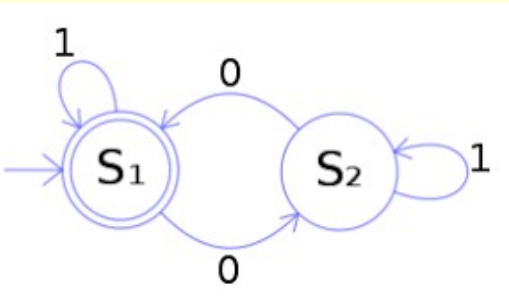
O halts on all inputs iff M halts on w

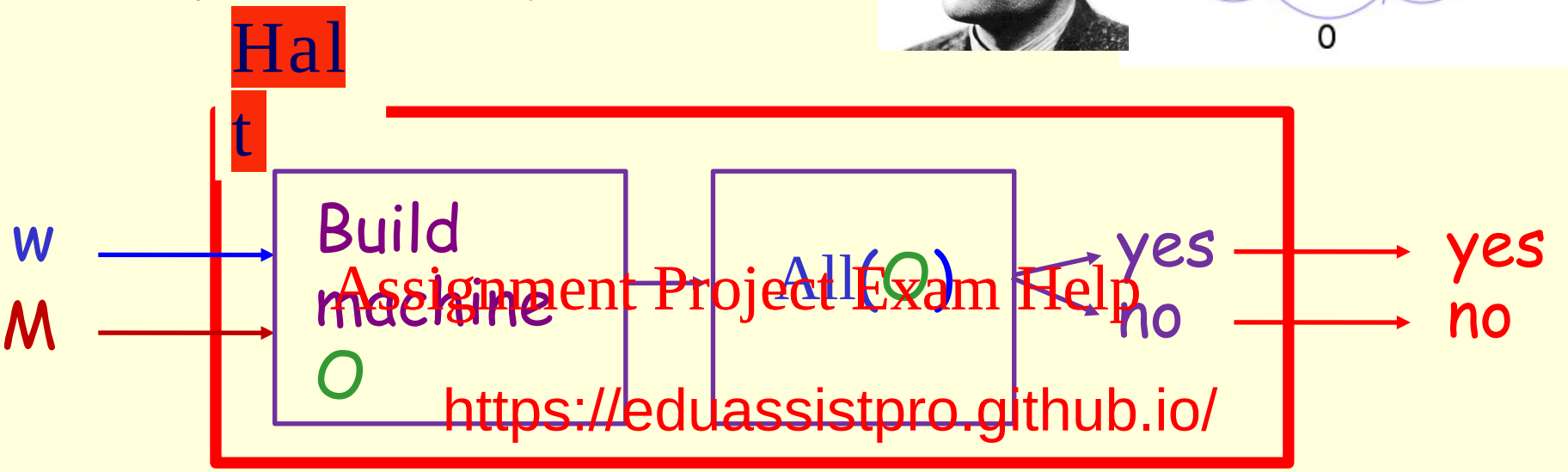
So All on input O will solve the Halting Problem for M on w

So the All Inputs problem is undecidable

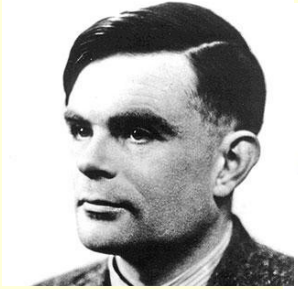
Computability

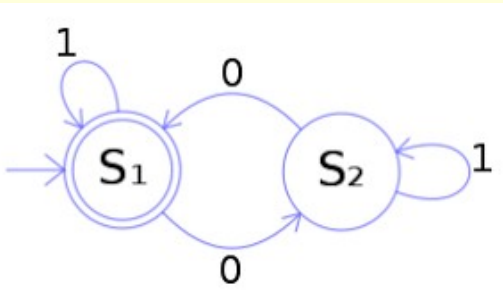






Computability





Undecidable problems include

Halting problem: Give a function f , does it halt on a given input x?
Totality problem: Give a function f , does it halt on every input x?
No input halting problem: Give a function f with no input, does it halt?
Program equivalence: Do two functions f and g always return same value?
Uninitialized variable ed before it's used?
Dead code eliminatio r get executed?
...

Does $L(G) = \Sigma^*$?
Does $L(G_1) = L(G_2)$?
For PDAs, does $L(M_1) = L(M_2)$?
Does a PDA M have the minimal number of states?

Decidable problems include

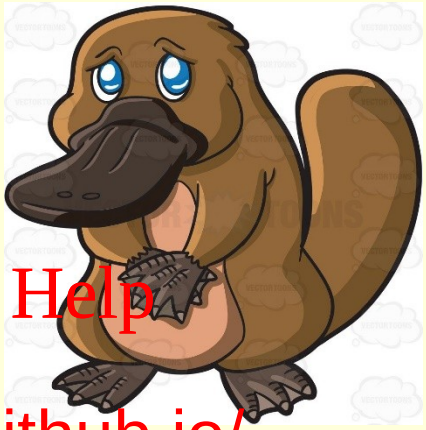
Does $L(G) = \emptyset$?
Is w $\in$ L(G)?

Questions?

Questions?

Questions?

The Platypus Game



The Platypus Game

Is the halting problem for the Platypus game decidable?

For the Platypus game the halting problem decidable is?

YES!

2^{21} possible tapes
21 x 21 possible head positions
4 x 4 possible states

$2^{21} \times 21 \times 21 \times 4 \times 4 =$ **14,797,504,512**

The Platypus Game

So there are "only" **14,797,504,512** possible configurations of the Platypus game …

Halt-platypus

P →

1. Set counte
2. Run game, incrementing cou ach turn
3. If the game halts, output y
4. If the counter exceeds 14,797,504,512, output no

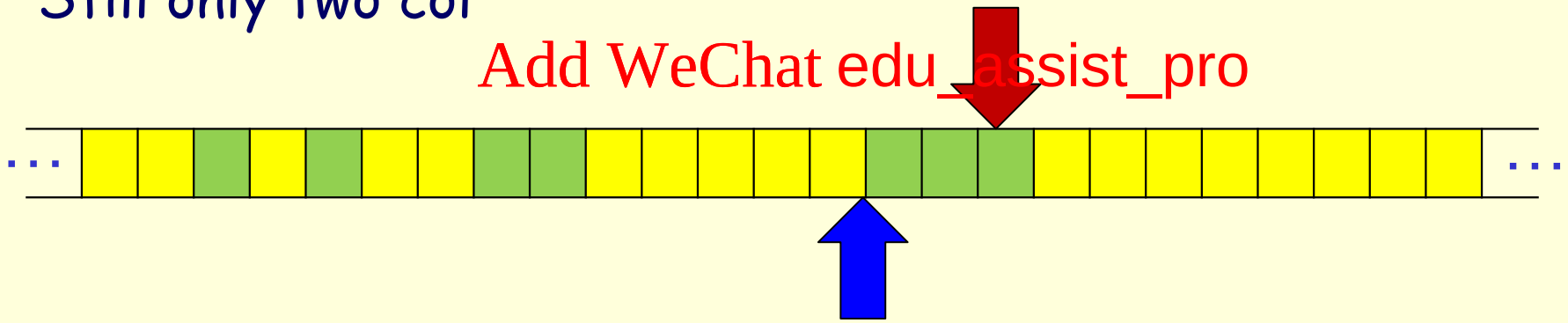
→ yes
→ no

Not very practical, but possible in principle

The Platypus Game

Generalised Platypus game

- Same as Platypus game, but with an infinite tape!
- Ghost gum and Wattle are infinitely far apart
- Arbitrary numbe
- Still only two col



Is the Halting problem for the Generalised Platypus game decidable?

The Platypus Game

Survey results (final)

The Platypus Game

Survey results (final)

The Platypus Game

Survey results (final)

The Platypus Game

Survey results (final)

Do you have any other suggestions or comments about tournaments?

Maybe instead of a champion's league with random matchups there are brackets to determine which machine you can fight against (like an elo rating system to determine which players are suitable matches to play with)..

Sounds interesting too!

A rankings system such as used by the Cha e suggestion is what is used by a majority of online and real wo haps we can apply what is learned through stimulating a Champion's league to the real world.

Sure! To a real platypus? ☺

The Platypus Game

Survey results (final)

Do you have any other suggestions or comments about tournaments?

Is there any application in creating tournaments that become more ranked then more tournaments that are completed? For example, a single tournament would show how well a set went against each single player, if they were to lose in their round. However, r score but a different pairing didn't win to as progress unfairly.

So completion rate is important?

Champion's league seems like an interesting way to rank the machines! :)

Ok!

The Platypus Game

Survey results (final)

Are there any other suggestions or comments about scoring?

For the tiebreaker, maybe let the match be that reaches the green cell with platypus be the winner.

Interesting --- so we r

I would like a change to the scoring syste he player to be rewarded for taking more risk.

Similar, presumably …

Simple is better, and changes to rules would not change the outcome too much.

No change is easy to do!

The Platypus Game

Survey results (final)

Are there any other suggestions or comments about scoring?

If we were to use a ranking system, it makes most sense to include a tiebreaker such that th

Sure, although we can r

If a grid had could turn a token green wit it this would be a terminating play. That should garner a pen

There may also be a play that has more changes to green than to yellow, which would also be more likely to trigger an end game, and on the flipside the more yellow would mean the player does not terminate at all which is almost cheating, difficult to say without the context of play...

So certain machines should be outlawed?

The Platypus Game

Survey results (final)

Are there any other suggestions or comments about scoring?

Bonus points for reaching the trees.

Ok!

The Platypus Game

Survey results (final)

Are there any other suggestions or comments about rule changes?

If we have a larger number of cells, then there might be some bonus for reaching a tree since it

Good thought. Presuma *ee options should be linked like this.*

Shorter games means faster progression, I with happily.

A short game is a good game!

Shortening or lengthening the maximum number of turns before a game ends (currently 100),Larger or smaller number of cells (eg 11? 31? 101?)

Sure.

The Platypus Game

Survey results (final)

Are there any other suggestions or comments about rule changes?

Machines changing somehow during play would make it harder to guess the outcome of matc an ideal winner would look like, however wo ally interesting, with emerging strategies

Machines change somehow during pla

Sure. How would they change?

Shortening or lengthening the maximum number of turns before a game ends (currently 100)

Sure. More responses yet to be analysed!

Questions?

Questions?

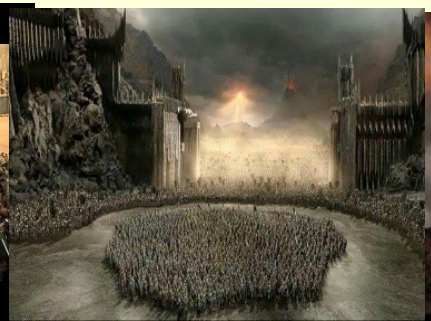
Questions?

That's it!

I am out of here!

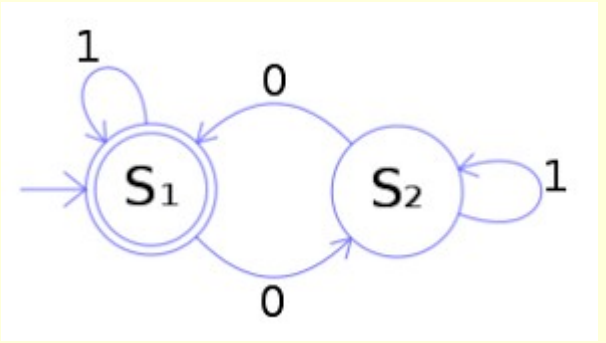
The lecture ed.

Break time! (We resume when all the pictures are gone! This will take 3 minutes!)



I AM BACK!

Marking



Why did I lose marks for …??

Where did you last see them?

Ar https://eduassistpro.github.io/

Perhaps you shou m in at night …

Never feed them after midnight!

…

Alternative Scheme?

Troll	Dreadful	Poor	Acceptable	Exceeds Expectations	Outstanding

Outstanding – CONGRATULATIONS! Your exemplary powers of deduction and a formidable knowledge of the inner workings of the magical world reveal you to be a witch or wizard of genuine skill and learning.

Exceeds Expectations – rformance!

Acceptable – demonstrat

Poor – Alas – we regret to inform you that you h iled. This may have been due to factors outside your control (eg: polterge , examination nerves or a malfunctioning quill.) Please do not disconsolate.

Dreadful – We are sorry to inform you that you have failed.

Troll – You would appear either to have abandoned the test due to factors outside your control (eg, earthquake, poltergeist attack), or else you are a troll, in which case you are to be congratulated on being able to use a computer and have achieved the grade of O.F.T. (Outstanding for Trolls).

Alternative Scheme?

Troll	Dreadful	Poor	Acceptable	Exceeds Expectations	Outstanding

Outstanding – CONGRATULATIONS! Your exemplary powers of deduction and a formidable knowledge of the inner workings of the magical world reveal you to be a witch or wizard of genuine skill and learning.

Exceeds Expectations – rformance!

Acceptable – demonstrat

Poor – Alas – we regret to inform you that you h iled. This may have been due to factors outside your control (eg: polterge , examination nerves or a malfunctioning quill.) Please do not disconsolate.

Dreadful – We are sorry to inform you that you have failed.

Troll – You would appear either to have abandoned the test due to factors outside your control (eg, earthquake, poltergeist attack), or else you are a troll, in which case you are to be congratulated on being able to use a computer and have achieved the grade of O.F.T. (Outstanding for Trolls).