Thermodynamics of computation.

Dominique Chu

School of Computing
University of Kent, UK
D.F.Chu@kent.ac.uk

C^3 Symposium
December 11, 2017

Outline

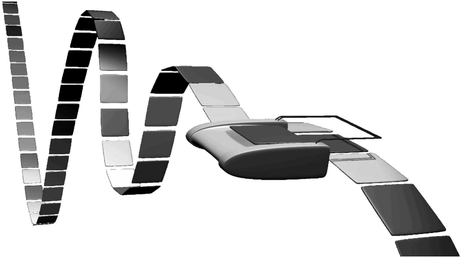
Computation

- Theoretical computer science is a mathematical theory.
- Crucially, it is not making reference to the laws of physics.
- Some of its postulates/assumptions are physically implausible.

Turing machine

A standard model in computer science is the idea of a Turing machine. It is believed that for every *computable function* there is a Turing machine that computes it.

- An input tape
- A reading head.
  - Is always in a particular state.
  - Reads a symbol from the input tape
  - Moves to the left or right depending on the internal state and the input it received.
  - Writes to the tape.
- The computation is finished when the machine enters the halting state.



Taken from: https://commons.wikimedia.org/w/index.php?curid=1505152

Physics and computation: basic insights

- It takes effort to switch a system into a particular state *at a particular time*.
- The world is fundamentally noisy. It therefore takes effort to keep the system in a particular state.

Next slide

Formal systems

Inference rules

$$
\begin{aligned}
\spadesuit\S &\mapsto \nabla \\
\nabla &\mapsto \spadesuit\sharp \\
\bigstar\square &\mapsto \lozenge\flat \\
\sharp\flat &\mapsto \S\bigstar
\end{aligned}
$$

Formal systems

Inference rules

$$\begin{array}{rcl} \spadesuit\S & \mapsto & \nabla \\ \nabla & \mapsto & \spadesuit\sharp \\ \bigstar\Box & \mapsto & \Diamond\flat \\ \sharp\flat & \mapsto & \S\bigstar \end{array}$$

Derive theorems

$$\begin{array}{l} \spadesuit\S\nabla\nabla\nabla\sharp\flat\cdots \\ \nabla\nabla\nabla\nabla\sharp\flat\cdots \\ \nabla\nabla\nabla\nabla\S\bigstar\cdots \\ \spadesuit\sharp\nabla\nabla\nabla\S\bigstar\cdots \\ \cdots \end{array}$$

A basic assumption of computer science

Assumption

- Software and hardware are separated.
- Specifically, in order to understand what a piece of software is doing, it is not necessary to understand how the hardware works.

Aristotelian causes

- Efficient cause
- Material cause
- Formal cause
- Final cause (telos)

Aristotelian causes

- Efficient cause (CPU)
- Material cause (silicon of which the CPU is made)
- Formal cause (the code)
- Final cause (the programmer who wrote the software with a purpose in mind)

Causation in electronic computers

The efficient cause of the computation is the hardware only and not connected to the software.

Robert Rosen: Efficient causation in organisms

Robert Rosen[1] claims that living systems are different from computers because they are closed with respect to efficient causation.

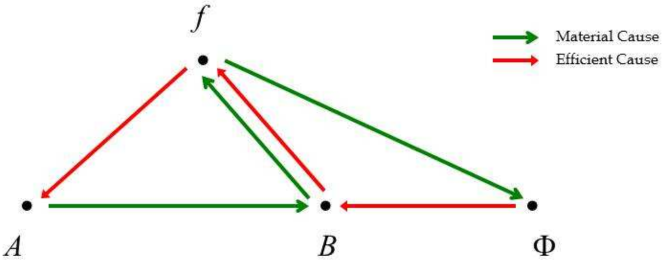
- Assume a metabolite A which is converted by some reaction f into some component of type B

$$f : A \mapsto B$$

- A and f are the material and efficient cause of B, but what causes f?
- Extend the diagram to include: some function Φ that maps B to the mappings from A to B, i.e. f.
- What is then the efficient cause of Φ?
- We can now continue with infinite regress or close the system.

[1] R. Rosen (1991). *Life Itself*. New York: Columbia University Press.

Metabolism-Repair System with Replication



Alternate notation:

$$A \xrightarrow{f} B \xrightarrow{\Phi} H(A, B) \xrightarrow{\beta} H(B, H(A, B))$$

Rosen complexity

- Rosen's central argument is that these (M,R) systems are fundamentally different from computation.
- It is not possible to retain the separation between software and hardware, while also keeping closure with respect to efficient causation.
- **Rosen complexity:** Systems that are closed wrt efficient causation "complex."

Don't trust me though on this topic!

Remarks on Chu-Ho Fall 2007

By Tim Gwinn | September 1, 2007 | Analytic vs. Synthetic, Critiques of Critiques, Rosennean Complexity, Turing Machines

Dominique Chu and Wen Kin Ho iterate their previous exercises in misunderstanding and misconstruing of Rosen's work with their latest paper, "Computational Realizations of Living Systems", in the Fall 2007 issue of the MIT journal Artificial Life [1]. The abstract: Robert Rosen's central theorem states that organisms are fundamentally different from machines, mainly because they are "closed with … Continue reading →

Taken from: `http://panmere.com/?cat=8`

Are biological systems computers?

- Rosen's point is that biological systems are fundamentally different from computers.
- There can be *simulations* of living systems, but not accurate *models*.
- Artificial life is impossible.

Example: Dynamical Hierarchies

- Assume an artificial physical world consisting of a set of components of types $\{A^0, B^0, \ldots\}$.
- Each of the components has certain rules how to interact with other components and with its environment.
- The question now is: How to design this world so that from the individual parts one gets aggregate components $\{A^1, B^1, \ldots\}$ formed of the lower level components, but with their own behaviours.
- How to get $\{A^2, B^2, \ldots\}$, etc..
- This turns out to be very difficult to do, but emerges naturally in the real world.
- One could think of simple molecules $\rightarrow$ proteins $\rightarrow$ cells $\rightarrow$ organisms

No need to bother

Higher level components can, however, be simulated.

What does it mean for a biological/biochemical system to compute?

Inference rules

$$\begin{array}{rcl} \spadesuit\S & \mapsto & \nabla \\ \nabla & \mapsto & \spadesuit\sharp \\ \bigstar\Box & \mapsto & \Diamond\flat \\ \sharp\flat & \mapsto & \S\bigstar \end{array}$$

Formally this is similar to enzymatic reactions.

$$E + A \rightharpoonup C \rightharpoonup E + B$$

Only that the reactions should be reversible.

$$E + A \rightleftharpoons C \rightleftharpoons E + B$$

Armchair chemistries

Computer simulations allow me to postulate physically implausible mechanisms.

Biochemical information processors

Examples

- Kinetic proofreading
- Chemotaxis
- Transcription/translation
- Sensing
- ...

Previous slide

A candidate notion of computation in biological systems

How can we recognise a biochemical system that computes (and distinguish it from one that does not)?

A candidate notion of computation in biological systems

How can we recognise a biochemical system that computes (and distinguish it from one that does not)?

Computation (in biochemical systems)

A biochemical system computes iff there is a Turing machine that emulates its behaviour.

A simple example

Chemical system

Assume the following chemical system:

$$A \rightharpoonup B$$

For simplicity we assume that there is only a single A at the beginning.

Turing machine

- Symbols on tape: `{A,B,_}`.
- Possible states: `{ 1|0 ,h}`.
- Initial state of the TM is `1|0`.
- Tape is
  `...,_,_,A,_,...`
- If the symbol `A` is on tape overwrite it with `B` and go into halting state.
- Otherwise step to right.

Entropy production of biochemical reactions

Assume the following system

$$A \underset{k^-}{\overset{k^+}{\rightleftharpoons}} B$$

Whenever the forward reaction happens, then the heat dissipated to the environment is given by

$$\Delta s \sim \ln\left(\frac{k^+}{k^-}\right)$$

This formula tells us:

1. Unidirectional reactions cannot exist!
2. The state $B = 1$ is not a halting state in our example above.

This means...

The above example of a chemical system is implausible.

A simple example extended

Chemical system

Assume the following chemical system:

$$A \underset{k^-}{\overset{k^+}{\rightleftharpoons}} B$$

For simplicity we assume that there is only a single *A* at the beginning.

Turing machine

- Symbols on tape: `{A,B,_}`.
- Possible states: `{ 1|0 ,0|1,h}`.
- Initial state of the TM is `1|0`.
- Tape is `...,_,_,A,_,...`
- If the symbol `A` is on tape overwrite it with `B` and go into state `0|1`, go to right.
- If the symbol `B` is on tape overwrite it with `A` and go into state `1|0`, go to right.
- If you encounter symbol _ go to left.

The computation does not halt.

About Markov chains

- A set of states.
- Transition rates between them (CTMC).
- Initial state.
- Unique steady-state (equilibrium).
- Approaching the steady-state produces entropy.
- Once in equilibrium, the system stays there.

Example

Take as an example:

$$A \underset{k^-}{\overset{k^+}{\rightleftharpoons}} B$$

- Start with 20 A and no B.
- $k^- = k^+$.
- Then the result will be, on average 10A, with some noise around this.
- Crucially, at equilibrium there will be ongoing chemical activity with reactions happening at random time points.

Computing with biochemical systems

Postulate

- The halting state of a biochemical computer is its steady-state.
- Programming the computer means to specify a CTMC.

Computing with biochemical systems

Postulate

- The halting state of a biochemical computer is its steady-state.
- Programming the computer means to specify a CTMC.

Computing procedure

1. Specify your MC and (arbitrary) initial state.
2. Let the system relax to equilibrium.
   - Entropy is produced in the process
   - The relaxation takes technically infinite amounts of time, but is characterised by a characteristic time-scale τ.
3. Read out the result.

Understanding the resource cost of computation

- Energy cost of the computation $\longrightarrow \approx$ entropy produced.
- Time required to compute $\longrightarrow \approx \tau$.
- Accuracy $\longrightarrow \approx$ noise.

van Kampen's linear noise approximation[2]

- Valid for mesoscopic chemical systems.
- Scale the volume to generate an equivalence class of systems.
- Deterministic equivalent: $V \to \infty$.
- Mean behaviour of finite systems is the same as deterministic equivalent.
- Actual systems have Gaussian noise around deterministic equivalent.
- The "noise" scales like $V^{-\frac{1}{2}}$, i.e. inverse with the volume.

[2] N. van Kampen (2007). *Stochastic Processes in Physics and Chemistry*. Third edition. Amsterdam: Elsevier.

Time to compute

- Linear noise approximation implies that system size does not affect the computation time τ.
- τ **only** depends on the rate constants of the system (which are not constrained in a relevant way).

Time to compute

- Linear noise approximation implies that system size does not affect the computation time τ.
- τ **only** depends on the rate constants of the system (which are not constrained in a relevant way).

No trade-off with time

The computation time of mesoscopic biochemical computers is fixed.

Entropy production & Noise: "Cost of the computation"

I assume now, for simplicity, that the steady-state is an equilibrium state.

- Entropy production scales linearly with the system size.
- Noise scales like $V^{-\frac{1}{2}}$.

Entropy production & Noise: “Cost of the computation”

I assume now, for simplicity, that the steady-state is an equilibrium state.

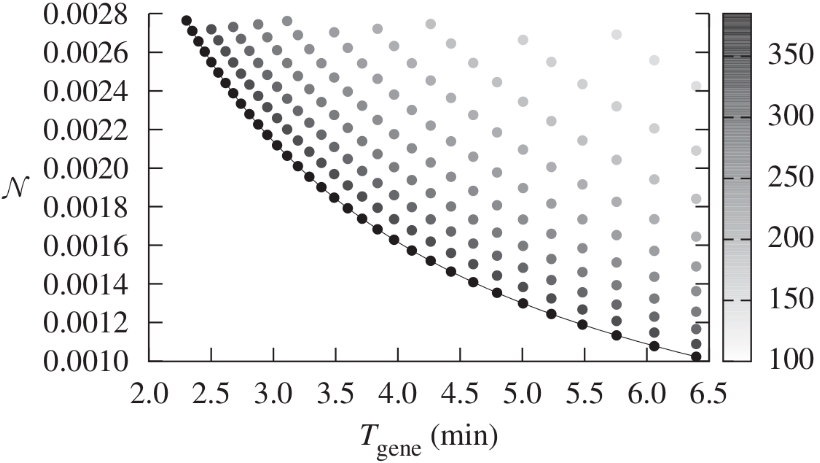
- Entropy production scales linearly with the system size.
- Noise scales like $V^{-\frac{1}{2}}$.

Accuracy-cost trade-off

For analogue computations in biochemical systems the cost and accuracy of the computation are traded off against one another.

Previous findings

There is a general trade-off in biochemical computers between the accuracy, the time taken for the computation and the energy usage/energy cost/dissipation rate.[3]



[3] N. Zabet and D. Chu (2010). "Computational limits to binary genes." eng. In: *Journal of the Royal Society Interface* 7.47, pp. 945–954.

Measurement process

How can we know the outcome of the computation?

Measurement process

How can we know the outcome of the computation?

- We need to estimate the moments of the probability distribution by repeated sampling of the system.
- Each sampling event comes at a cost (which depends on the speed of the measurement).
- The sampling frequency is limited by the relaxation period of the system.
- The accuracy of the measurement is limited by the number of samples taken.

Measurement process

How can we know the outcome of the computation?

- We need to estimate the moments of the probability distribution by repeated sampling of the system.
- Each sampling event comes at a cost (which depends on the speed of the measurement).
- The sampling frequency is limited by the relaxation period of the system.
- The accuracy of the measurement is limited by the number of samples taken.

Time-trade-offs

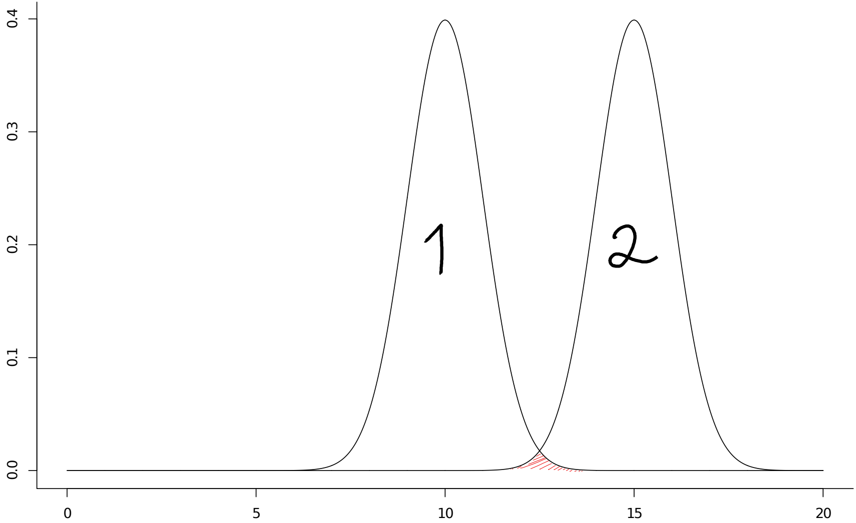
There is thus a trade-off between the time required to read the result and the accuracy of the result.

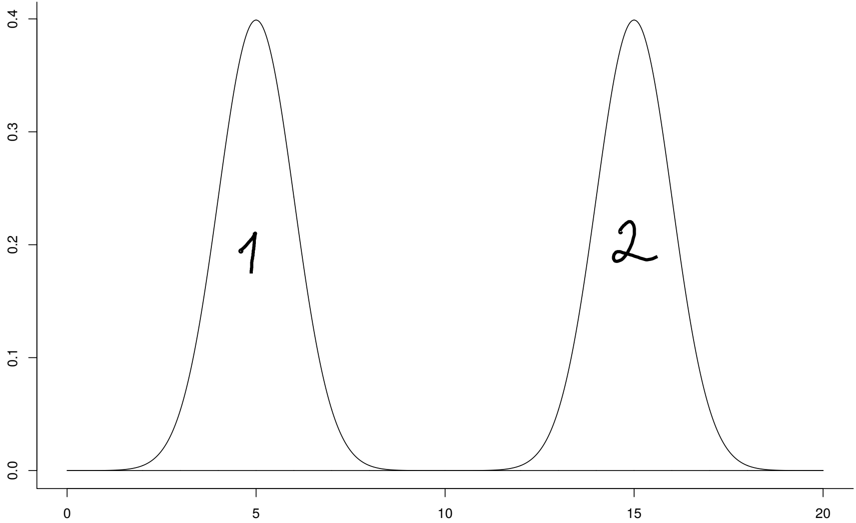
Summary: Analogue computation by mesoscopic biochemical computers

- Computation proper only leads to a trade-off between accuracy and cost.
- Time trade-offs are a result of the need to measure the outcome.

Improving the scaling: Digital computation

- Assume that our system can only be in one of two possible states.
- All we need to do then, is to distinguish between the two possible states.
- No more need to determine all moments.


0.4
0.3
0.2
0.1
0.0
0
5
10
15
20
1
2


0.0
0.1
0.2
0.3
0.4
0
5
10
15
20
1
2

Estimating the state

- Assume that the states are indicated by the value of some Gaussian distributed random variable X.
- In state 1, the variable has a mean of μ_1 and in state 2 it has a mean of μ_2.
- Now draw a sample to obtain a value x.
- The likelihood that the sample belongs to state 1 is given by

$$p_{\mu_1}(x)(1 - p_{\mu_1}(x)) \tag{1}$$

  Equivalently for state 2.
- The ratio of the likelihoods $\sim \exp(-|\mu_1 - \mu_2|)$.

Estimating the state

- Assume that the states are indicated by the value of some Gaussian distributed random variable X.
- In state 1, the variable has a mean of μ_1 and in state 2 it has a mean of μ_2.
- Now draw a sample to obtain a value x.
- The likelihood that the sample belongs to state 1 is given by

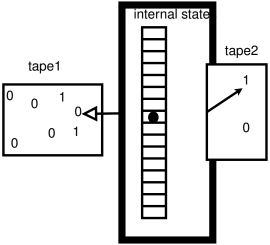
$$p_{\mu_1}(x)(1 - p_{\mu_1}(x)) \tag{1}$$

  Equivalently for state 2.
- The ratio of the likelihoods $\sim \exp(-|\mu_1 - \mu_2|)$.

Deterministic computation

It is therefore possible to determine the state of a stochastic system accurately by drawing a few samples only.

A machine that can reliably decode repetition codes.

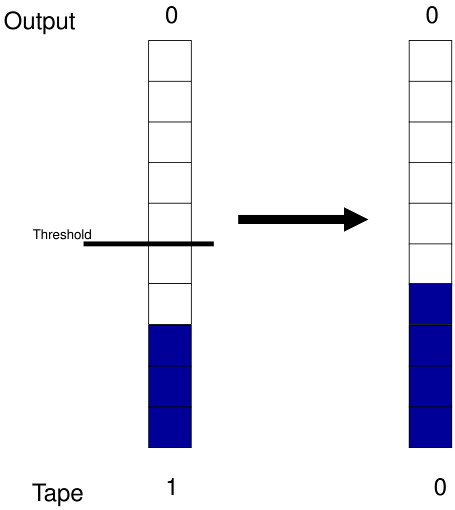


- Accepts input tape with symbols 0 and 1.
- Determines the majority of symbols with probability π.
- Machine can be tuned to be arbitrarily accurate.
- Finite time.
- Finite energy required to run the machine.

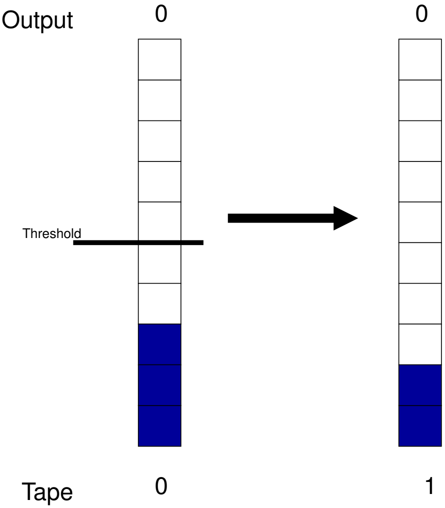
The machine is heavily inspired by Barato and Seifert[4].

[4] Andre C. Barato and Udo Seifert (2014). "Stochastic thermodynamics with information reservoirs". In: *Physical Review E* 90.4.

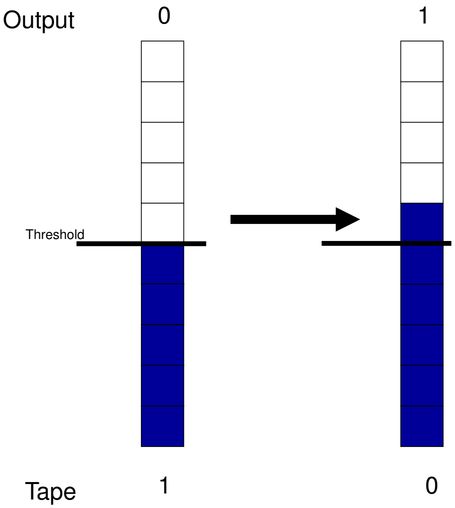
Machine reading a 1 from tape



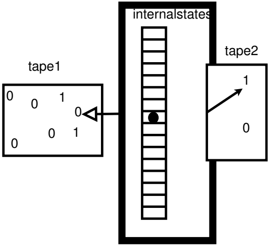
Machine reading a 0 from tape



Machine transitioning to output state 1



A machine that can reliably recognise input



- Entropy production

$$\Delta S = -\ln \pi_0 \sim K,$$

where K is the number of internal states.

- Accuracy (in the limit of infinite tapes)

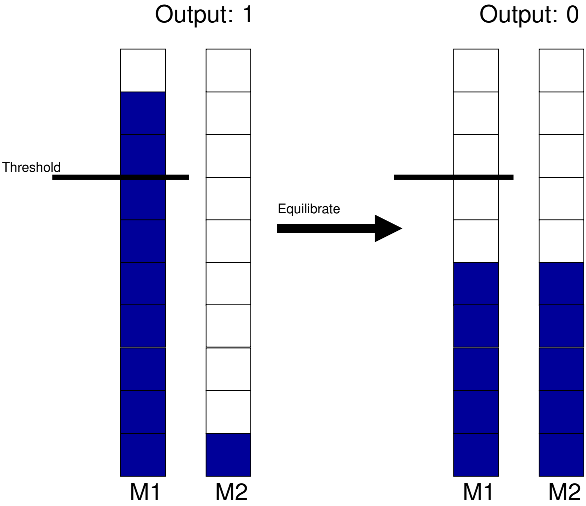
$$P(\text{error}) = \left(\frac{\epsilon}{1-\epsilon}\right)^{-\frac{K}{2}},$$

where ϵ is the proportion of majority symbols on tape.

Logic gates

- NOT gate trivial. Just relabel output.
- AND/OR gate:
  - Use two machines independently.
  - First machine now has a higher threshold.
  - Disconnect both machines from input.
  - Let the internal states equilibrate.
  - Check whether the first machine has crossed the threshold.

AND gate



Writing output

- As a last step we also need to write the result of the computation to output.
- This restores the machine (logically).
- This can be achieved by running the reading machine in reverse.

Binary Computation

- Strict determinism is thermodynamically implausible.
- Analogue computation in cells has unfavourable scaling relationship, forcing a trade-off between accuracy and energy usage.
- Digital computation has a favourable scaling, allowing quasi-deterministic computation at finite (and small) cost.

Concluding question

Biological computers

- Why do biological computers not universally take advantage of this benign scaling and compute deterministically?
- Instead, they seem to waste energy on poor computations. Or don't they?