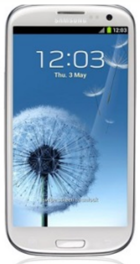
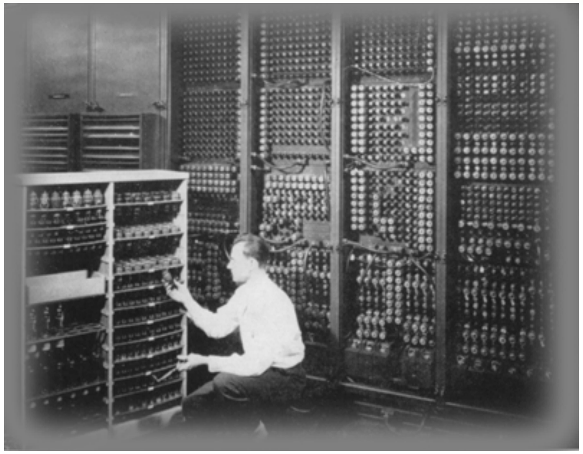
When does a slime mould compute?

Memorial University
24 July 2014

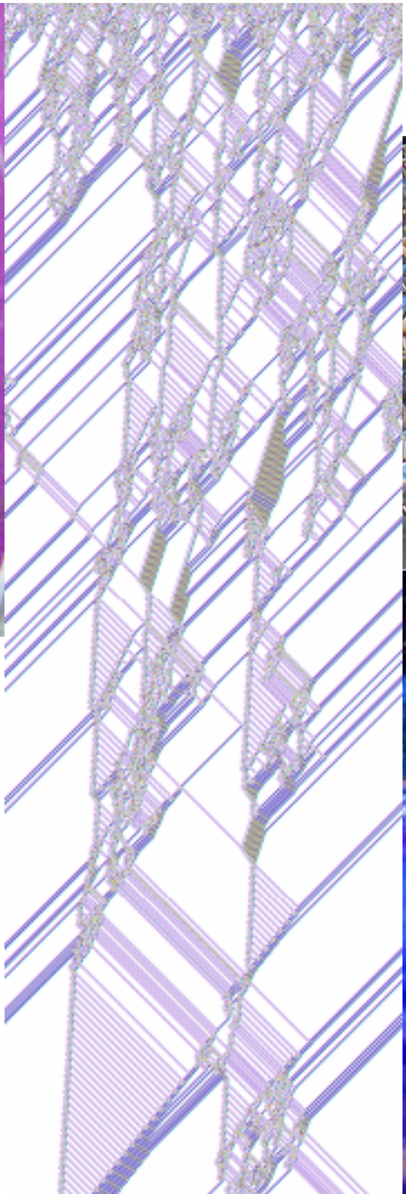
Susan Stepney
Non-Standard Computation Research Group
Department of Computer Science
THE UNIVERSITY of York

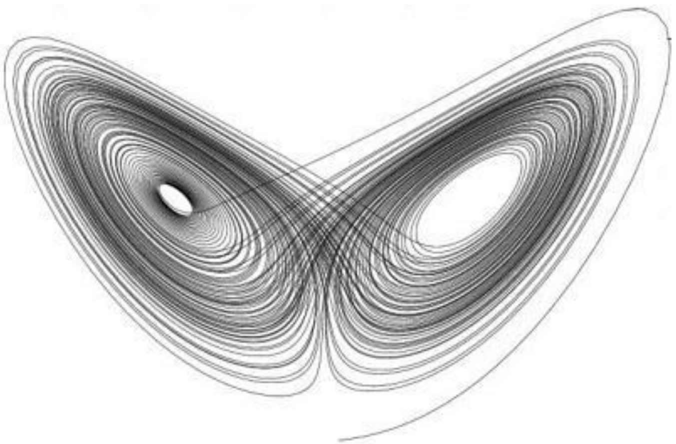
if I say “computer”, you probably think …





but do you think …?





you shouldn't be surprised

Antikythera, c.80CE

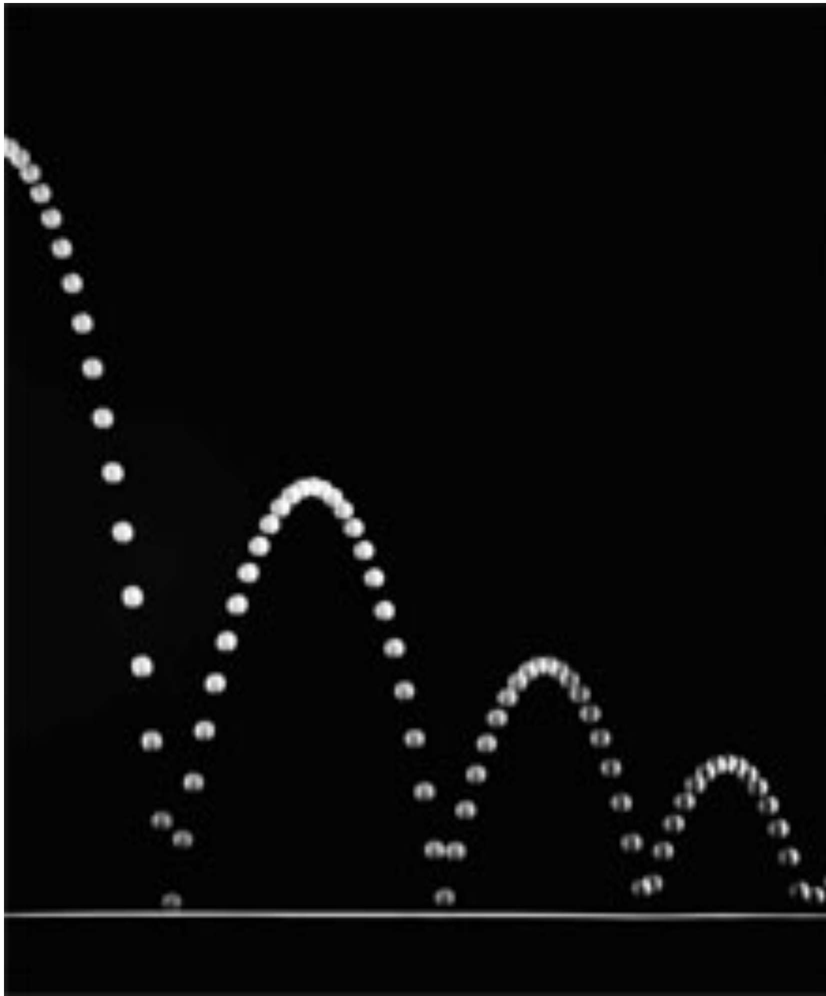
but what about these?

http://www.itinformatix.in/developer/StoneStore/dev_admin/p_image/Cuff_Hill_logan_stone_2.JPG
http://antifan-real.deviantart.com/art/Grand-Universe-17189369

what is a computer?

- What does it mean to say that some physical system is "running" a computation?
  - as opposed to just "doing its thing"

- When does a physical system "compute"?



three steps to computers

first we need to answer:

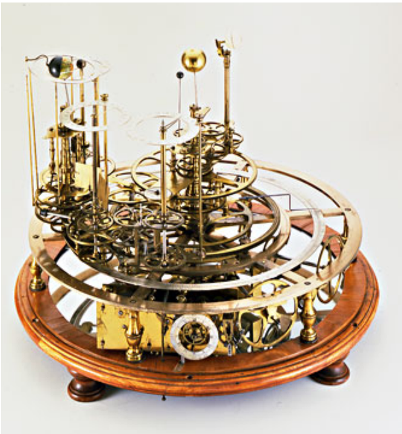
- what is **science**?
  - how we represent a physical system as an "abstract model"

this will let us answer:

- what is **engineering**?
  - how we instantiate an abstract model as a physical system

and then this will let us answer:

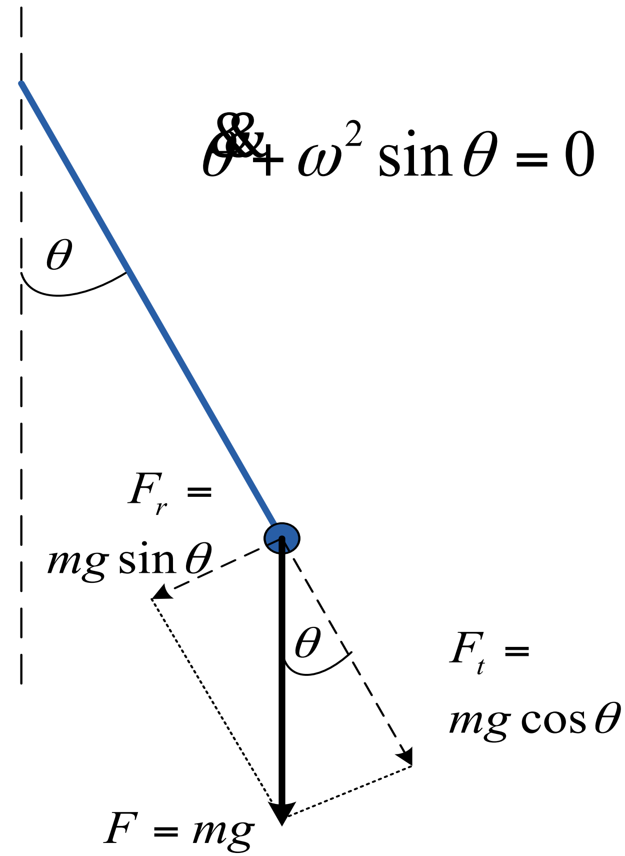
- what is **computing**?
  - how we instantiate a computational model in a physical system



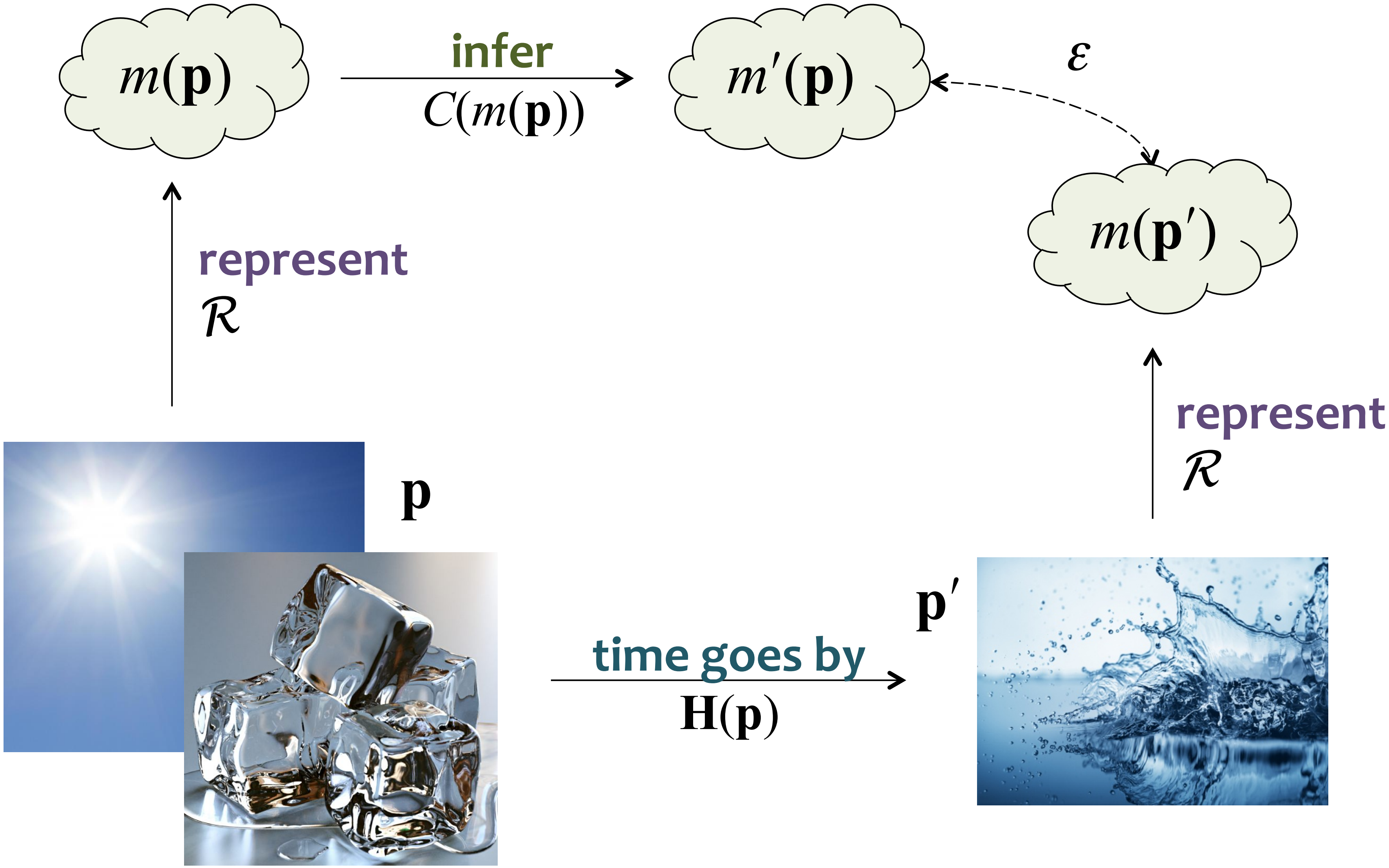
science

physical system → **represented** → abstract model

physical system ← **instantiated** ← abstract model



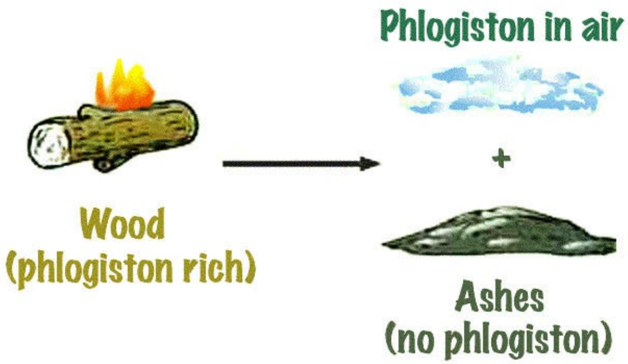
science



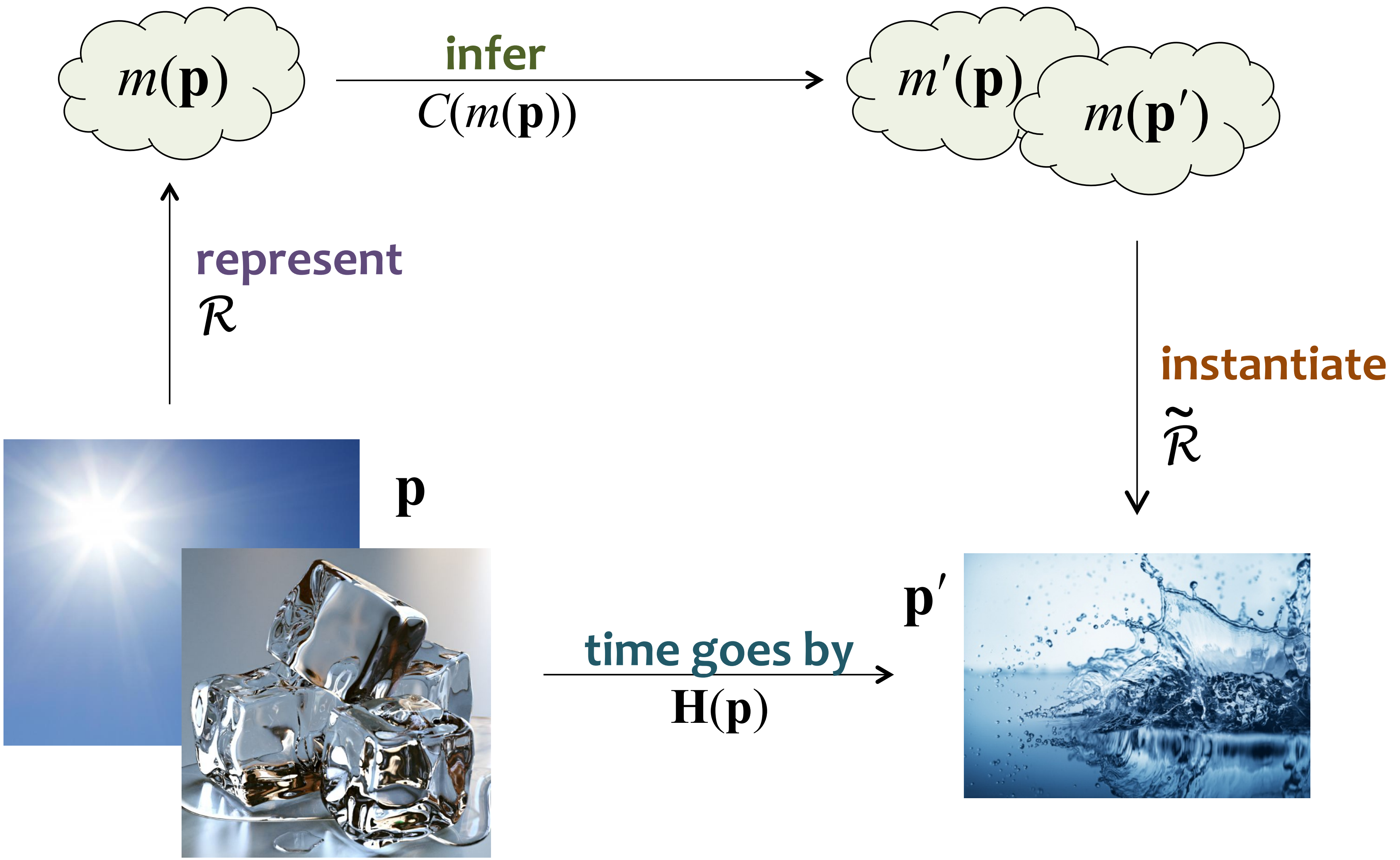
$m(\mathbf{p})$
infer
$C(m(\mathbf{p}))$
$m'(\mathbf{p})$
ε
$m(\mathbf{p}')$
represent
$\mathcal{R}$
$\mathbf{p}$
represent
$\mathcal{R}$
time goes by
$\mathbf{H}(\mathbf{p})$
$\mathbf{p}'$

science

- a “good” theory makes ε “small enough”
  - among other things…
- if ε is too large, change the theory
  - reality trumps theory
- a theory is a *model* of reality
  - models are *always* approximations
  - approximations *break down* outside the model’s valid domain
- a good theory allows prediction *without* needing a “reality check” every time a prediction is made
  - within the domain where the approximations hold

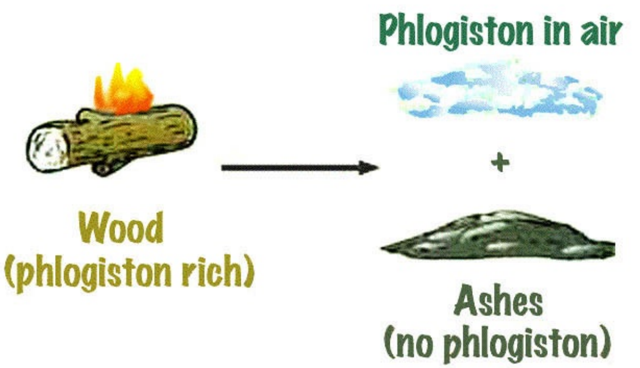


good scientific theory : prediction

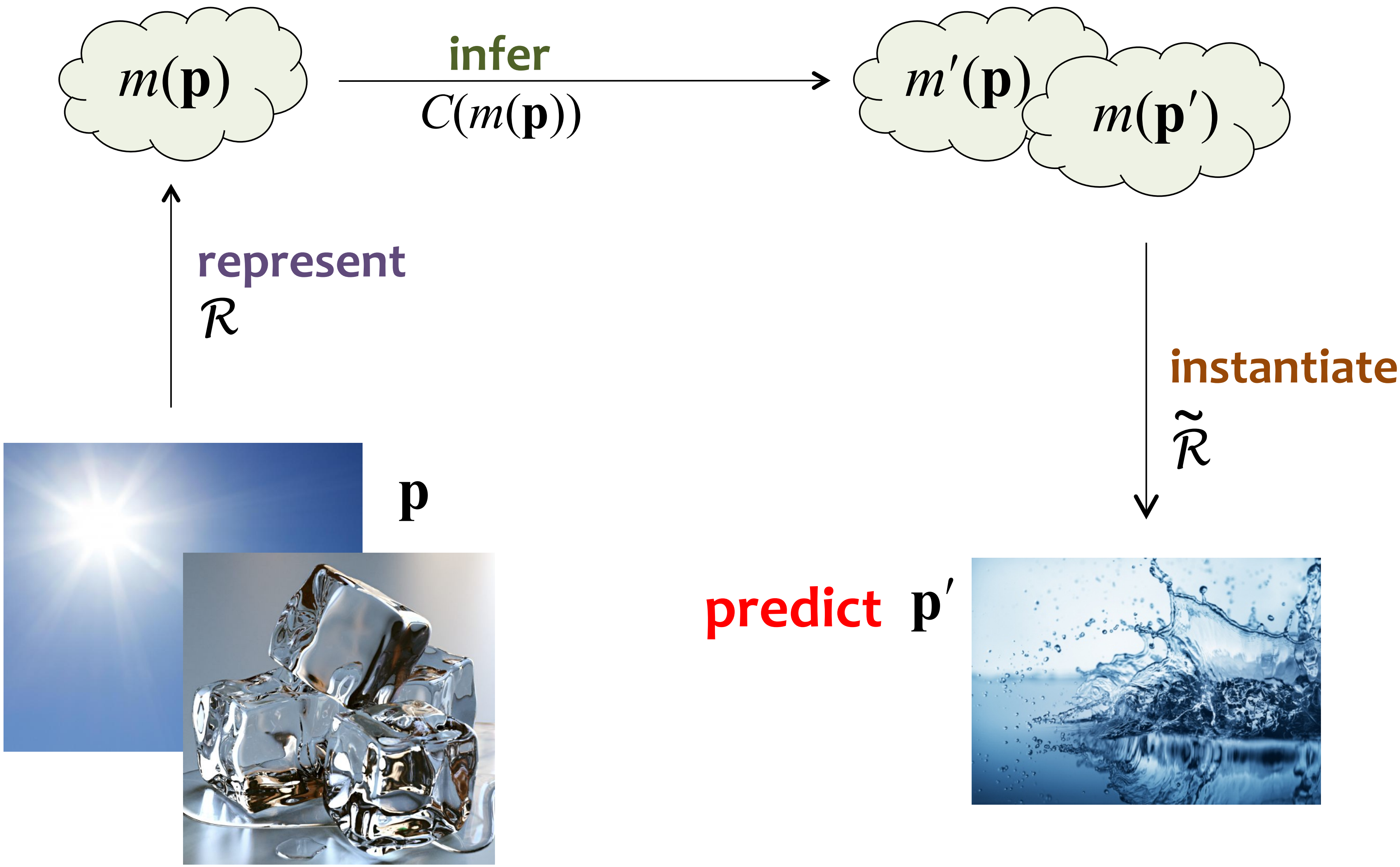


science

- a “good” theory makes ε “small enough”
  - among other things…
- if ε is too large, change the theory
  - reality trumps theory
- a theory is a *model* of reality
  - models are *always* approximations
  - approximations *break down* outside the model’s valid domain
- a good theory allows prediction *without* needing a “reality check” every time a prediction is made
  - within the domain where the approximations hold



good scientific theory : prediction



definition

here, prediction is :

using an **abstract** dynamics $\boldsymbol{C}$

of a ***well-characterised*** physical system

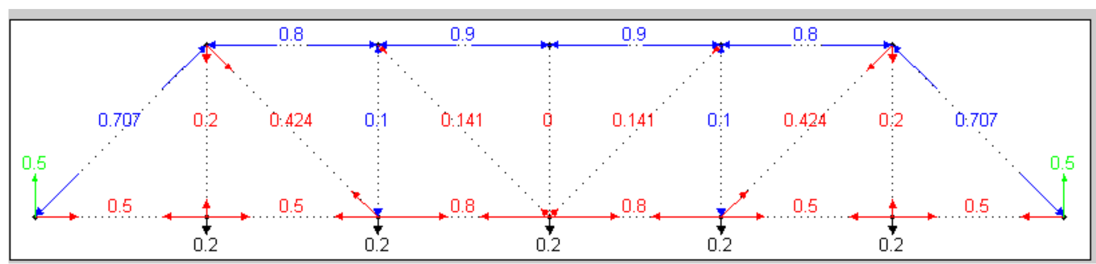
to ***infer*** its **physical** dynamics $\mathbf{H}$

(subject to a representation $\mathcal{R}$)

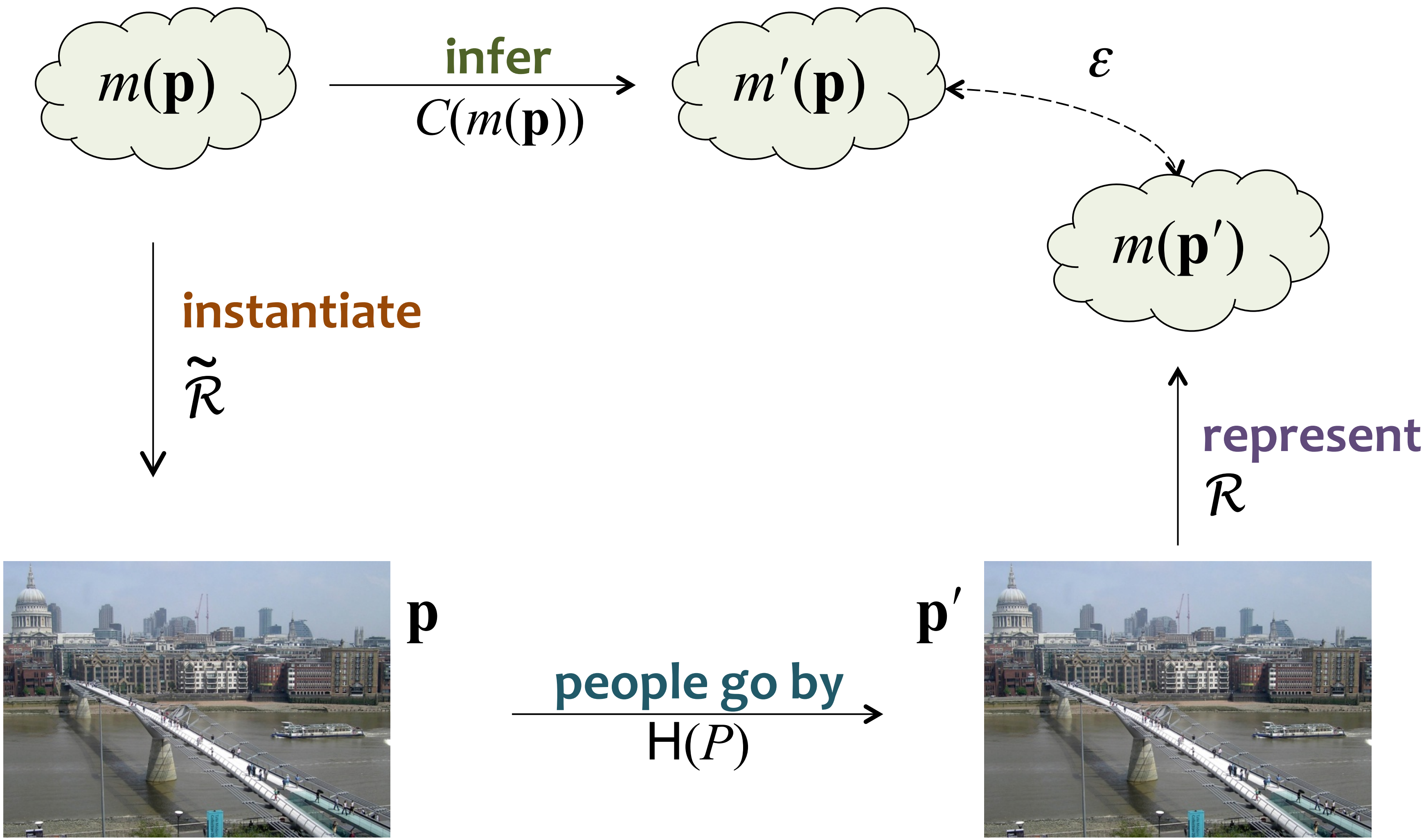
technology / engineering

physical artefact → **represented** → engineering model

physical artefact ← **instantiated** ← engineering model

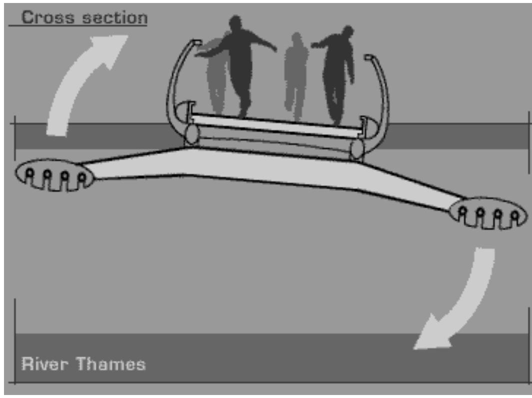


technology / engineering



engineering

- a “good” instantiation **p** makes ε “small enough”
  - among other things…
- if ε is too large, **p** needs to be changed
  - the (desired) model trumps reality
- a theory is a *model* of reality
  - models are *always* approximations
  - approximations *break down* outside the model’s valid domain
- a good instantiation allows use *without* needing a “theory check” every time the system is used
  - within the domain where the approximations hold



well-engineered technology

$m(\mathbf{p})$

infer

$C(m(\mathbf{p}))$

$m'(\mathbf{p})$ $m(\mathbf{p}')$

instantiate

$\tilde{\mathcal{R}}$

represent

$\mathcal{R}$

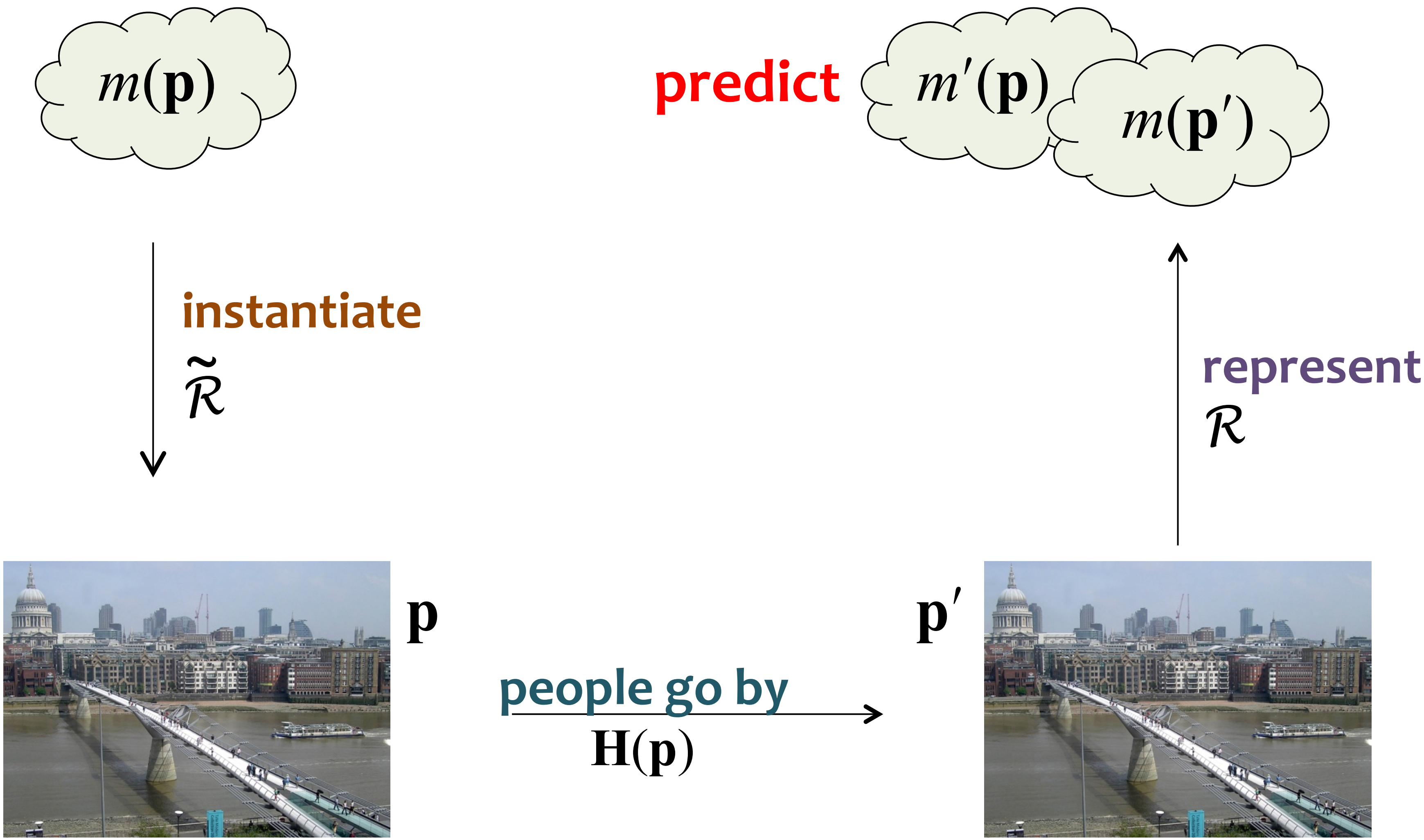
$\mathbf{p}$

people go by

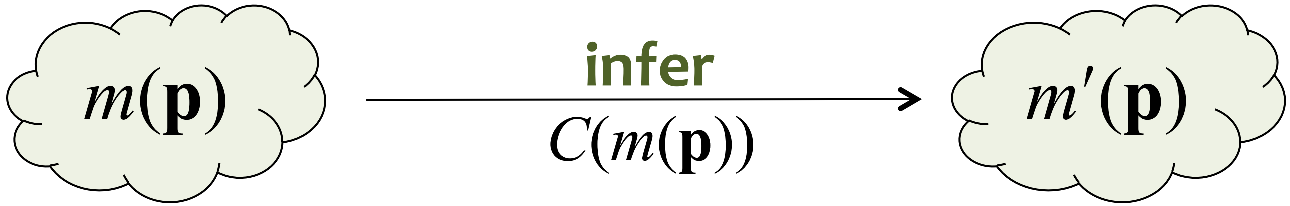
$\mathbf{H}(\mathbf{p})$

$\mathbf{p}'$

well-engineered technology



inferring



- who, or what, is doing the inferring?
  - a long complicated calculation
  - done with pen and paper …
  - … or with a computer!

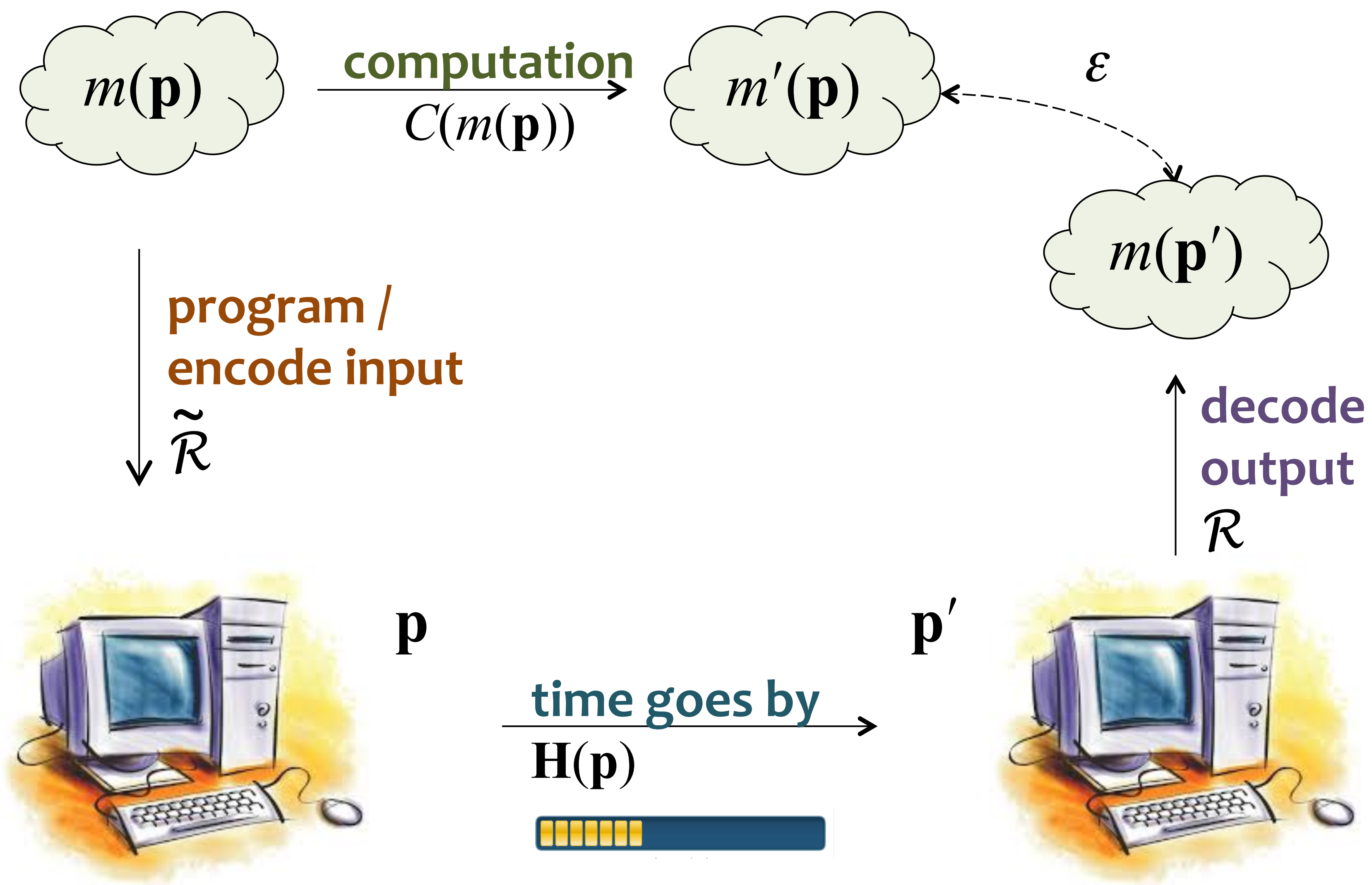
computing

computer **represented** → computation

computer ← **instantiated** computation

```
for t = 0 to p − 1 {each automaton state in the ash} do
  while there is a live cell c that has not been processed yet do
    if c ∉ Ω {c is not contained in any oscillator} then
      {create a new oscillator O containing just c and its current state}
      O := {(c, σ_t(c))}; Ω := Ω ∪ {O}
    else { c is in an oscillator, with (c, S) ∈ O }
      S := S + σ_t(c) {update c's state list with c's current state}
    end if
    for each n ∈ N(c) {each of cell c's neighbourhood cells} do
      if σ_t(n) = ■ {n is alive}
      or σ_t(n) = □ and |N_t^■(n)| ≥ 3
          {n is dead and has three or more live cells in its neighbourhood} then
        O := O ∪ {(n, σ_0(n), ..., σ_t(n))} {add n to O}
      end if
    end for
    if any of the cells n added to O are already a member of another oscillator R then
      O := O ∪ R; Ω := Ω − {R} {combine O and R}
    end if
    continue recursively processing all neighbourhood cells n added to O
  end while
end for
```

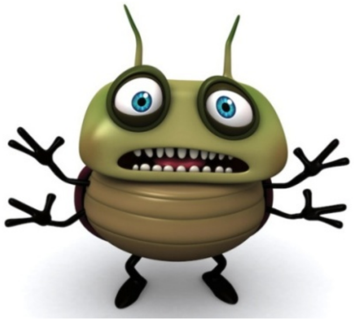
computing



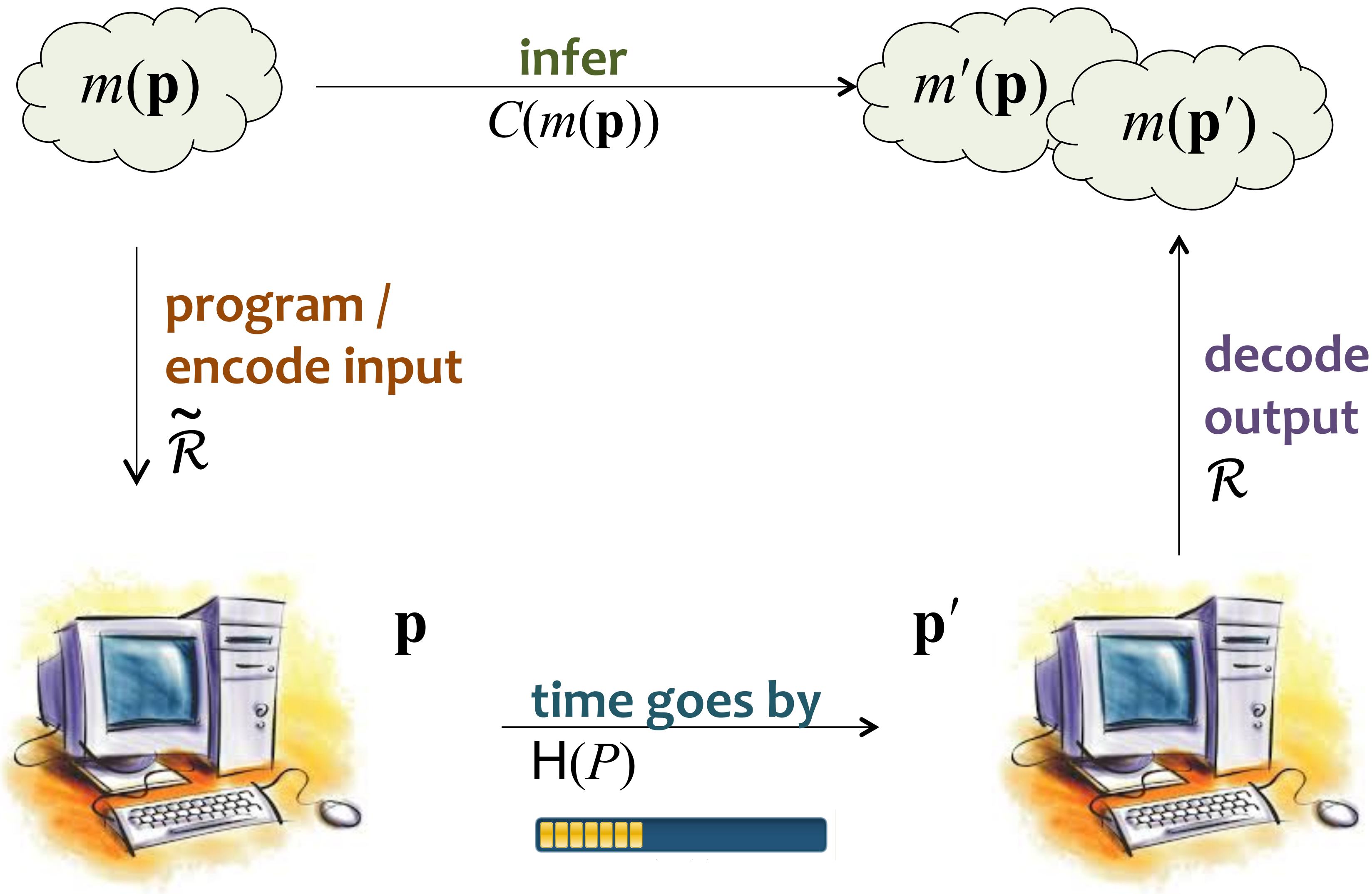
$m(\mathbf{p})$
computation
$C(m(\mathbf{p}))$
$m'(\mathbf{p})$
ε
$m(\mathbf{p}')$
program / encode input
$\tilde{\mathcal{R}}$
decode output
$\mathcal{R}$
$\mathbf{p}$
$\mathbf{p}'$
time goes by
$\mathbf{H}(\mathbf{p})$

computing

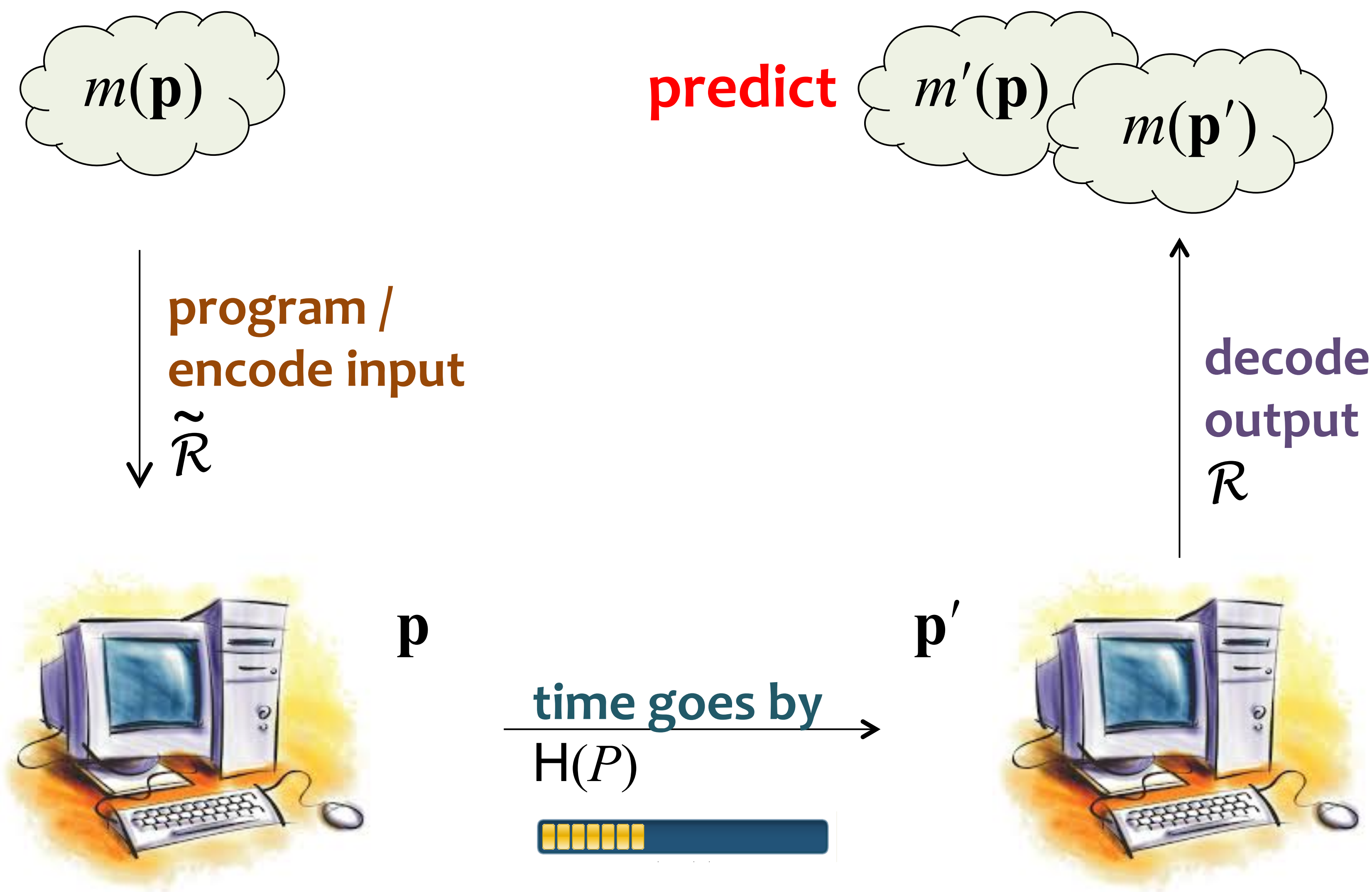
- a "good" instantiated computer **p** makes ε "small enough"
  - among other things…
- if ε is too large, **p** needs to be changed
  - the (desired) computation trumps reality
- a theory is a *model* of reality
  - models are *always* approximations
  - approximations *break down* outside the model's valid domain
- a well-instantiated computer allows use *without* needing a "computation check" every time the system is used
  - within the domain where the approximations hold



well-engineered computer / program



well-engineered computer / program



definition

computation is :

using the **physical** dynamics $\mathbf{H}$

of a ***well-engineered*** physical system

to ***predict*** an **abstract** dynamics $\boldsymbol{C}$

(subject to an encoding $\mathcal{R}$)

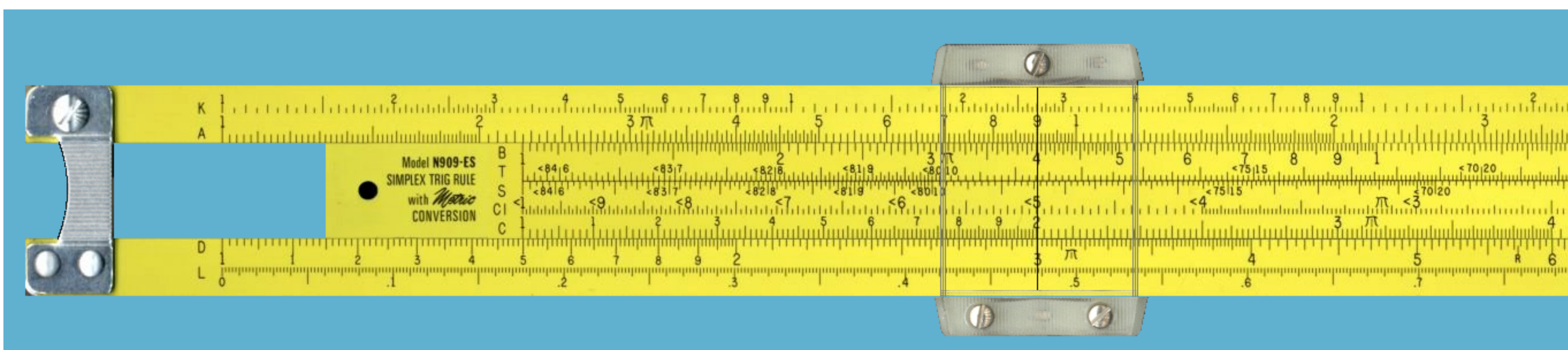
unconventional computing

- there is nothing in the definition about the nature of the physical system
  - beyond being "well-engineered"
- it doesn't have to be silicon
- it doesn't have to be a conventional computer

- we can use this definition to understand how unconventional physical systems compute

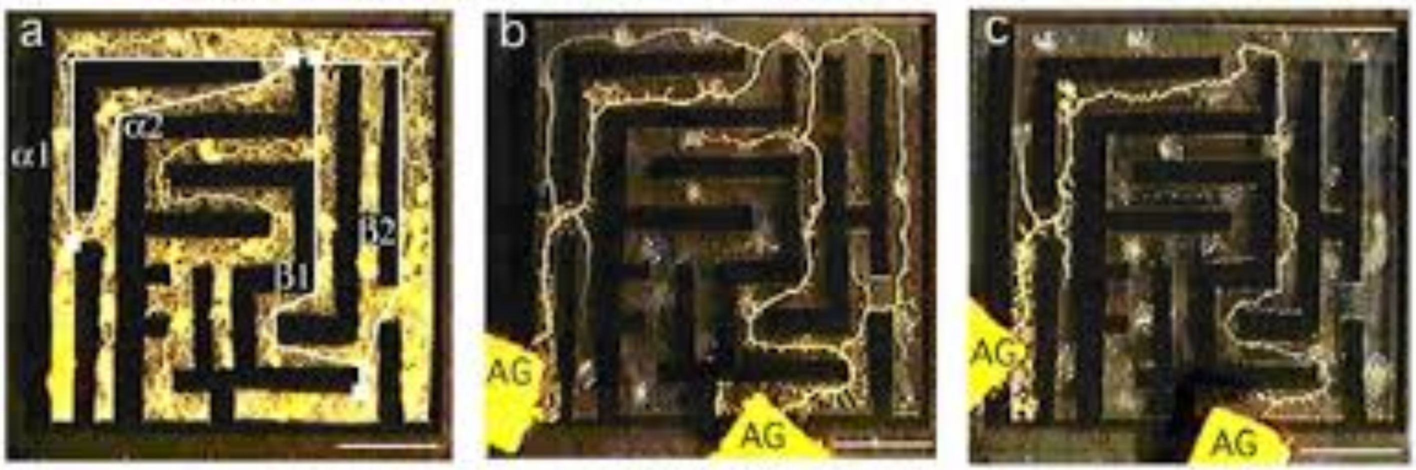
example : wooden sticks

- *calculation* : multiplication
- *theory* : how lengths of bits of wood combine
  - they add together linearly
- *instantiation* : abstract numbers instantiated as physical lengths
- *real world* : join lengths together
- *output* : read off the total length
  - logarithmic scale : so *multiplies* the values



example : slime mould

- *calculation* : solving a maze
- *theory* : how slime moulds behave in presence of food
  - they minimise distances
- *instantiation* : chopped up slime mould covers maze
  - food sources at entrance and exit
- *real world* : slime mould contracts, joining the sources
- *output* : read off path taken by slime mould



requirements for a physical computer

(i) a well-characterised substrate

$$\frac{\partial U_i^0}{\partial t}(\vec{x},t) - \frac{1}{Re}\sum_{j=1}^{3} \frac{\partial^2 U_i^0}{\partial x_j \partial x_j}(\vec{x},t) + \frac{\partial U_4^0}{\partial x_i}(\vec{x},t) + \sum_{j=1}^{3} \left(U_j^0(\vec{x},t)\frac{\partial U_i^0}{\partial x_j}(\vec{x},t) + \sum_{j_1=1}^{4} \int_{\vec{x}_1,t_1} U^1_{jj_1}(\vec{x},t;\vec{x}_1,t_1)\frac{\partial U^1_{ij_1}}{\partial x_j}(\vec{x},t;\vec{x}_1,t_1)\,d\vec{x}_1 dt_1\right) = 0$$

$$\sum_{j=1}^{3} \frac{\partial^2 U_4^0}{\partial x_j\partial x_j}(\vec{x},t) + \sum_{i,j=1}^{3}\left(\frac{\partial U_i^0}{\partial x_j}(\vec{x},t)\frac{\partial U_j^0}{\partial x_i}(\vec{x},t) + \sum_{j_1=1}^{4}\int_{\vec{x}_1,t_1}\frac{\partial U^1_{ij_1}}{\partial x_j}(\vec{x},t;\vec{x}_1,t_1)\frac{\partial U^1_{jj_1}}{\partial x_i}(\vec{x},t;\vec{x}_1,t_1)\right)=0$$

???

$$\frac{\partial U^1_{ij_1}}{\partial t}(\vec{x},t;\vec{x}_1,t_1) - \frac{1}{Re}\sum_{j=1}^{3}\frac{\partial^2 U^1_{ij_1}}{\partial x_j\partial x_j}(\vec{x},t;\vec{x}_1,t_1) + \frac{\partial U^1_{4j_1}}{\partial x_i}(\vec{x},t;\vec{x}_1,t_1) + \sum_{j=1}^{3}\left(U_j^0(\vec{x},t)\frac{\partial U^1_{ij_1}}{\partial x_j}(\vec{x},t;\vec{x}_1,t_1) + U^1_{jj_1}(\vec{x},t;\vec{x}_1,t_1)\frac{\partial U_i^0}{\partial x_j}(\vec{x},t)\right)=0$$

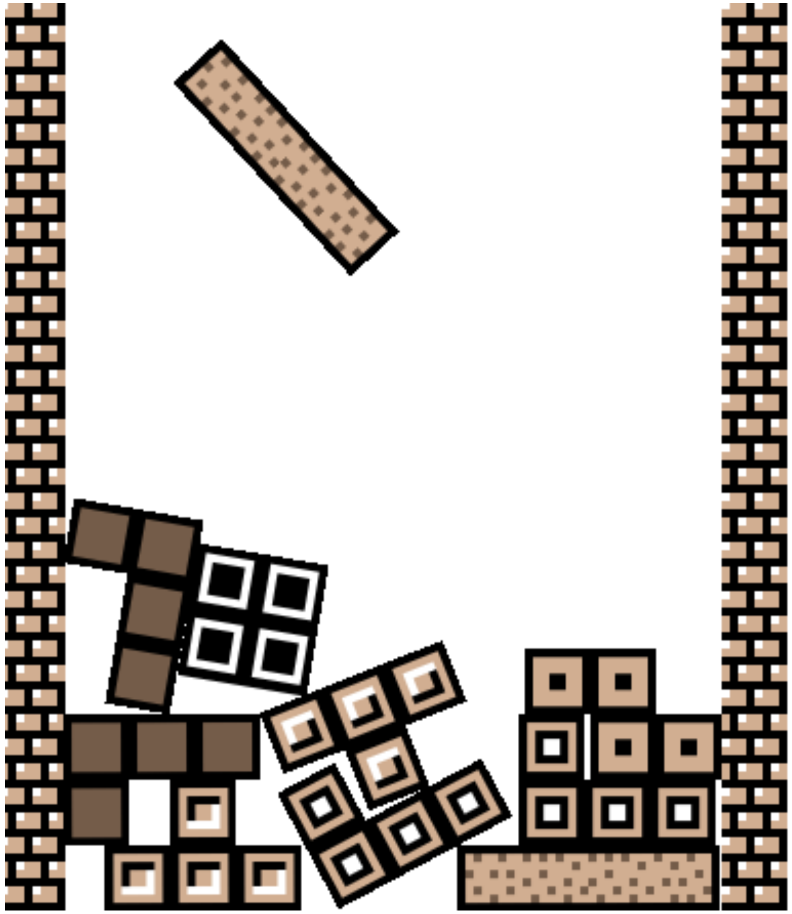
$$\sum_{j=1}^{3}\frac{\partial^2 U^1_{4j_1}}{\partial x_j\partial x_j}(\vec{x},t;\vec{x}_1,t_1) + \sum_{i,j=1}^{3} 2\frac{\partial U_i^0}{\partial x_j}(\vec{x},t)\frac{\partial U^1_{jj_1}}{\partial x_i}(\vec{x},t;\vec{x}_1,t_1) = 0$$

- including domain of applicability
  - eg, “shortest path” is a rough approximation, for small systems

substrate theories

- well-developed
  - solid state transistors
  - classical mechanics, quantum mechanics
  - reaction-diffusion chemistry
- phenomenological
  - biology
    - extrapolation and scaling issues
- naïve
  - approximate
    - shortest path
  - counterfactual
    - unbounded speeds
    - non-atomic

(ii) a well-engineered instantiation



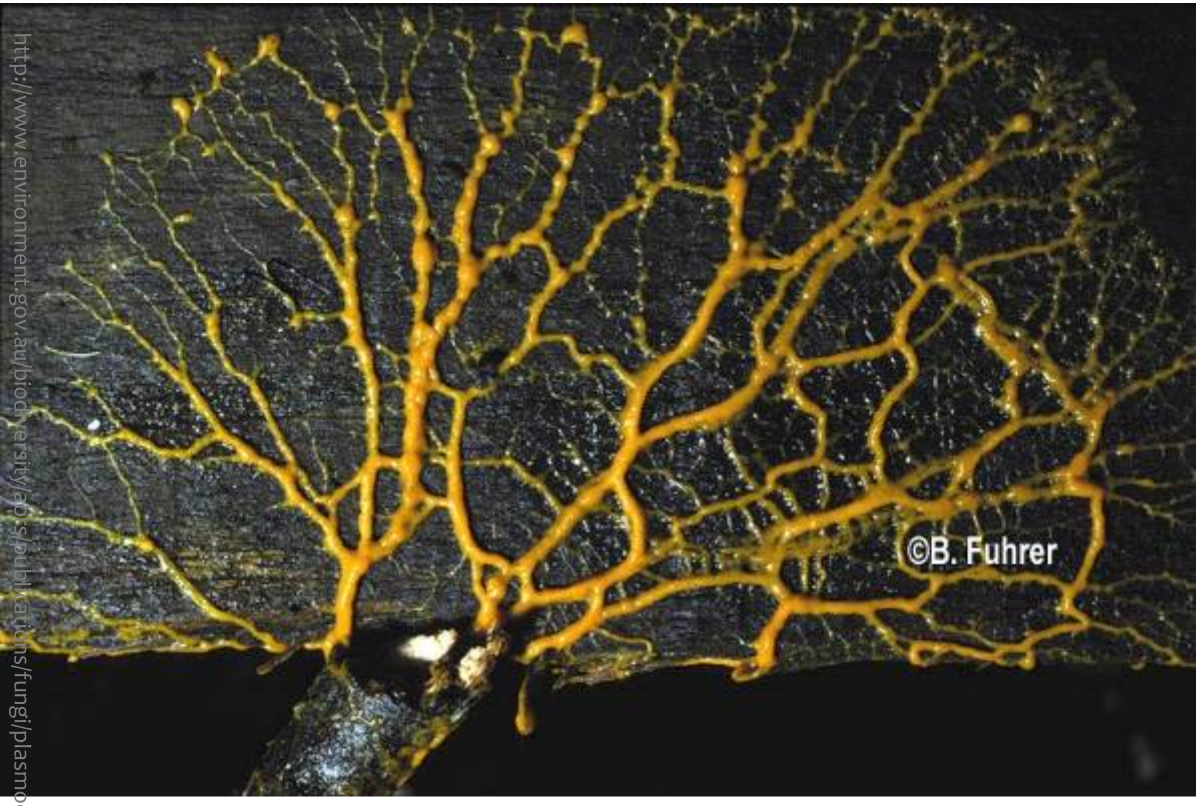
engineering issues

- theory composition
  - multiple components
  - multiple kinds of components
  - interconnections
  - control
  - ***programming***
- scaling
  - interpolation
  - extrapolation
    - model breaks down

(iii) a pre-defined encoding/decoding

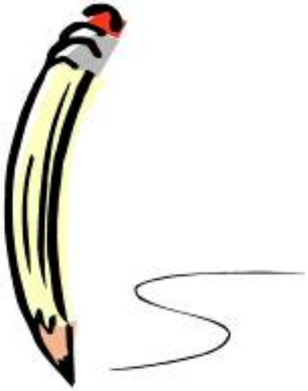
Nope

(iv) and a natural fit to the problem



http://www.environment.gov.au/biodiversity/abrs/publications/fungi/plasmodium-slime-mould.html

not here …



HM Revenue & Customs

Tax Return 2010

Tax year 6 April 2009 to 5 April 2010

natural fit

- the fit between the desired abstract dynamics and the possible physical dynamics

- small "semantic gap"
  - actually pretty poor for conventional computers!
  - "torturing" silicon to implement boolean logic

- smaller gap with other substrates, other computational models?
  - analogue computers
  - other unconventional approaches

acknowledgments

Clare Horsman, Susan Stepney, Rob C. Wagner, Viv Kendon.
When does a physical system compute?
Proceedings of the Royal Society A, **470**(2169):20140182
doi: 10.1098/rspa.2014.0182