Mike Stannett (Sheffield University)
Workshop on Turing's Legacy in Mathematics and Computer Science
BMC 2012, Canterbury, Kent, 17 April 2012

Computing via Physical Theories

What can be computed?

Depends what a "computer" is

- What inputs can it process?
- What actions are permitted?
- How are actions scheduled?
- How do you define the output?

General machine model

initialise	: Config
step	: Config $\to$ Config
run	: Time $\to$ Config

run 0 = initialise

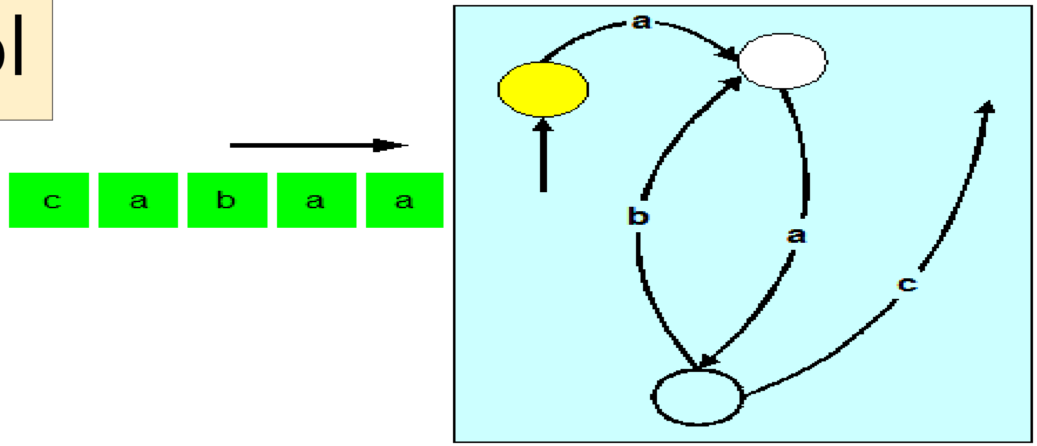
Still need to define

- Config, Time
- initialise, step
- run t for arbitrary t : Time

Finite state machine

Config	= String x State
Time	= $\mathbb{N}$
initialise	= (inputString, initialState)
run (t + 1)	= step (run t)

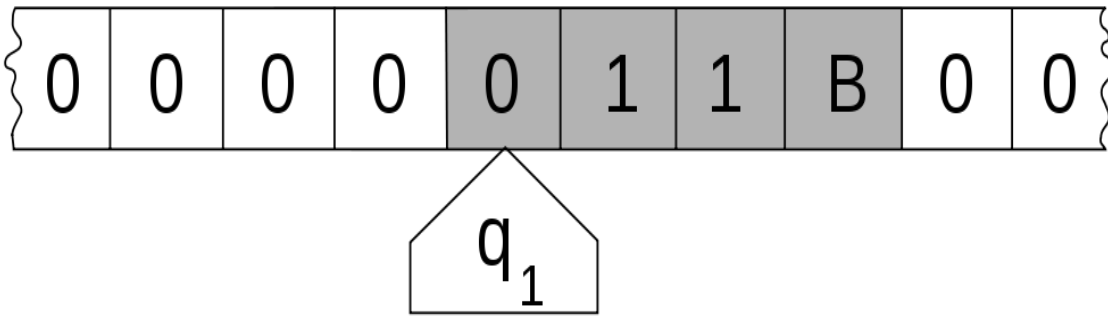
String = $\mathbb{N} \rightarrow$ Symbol



Turing machine

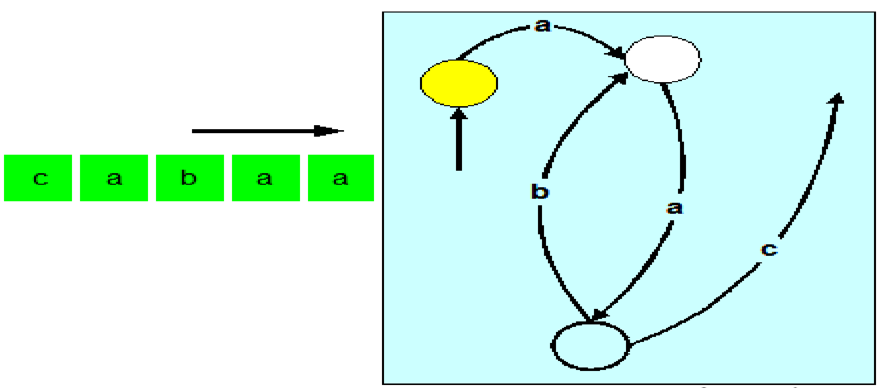
Config	$= \text{Tape} \times \mathbb{Z} \times \text{State}$
Time	$= \mathbb{N}$
initialise	= (inputTape, 0, initialState)
run (t + 1)	= step (run t)

Tape	$= \mathbb{Z} \to \text{Symbol}$

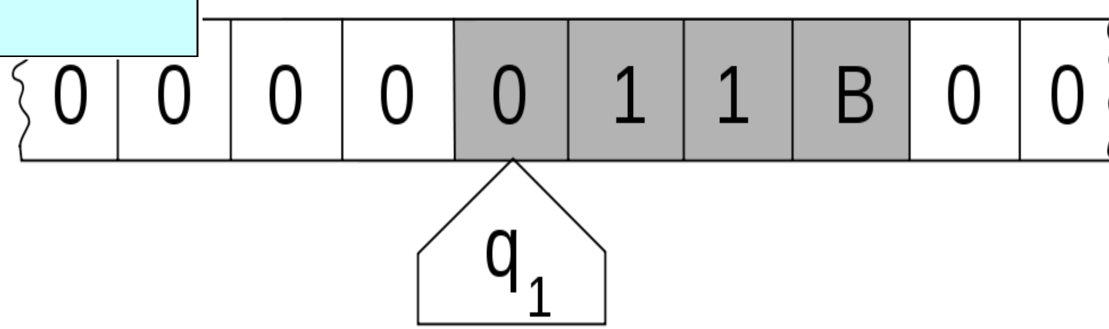


General machine

World	= Space → Symbol
Config	= World x Space x State
initialise	= (initWorld, initFocus, initState)



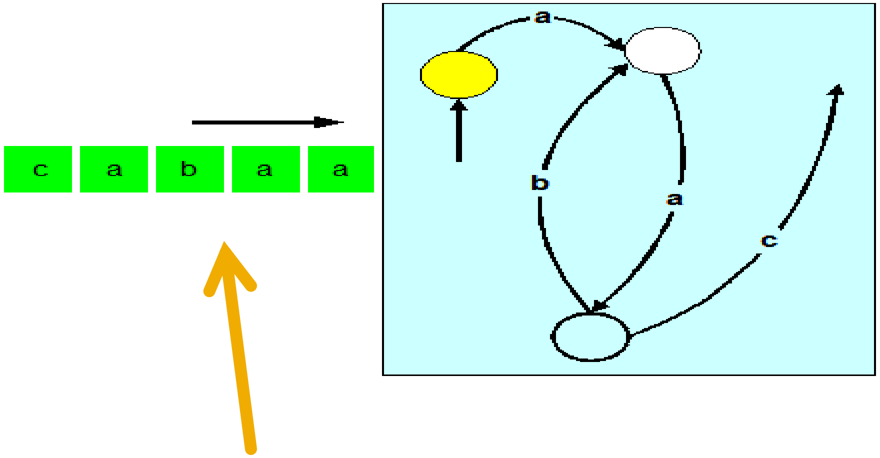
shared structure



FSM revisited

World = Space $\rightarrow$ Symbol

Config = World x Space x State



Space = $\mathbb{N}$

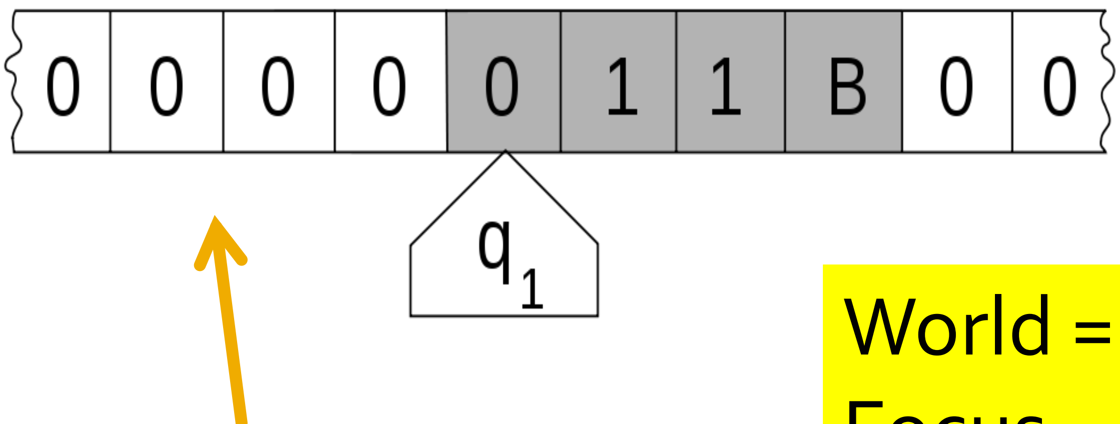
World = string

Focus = head of string

TM revisited

World = Space $\rightarrow$ Symbol

Config = World x Space x State



Space = $\mathbb{Z}$

World = current tape

Focus = head position

General model

initialise	: Config
step	: Config → Config
run	: Time → Config

next : Time → Time

run (next t) = step (run t)

Terms in red still need to be defined

World	= Space → Symbol
Config	= World x Space x State

Identifying the output

extract : [Config] → Output

- Initialise the machine
- Run the program
- The output may depend on one or more of the configurations generated during execution

output = extract [run t | t : Time]

What can be computed?

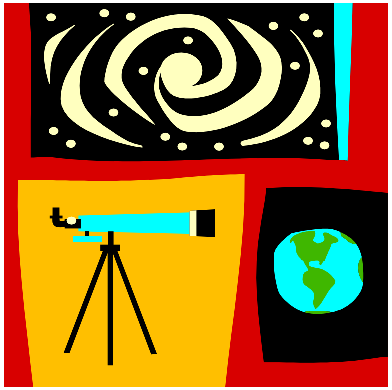
It depends on how we define various types and functions
and whether they can be implemented

Space Time Symbol Output

World = Space → Symbol
Config = World x Space x State

step : Config → Config
extract : [Config] → Output
run : Time -> Config

In particular, can we implement the structures Space and Time ?

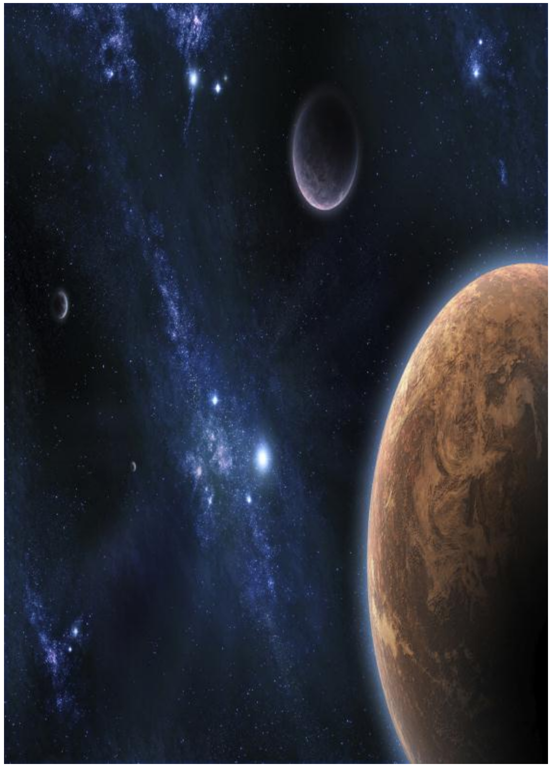


What can be computed?

Can we implement the structures Space and Time ?

This cannot be answered in absolute terms, because we don't know enough about the universe.

We can only discuss semantics relative to some particular theory of the physical universe



Newtonian physics?

FSM: Space = $\mathbb{N}$ Time = $\mathbb{N}$

TM: Space = $\mathbb{Z}$ Time = $\mathbb{N}$

The Universe is continuous and infinite in space and time, so we can easily represent a $\mathbb{Z}$-shaped collection of boxes and an $\mathbb{N}$-shaped sequence of clock ticks.

BEWARE

we cannot guarantee the result of <u>any</u> program

Relativistic physics?

FSM: Space = $\mathbb{N}$ Time = $\mathbb{N}$

TM: Space = $\mathbb{Z}$ Time = $\mathbb{N}$

The Universe need not be infinite.

Not a problem if we can embed $\mathbb{N}$ (and $\mathbb{Z}$) in the representations of time and distance, eg if we model spacetime as a real manifold.

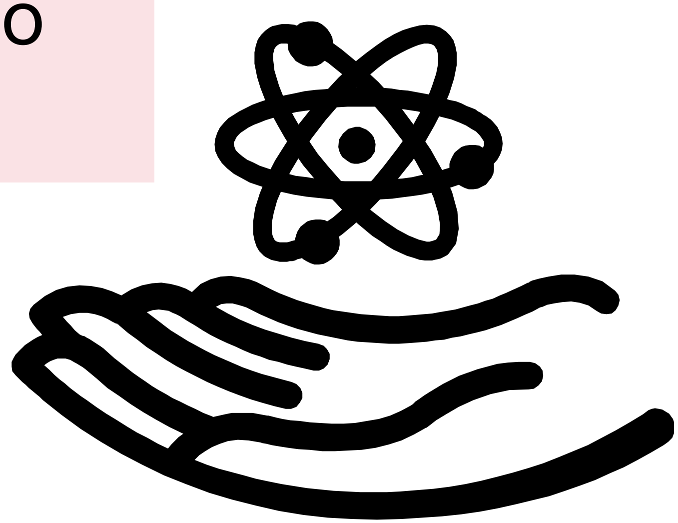
Question: Can we sometimes guarantee correctness?

Quantum gravity?

FSM: Space = $\mathbb{N}$ Time = $\mathbb{N}$

TM: Space = $\mathbb{Z}$ Time = $\mathbb{N}$

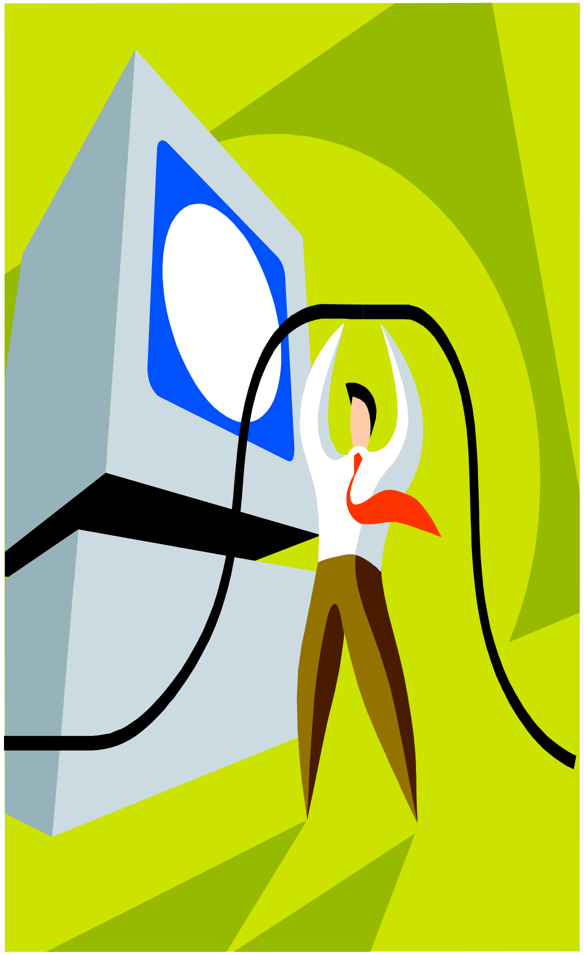
The Universe may or may not be infinite in space and time, and it may or may not be possible to embed $\mathbb{N}$ (or $\mathbb{Z}$).



Is Turing computation the ultimate?

For TMs to be implemented, it must be possible to embed $\mathbb{N}$ and $\mathbb{Z}$.

Newtonian physics allows this, other versions may not, so is Newtonian TM-computation the best we can do?



The extract function also matters!

extract : [Config] → Output

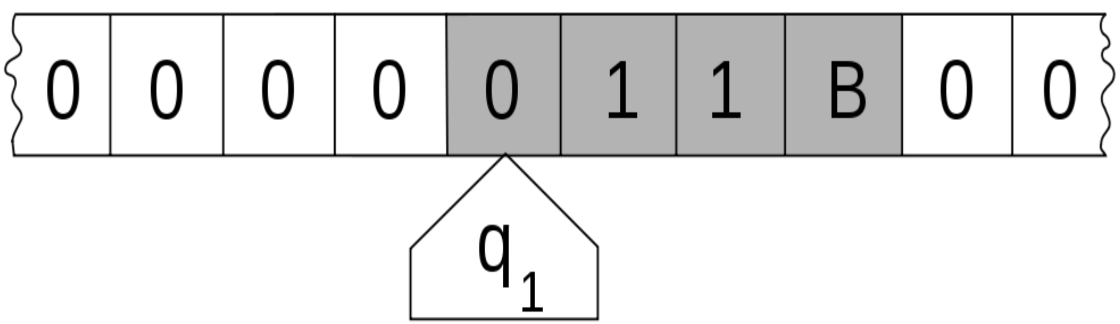
- Initialise the machine
- Run the program
- The output may depend on one or more of the configurations generated during execution

output = extract [run t | t : Time]

TM output extraction

extract : [Config] → Output

extract cfgs
is defined only when
cfgs is eventually constant



More powerful extraction?

extract : [Config] → Output

can extract cfgs be implemented when cfgs isn't eventually constant?

If we could compute [run t | t : Time] in finite physical time, we could compute eg

output = $\lim_{t\,:\,\mathrm{Time}}$ (extract [run s | s < t])

Can it be done?

run : Time -> Config calls
step : Config -> Config where
Config = World x Space x State

COMPUTER

- The effect of executing run depends on where the machine is in Space and Time

extract : [Config] -> Output

OBSERVER

- Extract is defined on completed sequences. It doesn't depend on where or when it is computed.

Separating machine from observer

- Can we separate the observer O from the machine M?
- Can it be done so that O sees the whole of M's configuration sequence?

<u>QUITE POSSIBLY YES!</u>

See eg papers by Mark Hogarth
for a discussion of computation
in Malament-Hogarth spaces

Example

- Use a massive slowly-rotating black hole [Németi et al]

- NB. If the TM can be implemented, so can this!

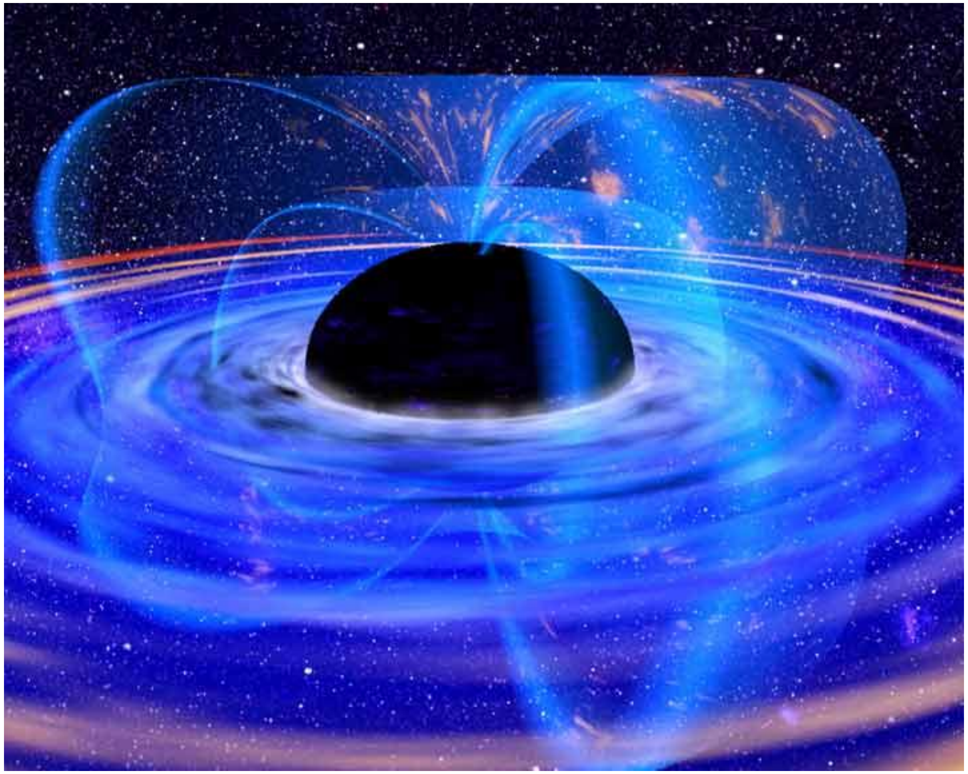


Image source: NASA / ESA / SSM-Newton

Can computability tell us anything about physics?

Even if next: Time -> Time is defined, it doesn't follow that time is ordered. It could be cyclic.

- Consider embedding a finite sequence along a CTC [closed timelike curve].
- Assume that whenever a system returns to a point in the CTC, it is in the same configuration each time.

NB. CTCs may not exist!

Stannett (2011) [arXiv:1103.1127v1]

CTCs can store information?

Launch a TM along a CTC of length n. After n statements are executed, the machine is back in the initial state.

- Computation typically loses information (eg we can't usually deduce x and y if we only know their product)
- The information must be stored somewhere so that the initial state can be reconstructed
- It is presumably stored in the CTC itself

A CTC's storage capacity is low?

Launch a TM along a CTC of length n. After n statements are executed, the machine is back in the initial state.

- Suppose the TM is loaded with the program

  `x = 0; while (true) {x = 1}`
- This doesn't re-initialise after n steps but does halt
- This program cannot be run on the machine, but the program would be runnable if the CTC were big enough to hold both the program and the biggest resulting tape
- Therefore: the CTC isn't big enough to store both this program and its resulting tapes

Choice of number system

- Suppose we use a first-order theory of physical spacetime
- Even if we intend using $\mathbb{R}$ as the underlying number field, we can't be sure about it
- There are other ordered fields for which the set of first-order theorems is identical to those over $\mathbb{R}$

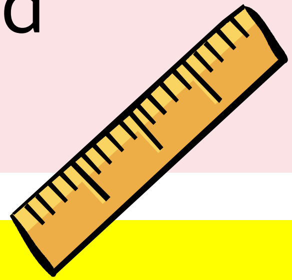
Topological arguments may fail

- Some of these ordered fields contain infinitesimals
- They are necessarily of characteristic zero (since ordered), so we can define "integers" (n = 1 + … + 1)
- Convergence doesn't always work as expected, eg $^1/_n \to 0$ can fail

Thinking about measurement

- Suppose observed values always to belong to a field D…
- … but the underlying field of the physical model is actually F

- How are D and F related?
- How does this affect the validity of measured outputs?



Example. We think of spacetime as a real manifold, but all measurements seem to be in $\mathbb{Q}$

Summary

- Computation is a physical process
- What can be computed depends on the underlying model of physics
- Some (entirely reasonable) models of physics seem to have the property that "if Turing computation is feasible, so is super-Turing computation"
- Thinking about computation can suggest results concerning cosmology
- Logical choices also matter

Thanks :)

Feel free to email me for details of papers, etc

m.stannett@dcs.shef.ac.uk