

C Thermodynamics of computation

This lecture is based on Charles H. Bennett's "The Thermodynamics of Computation — a Review," *Int. J. Theo. Phys.*, 1982 [B82]. Unattributed quotations are from this paper.

"Computers may be thought of as engines for transforming free energy into waste heat and mathematical work." [B82]

C.1 Brownian computers

- ¶1. Rather than trying to avoid randomization of kinetic energy (transfer from mechanical modes to thermal modes), perhaps it can be exploited. An example of *respecting the medium* in *embodied computation*.
- ¶2. **Brownian computer:** Makes logical transitions as a result of thermal agitation.
It is about as likely to go backward as forward.
It may be biased in the forward direction by a very weak external driving force.
- ¶3. **DNA:** DNA transcription is an example. It runs at about 30 nucleotides per second and dissipates about $20kT$ per nucleotide, making less than one mistake per 10^5 nucleotides.
- ¶4. **Chemical Turing machine:** Tape is a large macromolecule analogous to RNA.
An added group encodes the state and head location.
For each transition rule there is a hypothetical enzyme that catalyzes the state transition.
- ¶5. Drift velocity is linear in dissipation per step.
- ¶6. We will look molecular computation in much more detail later in the class.
- ¶7. **Clockwork Brownian TM:** He considers a "clockwork Brownian TM" comparable to the billiard ball computers. It is a little more realistic, since it does not have to be machined perfectly and tolerates environmental noise.
- ¶8. Drift velocity is linear in dissipation per step.

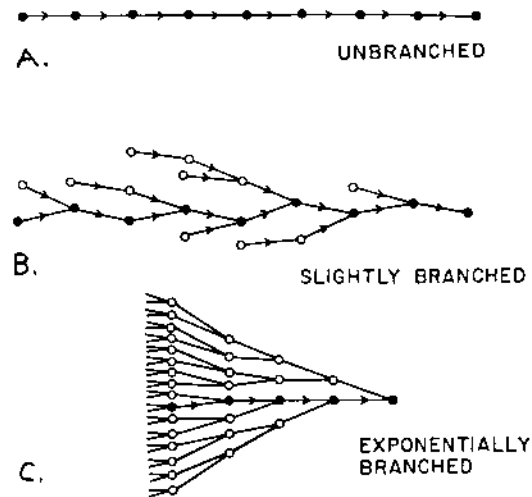


Figure II.23: Different degrees of logical reversibility. [from B82]

C.2 Reversible computing

C.2.a REVERSIBLE TM

- ¶1. Bennett (1973) defined a reversible TM.

C.2.b REVERSIBLE LOGIC

- ¶1. As in ballistic computing, Brownian computing needs logical reversibility.
- ¶2. A small degree of irreversibility can be tolerated (see Fig. II.23).
 - (A) Strictly reversible computation. Any driving force will ensure forward movement.
 - (B) Modestly irreversible computation. There are more backward detours, but can still be biased forward.
 - (C) Exponentially backward branching trees. May spend much more of its time going backward than going forward, especially since one backward step is likely to lead to more backward steps.
- ¶3. For forward computation on such a tree the dissipation per step must exceed kT times the log of the mean number of immediate predecessors

to each state.

- ¶4. These undesired states may outnumber desired states by factors of 2^{100} , requiring driving forces on the order of $100kT$.
Why 2^{100} ? Think of the number of possible predecessors to a state that does something like $x := 0$. Or the number of ways of getting to the next statement after a loop.

C.3 Erasure

- ¶1. As it turns out, it is not measurement or copying that necessarily dissipates energy, it is the erasure of previous information to restore it to a standard state so that it can be used again.
Thus, in a TM writing a symbol on previously used tape requires two steps: erasing the previous contents and then writing the new contents. It is the former process that increases entropy and dissipates energy. Cf. the mechanical TM we saw in CS312; it erased the old symbol before it wrote the new one.
Cf. also old computers on which the store instruction was called “clear and add.”
- ¶2. A bit can be copied reversibly, with arbitrarily small dissipation, if it is initially in a standard state (Fig. II.24).
The *reference bit* (fixed at 0, the same as the moveable bit’s initial state) allows the initial state to be restored, thus ensuring reversibility.
- ¶3. Susceptible to degradation by thermal fluctuation and tunneling. However, the error probability η and dissipation ϵ/kT can be kept arbitrarily small and much less than unity.
 ϵ here is the driving force.
- ¶4. Whether erasure must dissipate energy depends on the prior state (Fig. II.25).
(B) The initial state is genuinely unknown. This is reversible (step 6 to 1). $kT \ln 2$ work was done to compress it into the left potential well (decreasing the entropy). Reversing the operation increases the entropy by $kT \ln 2$.
That is, the copying is reversible.

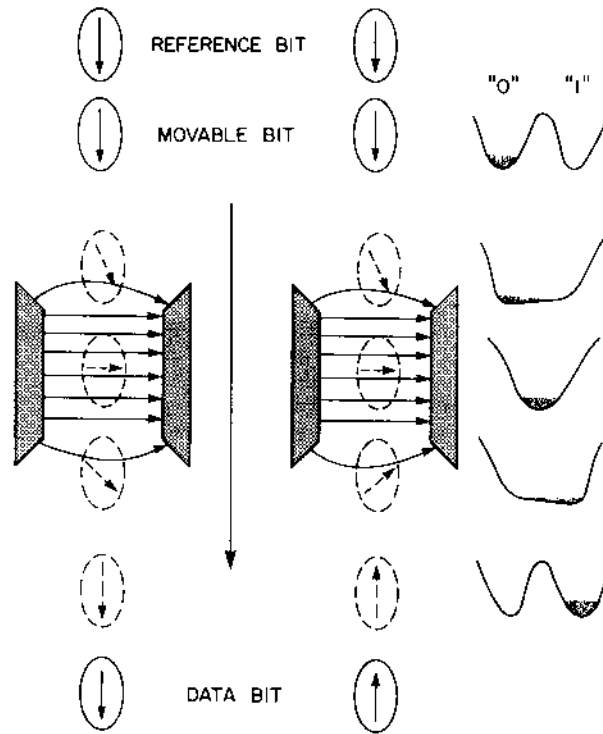


Fig. 14. Reversible copying using a one-domain ferromagnet. The movable bit, initially zero, is mapped into the same state as the data bit (zero in left column; one in center column). Right column shows how the probability density of the movable bit's magnetization, initially concentrated in the "0" minimum, is deformed continuously until it occupies the "1" minimum, in agreement with a "1" data bit.

Figure II.24: [from B82]

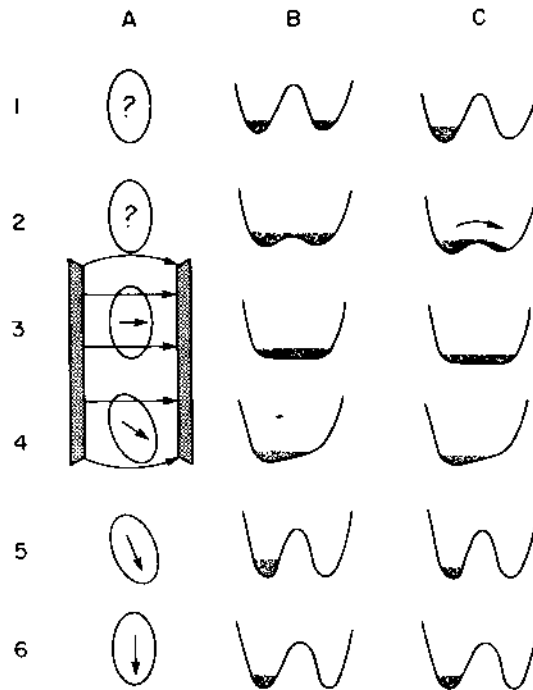


Fig. 16. Erasure of a bistable one-domain ferromagnet. Column A: the bistable element, which may be magnetized either up or down (1), is moved gradually (2) into a transverse field that abolishes its bistability symmetrically (3). It is then rotated slightly counterclockwise (4), to bias the soft mode downward, removed from the field (5), and rotated back again, leaving it in the down or zero state (6). Column B: Evolution of the probability density when the manipulation just described is used to erase a random, unknown bit. Column C: Behavior of the probability density when the manipulation is applied to a known bit (here zero). An irreversible entropy increase of $k \ln 2$ occurs at stage 2, when the probability density leaks out of the initially occupied minimum.

Figure II.25: [from B82]

(C) The initial state is known (perhaps as a result of a prior computation or measurement). This initial state is lost (forgotten), converting $kT \ln 2$ of work into heat with no compensating decrease of entropy. The irreversible entropy increase happens because of the expansion at step 2. This is where we're forgetting the initial state (vs. case B, where there was nothing known to be forgotten).

- ¶5. In research reported in March 2012 [EVLP] a setup very much like this was used to confirm experimentally the Landauer Principle and that it is the erasure that dissipates energy.

“incomplete erasure leads to less dissipated heat. For a success rate of r , the Landauer bound can be generalized to”

$$\langle Q \rangle_{\text{Landauer}}^r = kT[\ln 2 + r \ln r + (1 - r) \ln(1 - r)].$$

“Thus, no heat needs to be produced for $r = 0.5$.” [EVLP]

- ¶6. We have seen that erasing a random register increases entropy in the environment in order to decrease it in the register: $NkT \ln 2$ for an N -bit register.
- ¶7. Such an initialized register can be used as “fuel” as it thermally randomizes itself to do $NkT \ln 2$ useful work. Keep in mind that these 0s (or whatever) are physical states (charges, compressed gas, magnetic spins, etc.).
- ¶8. Any computable sequence of N bits (e.g., the first N bits of π) can be used as fuel by a slightly more complicated apparatus. Start with a reversible computation from N 0s to the bit string. Reverse this computation, which will transform the bit string into all 0s, which can be used as fuel. This “uncomputer” puts it in usable form.
- ¶9. Suppose we have a random bit string, initialized, say, by coin tossing. Because it has a specific value, we can in principle get $NkT \ln 2$ useful work out of it, but we don't know the mechanism (the “uncomputer”) to do it. The apparatus would be specific to each particular string.

C.4 Algorithmic entropy

- ¶1. *Algorithmic information theory* was developed by Ray Solomonoff c. 1960, Andrey Kolmogorov in 1965, and Gregory Chaitin, around 1966.
- ¶2. **Algorithmic entropy:** The algorithmic entropy $H(x)$ of a microstate x is the number of bits required to describe x as the output of a universal computer, roughly, the size of the program required to compute it.
Specifically the smallest “self-delimiting” program (i.e., its end can be determined from the bit string itself).
- ¶3. Differences in machine models lead to differences of $\mathcal{O}(1)$, and so $H(x)$ is well-defined up to an additive constant (like thermodynamical entropy).
- ¶4. Note that H is not computable.
- ¶5. **Algorithmically random:** A string is *algorithmically random* if it cannot be described by a program very much shorter than itself.
- ¶6. For any N , most N -bit strings are algorithmically random.
(For example, “there are only enough $N - 10$ bit programs to describe at most $1/1024$ of all the N -bit strings.”)
- ¶7. Deterministic transformations cannot increase algorithmic entropy very much.
Roughly, $H[f(x)] \approx H(x) + |f|$, where $|f|$ is the size of program f .
Reversible transformations also leave H unchanged.
- ¶8. A transformation must be probabilistic to be able to increase H significantly.
- ¶9. **Statistical entropy:** Statistical entropy in units of bits is defined:

$$S_2(\mathbf{p}) = - \sum_x p(x) \lg p(x).$$

- ¶10. Statistical entropy (a function of the macrostate) is related to algorithmic entropy (an average over algorithmic entropies of microstates) as follow:

$$S_2(\mathbf{p}) < \sum_x p(x) H(x) \leq S_2(\mathbf{p}) + H(\mathbf{p}) + \mathcal{O}(1).$$

- ¶11. A macrostate \mathbf{p} is *concisely describable* if, for example, “it is determined by equations of motion and boundary conditions describable in a small number of bits.”
In this case S_2 and H are closely related as given above.
- ¶12. For macroscopic systems, typically $S_2(\mathbf{p}) = \mathcal{O}(10^{23})$ while $H(\mathbf{p}) = \mathcal{O}(10^3)$.
- ¶13. If the physical system increases its H by N bits, which it can do only by acting probabilistically, it can “convert about $NkT \ln 2$ of waste heat into useful work.”
- ¶14. “[T]he conversion of about $NkT \ln 2$ of work into heat in the surroundings is necessary to decrease a system’s algorithmic entropy by N bits.”

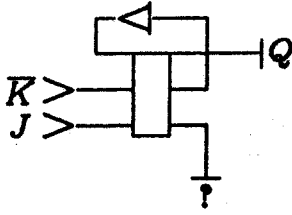
D Sources

- B82** Bennett, C. H. The Thermodynamics of Computation — a Review. *Int. J. Theo. Phys.*, 21, 12 (1982), 905–940.
- EVLP** Berut, Antoine, Arakelyan, Artak, Petrosyan, Artyom, Ciliberto, Sergio, Dillenschneider, Raoul and Lutz, Eric. Experimental verification of Landauer’s principle linking information and thermodynamics. *Nature* 483, 187–189 (08 March 2012). doi:10.1038/nature10872
- F** Frank, Michael P. Introduction to Reversible Computing: Motivation, Progress, and Challenges. CF ‘05, May 4–6, 2005, Ischia, Italy.
- FT82** Fredkin, E. F., Toffoli, T. Conservative logic. *Int. J. Theo. Phys.*, 21, 3/4 (1982), 219–253.

E Exercises

Exercise II.1 Show that the Fredkin gate is reversible.

Exercise II.2 Show that the Fredkin gate implementations of the NOT, OR, and FAN-OUT gates (Fig. II.12) are correct.

Figure II.26: Implementation of J- \bar{K} flip-flop. [from FT82]

Exercise II.3 Use the Fredkin gate to implement XOR.

Exercise II.4 Show for the eight possible inputs that Fig. II.13 is a correct implementation of a 1-line to 4-line demultiplexer. That is, show in each of the four cases $A_1A_0 = 00, 01, 10, 11$ the bit $X = 0$ or 1 gets routed to Y_0, Y_1, Y_2, Y_3 , respectively. You can use an algebraic proof, if you prefer.

Exercise II.5 Show that implementation of a J- \bar{K} flip-flop with Fredkin gates in Fig. II.26 is correct. A J- \bar{K} flip-flop has the following behavior:

J	\bar{K}	behavior
0	0	reset, $Q \rightarrow 0$
0	1	hold, Q doesn't change
1	0	toggle, $Q \rightarrow \bar{Q}$
1	1	set, $Q \rightarrow 1$

Exercise II.6 Show that the inverse of the interaction gate works correctly.

Exercise II.7 Show that the realization of the Fredkin gate in terms of interaction gates (Fig. II.22) is correct, by labeling the inputs and outputs of the interaction gates with Boolean expressions of a , b , and c .