B Conservative logic

These lectures are based primarily on Edward Fredkin and Tommaso Toffoli's "Conservative logic," *Int. J. Theo. Phys.*, 1982 [FT82].

B.1 Mechanical and thermal modes

¶1. Systems can be classified by their size and completeness of specification:

specification:	complete	incomplete	
size:	~ 1	~ 100	$\sim 10^{23}$
laws:	dynamical	statistical	thermodynamical
reversible:	yes	no	no

¶2. **Dynamical system:** Some systems with a relatively small number of particles or degrees of freedom can be completely specified.
E.g., 6 DoF for each particle (x, y, z, p_x, p_y, p_z).

¶3. That is, we can prepare an *individual* system in an initial state and expect that it will behave according to the dynamical laws that describe it.

¶4. Think of billiard balls or pucks on a frictionless surface, or electrons moving through an electric or magnetic field.

¶5. So far as we know, the laws of physics at this level (classical or quantum) are reversible.

¶6. **Statistical system:** If there are a large number of particles with many degrees of freedom (several orders of magnitude), then it is impractical to specify the system completely.

¶7. Small errors in the initial state will have a larger effect, due to complex interaction of the particles.

¶8. Therefore we must resort to *statistical laws.*

¶9. They don't tell us how an individual system will behave (there are too many sources of variability), but they tell us how *ensembles* of similar systems (or preparations) behave.

¶10. We can talk about the average behavior of such systems, but we also have to consider the variance, because unlikely outcomes are not impossible.
For example, tossing 10 coins has a probability of 1/1024 of turning up all heads.

¶11. Statistical laws are in general irreversible (because there are many ways to get to the same state).

¶12. **Thermodynamical system:** Macroscopic systems have a very large number of particles ($\sim 10^{23}$) and a correspondingly large number of DoF.

¶13. Obviously such systems cannot be completely specified (we cannot describe the initial state and trajectory of every atom).

¶14. We can derive statistical laws, but in these cases most macrostates become so improbable that they are virtually impossible: Example: the cream unmixing from your coffee.

In the thermodynamic limit, the *likely* is *inevitable*, and the *unlikely* is *impossible*.

¶15. In these cases, *thermodynamical laws* describe the virtually deterministic (but *irreversible*) dynamics of the system.

¶16. **Mechanical vs. thermal modes:** Sometimes in a macroscopic system we can separate a small number of *mechanical modes* (DoF) from the *thermal modes*.

"Mechanical" includes "electric, magnetic, chemical, etc. degrees of freedom."

¶17. The mechanical modes are strongly coupled to each other but weakly coupled to the thermal modes.
(e.g., bullet, billiard ball)

¶18. Thus the mechanical modes can be treated exactly or approximately independently of the thermal modes.

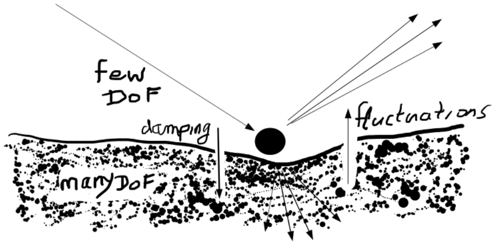

Figure II.5: Complementary relation of damping and fluctuations.

¶19. **Conservative mechanisms:** In the ideal case the mechanical modes are completely decoupled from the thermal modes, and so the mechanical modes can be treated as an isolated (and reversible) dynamical system.

¶20. The energy of the mechanical modes (once initialized) is independent of the energy ($\sim kT$) of the thermal modes.

¶21. The mechanical modes are *conservative*; they don't dissipate any energy.

¶22. This is the approach of reversible computing.

¶23. **Damped mechanisms:** Suppose we want irreversible mechanical modes, e.g., for implementing irreversible logic.

¶24. The physics is reversible, but the information lost by the mechanical modes cannot simply disappear; it must be transferred to the thermal modes. This is *damping.*

¶25. But the transfer is reversible, so *noise* will flow from the thermal modes back to the mechanical modes, making the system nondeterministic.

¶26. "If we know where the *damping* comes from, it turns out that that is also the source of the *fluctuations*" (Feynman, 1963).

Think of a bullet ricocheting off a flexible wall filled with sand. It dissipates energy into the sand and also acquires noise in its trajectory (see Fig. II.5.

¶27. To avoid nondeterminacy, the information may be encoded redundantly so that the noise can be filtered out.

I.e., signal is encoded in multiple mechanical modes, on which we take a majority vote or an average.

¶28. The signal can be encoded with energy much greater than any one of the thermal modes, $E \gg kT$, to bias the energy flow from mechanical to thermal preferring dissipation to noise).

¶29. **Signal regeneration:** Free energy must refresh the mechanical modes and heat must be flushed from the thermal modes.

¶30. "[I]mperfect knowledge of the dynamical laws leads to uncertainties in the behavior of a system comparable to those arising from imperfect knowledge of its initial conditions... Thus, the same regenerative processes which help overcome thermal noise also permit reliable operation in spite of substantial fabrication tolerances."

¶31. Damped mechanisms have proved to be very successful, but they are inherently inefficient.

¶32. "In a damped circuit, the rate of heat generation is proportional to the number of computing elements, and thus approximately to the useful volume; on the other hand, the rate of heat removal is only proportional to the free *surface* of the circuit. As a consequence, computing circuits using damped mechanisms can grow arbitrarily large in two dimensions only, thus precluding the much tighter packing that would be possible in three dimensions."

B.2 Physical assumptions of computing

B.2.a Dissipative logic

¶1. The following physical principles are implicit in the existing theory of computation.

¶2. **P1. The speed of propagation of information is bounded:** No action at a distance.

¶3. **P2. The amount of information which can be encoded in the state of a finite system is bounded:** This is a consequence of thermodynamics and quantum theory.

¶4. **P3. It is possible to construct macroscopic, dissipative physical devices which perform in a recognizable and reliable way the logical functions AND, NOT, and FAN-OUT:** This is an empirical fact.

B.2.b Conservative logic

¶1. "Computation is based on the storage, transmission, and processing of discrete signals."

¶2. Only macroscopic systems are irreversible, so as we go to the microscopic level, we need to understand reversible logic. This leads to new physical principles.

¶3. **P4. Identity of transmission and storage:** In a relativistic sense, they are identical.

¶4. **P5. Reversibility:** Because microscopic physics is reversible.

¶5. **P6. One-to-one composition:** Physically, fan-out is not trivial, so we cannot assume that one function output can be substituted for any number of input variables.
We have to treat fan-out as a specific signal-processing element.

¶6. **P7. Conservation of additive quantities:** It can be shown that in a reversible systems there are a number of independent conserved quantities.

¶7. In many systems they are *additive* over the subsystems.

¶8. **P8. The topology of space-time is locally Euclidean:** "Intuitively, the amount of 'room' available as one moves away from a certain point in space increases as a power (rather than as an exponential) of the distance from that point, thus severely limiting the connectivity of a circuit."

¶9. What are sufficient primitives for conservative computation?
The *unit wire* and the *Fredkin gate*.

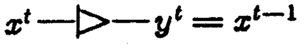

Figure II.6: Symbol for unit wire. [FT82]

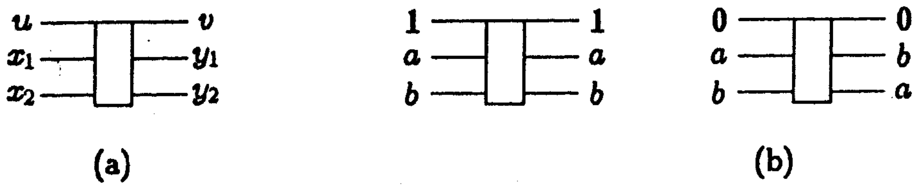

Figure II.7: "(a) Symbol and (b) operation of the Fredkin gate." [FT82]

B.3 Unit wire

¶1. Information storage in one reference frame may be information transmission in another.
E.g., leaving a note on a table in an airplane (at rest with respect to earth or not, or to sun, etc.).

¶2. The *unit wire* moves one bit of information from one space-time point to another space-time point separated by one unit of time. See Fig. II.6.

¶3. **State:** "The value that is present at a wire's input at time t (and at its output at time $t+1$) is called the *state* of the wire *at time t*."

¶4. It is invertible and conservative (since it conserves the number of 0s and 1s in its input).
(Note that there are mathematically reversible functions that are not conservative, e.g., NOT.)

B.4 Fredkin gate

¶1. **Conservative logic gate:** Any Boolean function that is invertible and conservative.

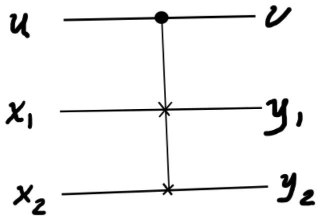

Figure II.8: Alternative notation for Fredkin gate.

¶2. **Conditional rerouting:** Since the number of 1s and 0s is conserved, conservative computing is essentially *conditional rerouting*

¶3. **Rearranging vs. rewriting:** Conventional models of computation are based on *rewriting*
(e.g., TMs, lambda calculus, register machines, term rewriting systems, Post and Markov productions).
But we have seen that overwriting dissipates energy (and is non-conservative).

¶4. In conservative logic we *rearrange* bits without creating or destroying them.
(No infinite "bit supply" and no "bit bucket.")

¶5. **Fredkin gate:** The *Fredkin gate* is a conditional swap operation:

$$\begin{aligned} (1, a, b) &\mapsto (1, a, b), \\ (0, a, b) &\mapsto (0, b, a). \end{aligned}$$

The first input is a *control* signal and the other two are *data* signals.
Here, 0 signals a swap, but some authors use 1 to signal a swap.
See Fig. II.7 and Fig. II.10. Fig. II.8 shows an alternative notation for the Fredkin gate.

¶6. Note that it is reversible and conservative.

¶7. **Universal:** The Fredkin gate is a universal Boolean primitive for conservative logic.

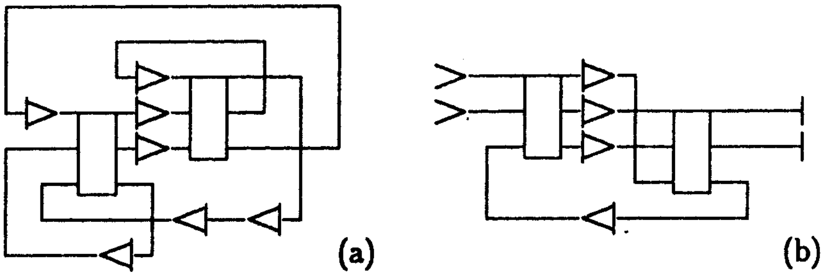

Figure II.9: “(a) closed and (b) open conservative-logic circuits.” [FT82]

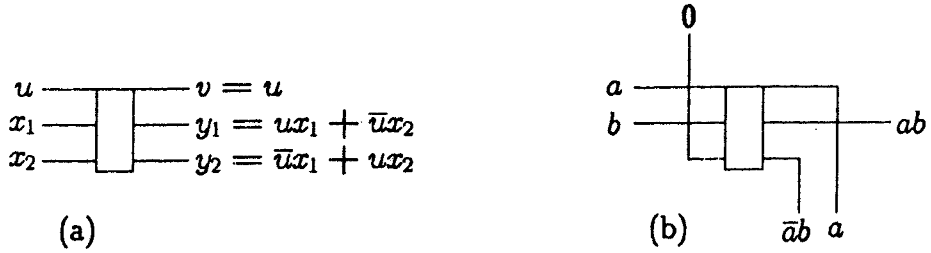

Figure II.10: “Behavior of the Fredkin gate (a) with unconstrained inputs, and (b) with x_2 constrained to the value 0, thus realizing the AND function.” [FT82]

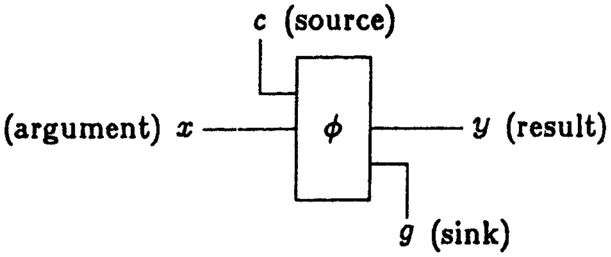

Figure II.11: "Realization of f by ϕ using source and sink. The function $\phi : (c, x) \mapsto (y, g)$ is chosen so that, for a particular value of $c, y = f(x)$." [FT82]

B.5 Conservative logic circuits

¶1. "A conservative-logic circuit is a directed graph whose nodes are conservative-logic gates and whose arcs are wires of any length [Fig. II.9]."

¶2. We can think of the gate as instantaneous and the unit wire as being a unit delay, of which we can make a sequence (or imagine intervening identity gates).

¶3. **Closed vs. open:** A *closed circuit* is a *closed* (or *isolated*) physical system.
An *open circuit* has external inputs and outputs.

¶4. The number of outputs must equal the number of inputs.

¶5. It may be part of a larger conservative circuit, or connected to the environment.

¶6. **Discrete-time dynamical system:** A conservative-logic circuit is a *discrete-time dynamical system.*

¶7. **Degrees of freedom:** The number N of unit wires in the circuit is its number of DoF.
The numbers of 0s and 1s at any time is conserved, $N = N_0 + N_1$.

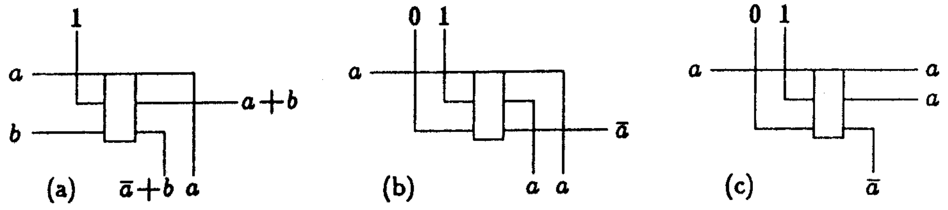

Figure II.12: "Realization of the (a) OR, (b) NOT, and (c) FAN-OUT functions by means of the Fredkin gate." [FT82]

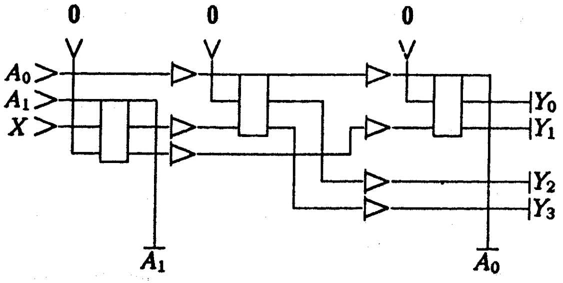

Figure II.13: "1-line-to 4-line demultiplexer. The 'address' lines A_0, A_1 specify to which of the four outputs $Y_0, \ldots, Y_3$ the 'data' signal X is to be routed. (Note that here the sink lines happen to echo the address lines.)" [FT82]

B.6 Constants and garbage

¶1. The Fredkin gate can be used to compute non-invertible functions such as AND, if we are willing to provide appropriate constants and to accept unwanted outputs (see Fig. II.10).

¶2. In general, one function can be embedded in another by providing appropriate constants from a *source* and ignoring some of the outputs, the *sink*, which are considered *garbage*.

¶3. However, this garbage cannot be thrown away (which would dissipate energy), so it must be recycled in some way.

B.7 Universality

¶1. **OR, NOT, and FAN-OUT:** Fig. II.12 shows Fredkin realizations of other common gates.

¶2. **Demultiplexer example:** Fig. II.13 shows a 1-line to 4-line demultiplexer.

¶3. Hence you can convert conventional logic circuits into conservative circuits, but the process is not very efficient. It's better to design the conservative circuit from scratch.

¶4. **Universality:** "any computation that can be carried out by a conventional sequential network can also be carried out by a suitable conservative-logic network, provided that an external supply of constants and an external drain for garbage are available."
(Will see how to relax these constraints: Sec. B.8)

B.8 Garbageless conservative logic

¶1. To reuse the apparatus for a new computation, we will have to throw away the garbage and provide fresh constants, both of which will dissipate energy.

¶2. **Exponential growth of garbage:** This is a significant problem if dissipative circuits are naively translated to conservative circuits because:
(1) the amount of garbage tends to increase with the number of gates, and
(2) with the naive translation, the number of gates tends to increase exponentially with the number of input lines.
"This is so because almost all boolean functions are 'random', i.e., cannot be realized by a circuit simpler than one containing an exhaustive look-up table."

¶3. However there is a way to make the garbage the same size as the input (in fact, identical to it).

¶4. First observe that a *combinational* conservative-logic network (one with no feedback loops) can be composed with its inverse to consume all garbage (Fig. II.14).

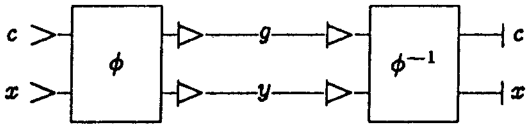

Figure II.14: Composition of combinational conservative-logic network with its inverse to consume the garbage. [fig. from FT82]

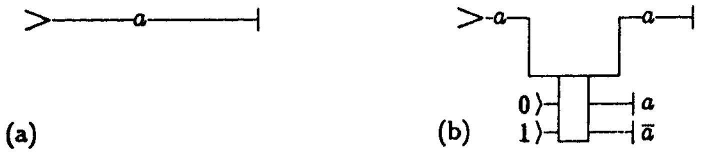

Figure II.15: The “spy circuit” for tapping into the output. Note that in the diagram the 0 and 1 constant inputs are switched (or, equivalently, the a and $\overline{a}$ outputs are switched). [fig. from FT82]

¶5. The desired output can be extracted by a “spy circuit” (Fig. II.15).

¶6. Fig. II.16 shows the general arrangement for garbageless computation. This requires the provision of n new constants (n = number output lines).

¶7. Consider the more schematic diagram in Fig. II.17.

¶8. Think of arranging tokens (representing 1-bits) in the input registers, both to represent the input x, but also a supply of n of them in the black lower square.

¶9. Run the computation.

¶10. The input argument tokens have been restored to their initial positions. The $2n$-bit string $00 \cdots 0011 \cdots 11$ in the lower register has been rearranged to yield the result and its complement $y\bar{y}$.

¶11. Restoring the $0 \cdots 01 \cdots 1$ inputs for another computation dissipates energy.

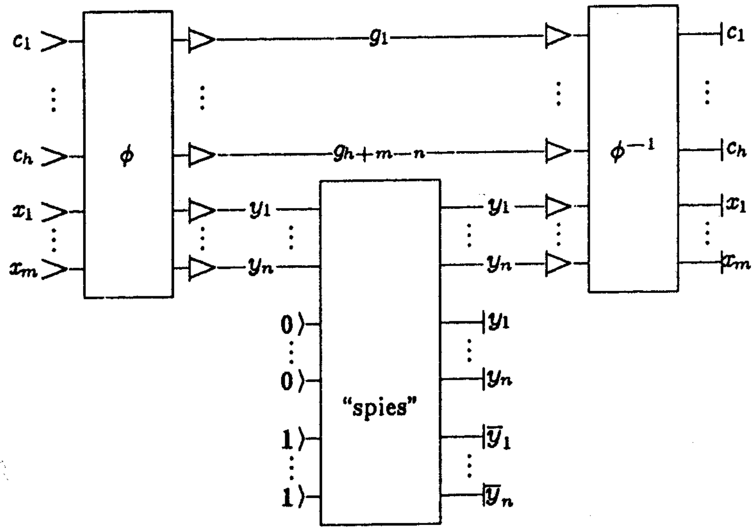

Figure II.16: Garbageless circuit. [FT82]

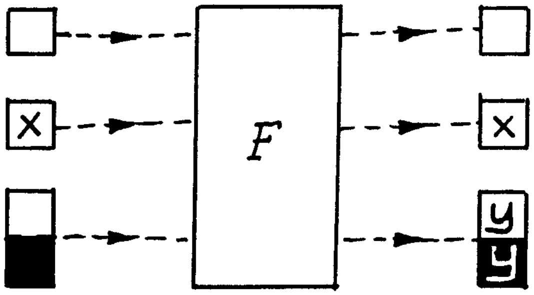

Figure II.17: "The conservative-logic scheme for garbageless computation. Three data registers are 'shot' through a conservative-logic black-box F. The register with the argument, x, is returned unchanged; the clean register on top of the figure, representing an appropriate supply of input constants, is used as a scratchpad during the computation (cf. the c and g lines in Figure [II.16]) but is returned clean at the end of the computation. Finally, the tokens on the register at the bottom of the figure are rearranged so as to encode the result y and its complement $\neg y$" [FT82]

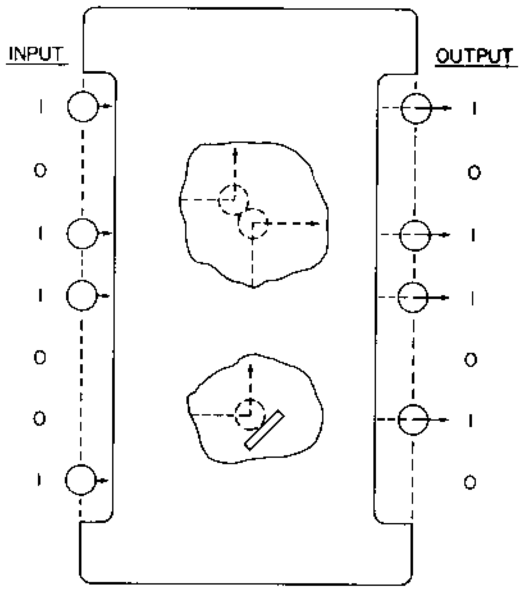

Figure II.18: Overall structure of ballistic computer. [B82]

¶12. **Feedback:** Finite loops can be unrolled, which shows that they can be done without dissipation.
(Cf. also that billiard balls can circulate in a frictionless system.)

B.9 Ballistic computation

“Consider a spherical cow moving in a vacuum. . . ”

¶1. **Billiard ball model:** To illustrate dissipationless *ballistic computation*, Fredkin and Toffoli defined a *billiard ball* model of computation.

¶2. It is based on the same assumptions as the classical kinetic theory of gasses: perfectly elastic spheres and surfaces.
In this case we can think of pucks on frictionless table.

¶3. Fig. II.18 shows the general structure of the billiard ball model.

¶4. 1s are represented by the presence of a ball at a location, and 0s by their absence.

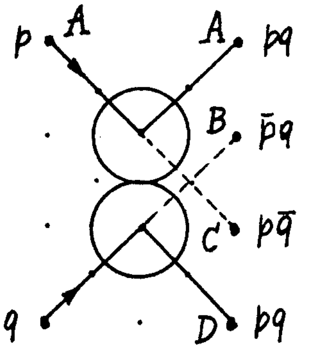

Figure II.19: “Billiard ball model realization of the interaction gate.” [FT82]

¶5. Input is provided by simultaneously firing balls into the input ports for the 1s in the argument.

¶6. Inside the box the balls ricochet off each other and fixed reflectors, which performs the computation.

¶7. After a fixed time delay, the balls emerging (or not) from the output ports define the output.

¶8. Obviously the number of 1s (balls) is conserved.

¶9. The computation is reversible because the laws of motion are reversible.

¶10. **Interaction gate:** Fig. II.19 shows the realization of the computational primitive, the *interaction gate.*

¶11. Fig. II.20 is the symbol for the interaction gate and its inverse.

¶12. **Universal:** The interaction gate is universal because it can compute both AND and NOT.

¶13. **Interconnections:** However, we must make provisions for arbitrary interconnections in a planar grid. So need to implement signal *crossover* and control *timing.*
(This is *non-trivial* crossover; *trivial* crossover is when two balls cannot possible be at the same place at the same time.)

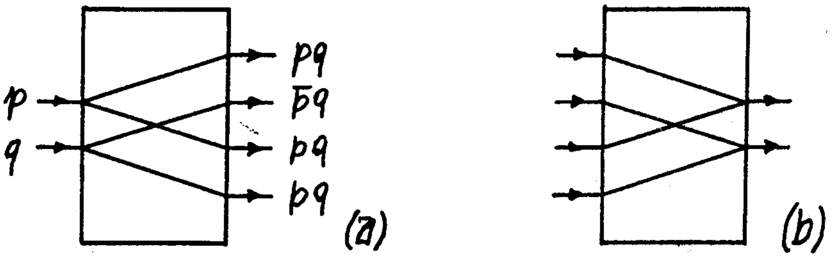

Figure II.20: "(a) The interaction gate and (b) its inverse." [FT82] Note that the second pq from the bottom should be $p\overline{q}$.

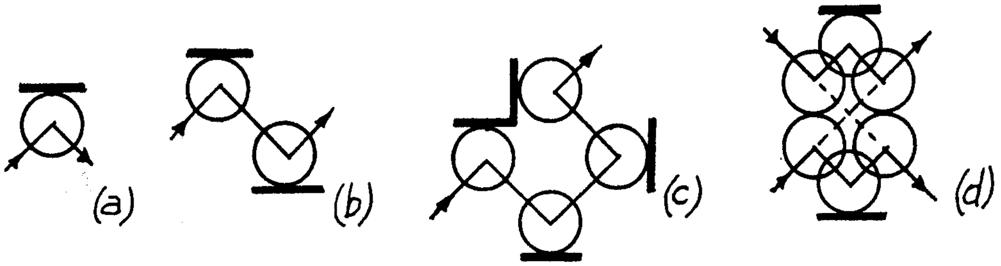

Figure II.21: "The mirror (indicated by a solid dash) can be used to deflect a ball's path (a), introduce a sideways shift (b), introduce a delay (c), and realize nontrivial crossover (d)." [FT82]

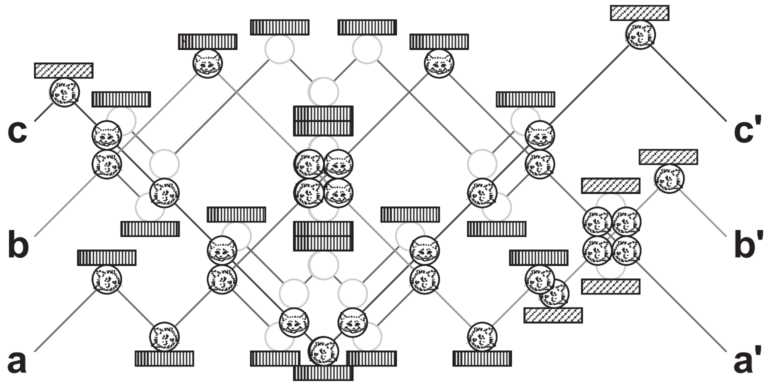

Figure II.22: Realization of the Fredkin gate in terms of multiple interaction gates. [NC]

¶14. Fig. II.21 shows mechanisms for realizing these functions.

¶15. Fig. II.22 shows a realization of the Fredkin gate in terms of multiple interaction gates. (The "bridge" indicates non-trivial crossover.)

¶16. **Practical problems:** Minuscule errors of any sort (position, velocity, alignment) will accumulate rapidly (by about a factor of 2 at each collision).

¶17. E.g., initial random error of $1/10^{15}$ in position or velocity (about what would be expected from uncertainty principle) would lead to a completely unpredictable trajectory after a few dozen collisions.
It will lead to a Maxwell distribution of velocities, as in a gas.

¶18. "Even if classical balls could be shot with perfect accuracy into a perfect apparatus, fluctuating tidal forces from turbulence in the atmosphere of nearby star would be enough to randomize their motion within a few hundred collisions." (Bennett 82, p. 910)

¶19. Various solutions have been considered, but they all have limitations.

¶20. "In summary, although ballistic computation is consistent with the laws of classical and quantum mechanics, there is no evident way to prevent

> the signals' kinetic energy from spreading into the computer's other degrees of freedom." (Bennett 82, p. 911)

¶21. Signals can be restored, but this introduces dissipation.