Embracing the Laws of Physics

Amr Sabry
Indiana University

MECO'2017

Bar, Montenegro,
11th-15th June, 2017

Abstraction vs. Reality (Sorting)

- Abstract sorting algorithm: easy, well-understood, completely formalized, taught in high school
- Sorting real data in the real world is still a challenge; Google petasort, 33 hours on 8000 computers, distributed, hardware component failure frequent, global synchronization impossible, speculative, only eventually consistent

Abstraction vs. Reality (Sortedness)

- In the abstract model, checking sortedness trivial
- If the real world with petabytes of evolving data, not clear it is even possible to check sortedness; easy to get inconsistent results

Abstraction vs. Reality (Privacy)

- In the real world, differential privacy achieved by perturbing the output by a suitable amount of noice
- The very idea of noise is abstracted away in our computational models

Abstraction vs. Reality (Security)

- Proving the security properties of an abstract protocol is routine; can be used to conclude that an isolated device is secure
- Real devices (e.g. RFID devices, electronic passports, etc.) have power consumption and electromagnetic signatures that are abstracted away; attacks based on DPA (differential power analysis) and DEMA (differential electromagnetic analysis) exit.

Information Flow Security

- Consider a tiny 2-bit password = "10". The password checker looks like:

```
check-password (guess) =
  if guess == "10"
  then True
  else False
```

- How much information is leaked by this program?

- Assume attacker has no prior knowledge except that the password is 2 bits, i.e., the four possible 2-bits are equally likely.

Information Flow Security

- If the attacker guesses "10" (with probability $1/4$) the password (2 bits) is leaked.
- If attacker guesses one of the other choices (with probability $3/4$) the number of possibilities is reduced from 4 to 3, i.e., the attacker learns $\log 4 - \log 3$ bits of information.
- So in general the attacker learns:

$$
\begin{array}{rl}
 & 1/4 * 2 + 3/4(\log 4 - \log 3) \\
= & 1/4 \log 4 + 3/4 \log 4/3 \\
= & -1/4 \log 1/4 - 3/4 \log 3/4 \\
\sim & 0.8 \text{ bits in the first probe}
\end{array}
$$

Information Flow Security

- Another way to look at the password checker.
- Shannon's entropy
- Input is a random variable; 4 possibilities; uniform distribution = 2 bits of information
- Output is another random variable; 4 possibilities; distribution { (*True*, 1/4), (*False*, 3/4) } = 0.8 bits of information.
- Where did the 1.2 bits of information go???

Information Flow Security

- Where did the 1.2 bits of information go???
- They must have been erased by the function
- The energy in the information dissipates as heat.

Models of Computation

Mathematical models of computation are abstract constructions, by their nature unfettered by physical laws. However, if these models are to give indications that are relevant to concrete computing, they must somehow capture, albeit in a selective and stylized way, certain general physical restrictions to which all concrete computing processes are subjected.

(Toffoli 1980)

Cyber-Physical Systems - Are Computing Foundations Adequate?

Edward A. Lee

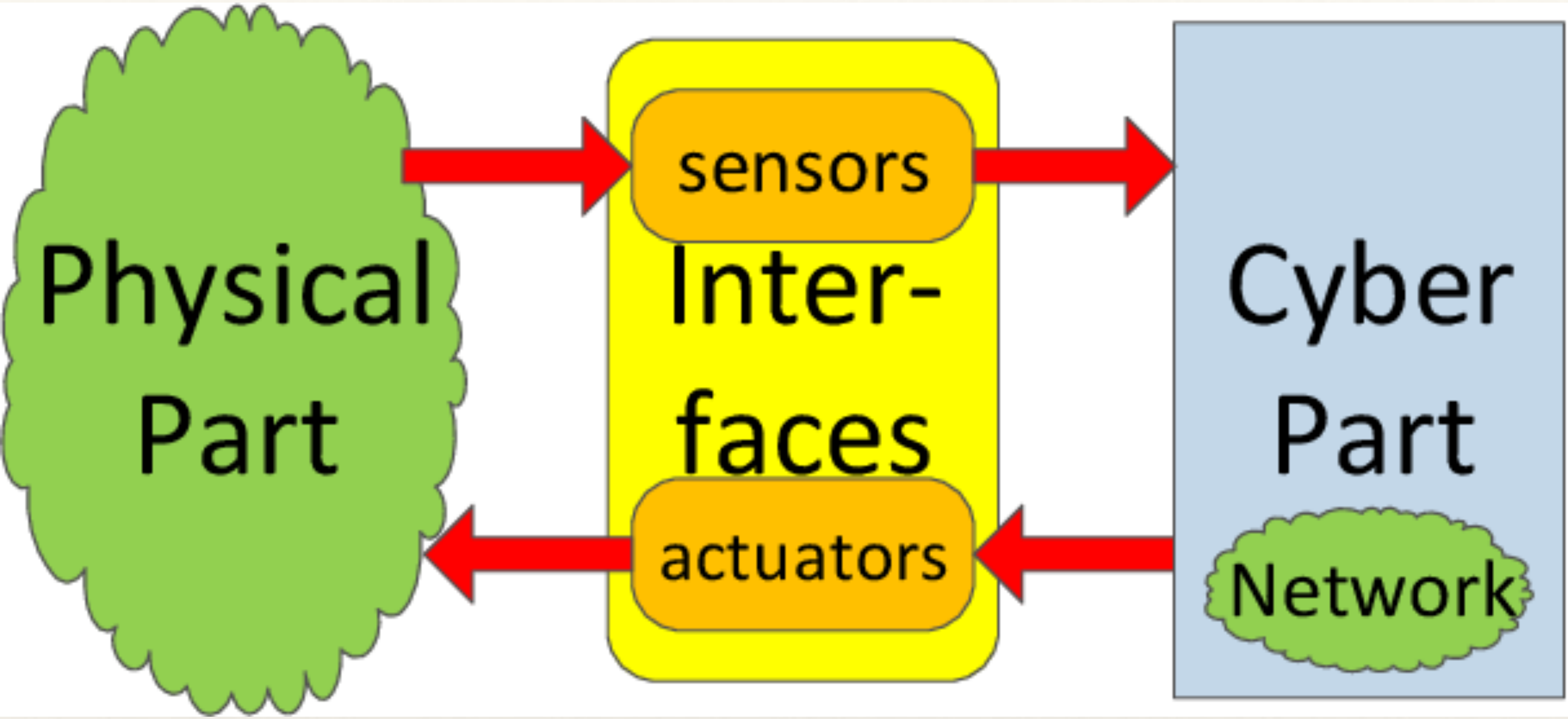
Department of EECS, UC Berkeley

Position Paper for
NSF Workshop On Cyber-Physical Systems:
Research Motivation, Techniques and Roadmap
October 16 - 17, 2006
Austin, TX

1 Summary

Cyber-Physical Systems (CPS) are integrations of computation with physical processes. Embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice versa. In the physical world, the passage of time is inexorable and concurrency is intrinsic. Neither of these properties is present in today's computing and networking abstractions.

I argue that the mismatch between these abstractions and properties of physical processes impede technical progress, and I identify promising technologies for research and investment. There are technical approaches that partially bridge

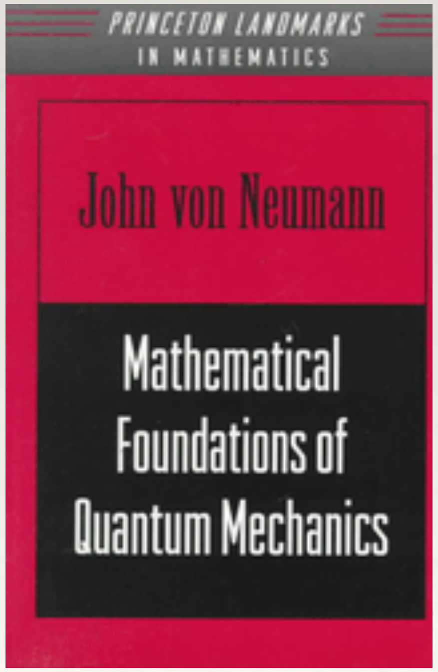


From http://www.ics.uci.edu/~steffenp/

Physics and Computation

Physics

- Most accurate theory known to us is ***Quantum Mechanics***



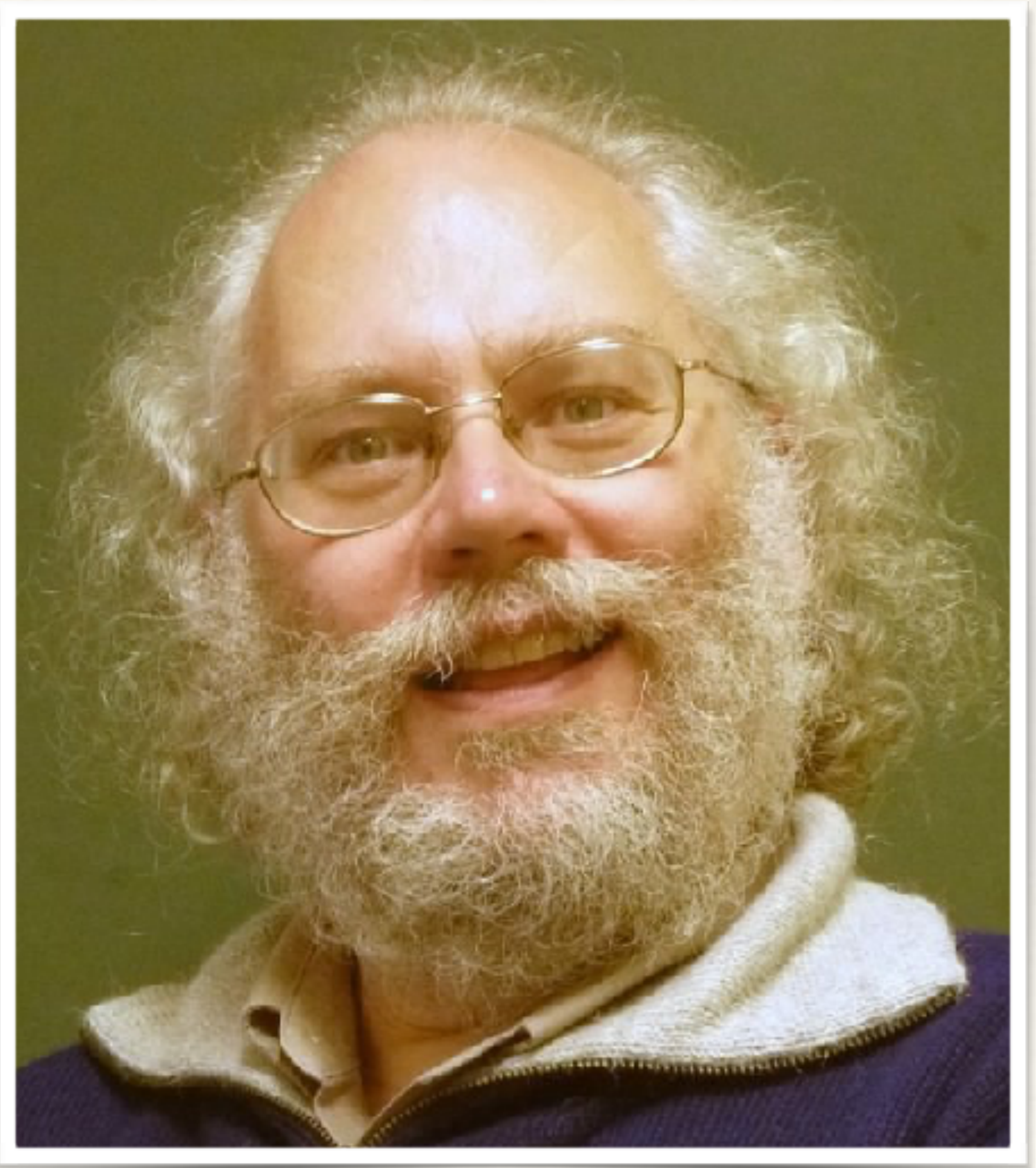
Computation

- Universal model of computation is based on the Von Neumann architecture (*)
- Equivalent to Turing machines, the λ-calculus, and many other models
- (*) But… listen to the next speaker!!!

First Draft of a Report on the EDVAC
by John von Neumann,
Contract No. W-670-ORD-4926,
Between the United States Army Ordinance Department
and the University of Pennsylvania Moore School of Electrical Engineering
University of Pennsylvania
June 30, 1945

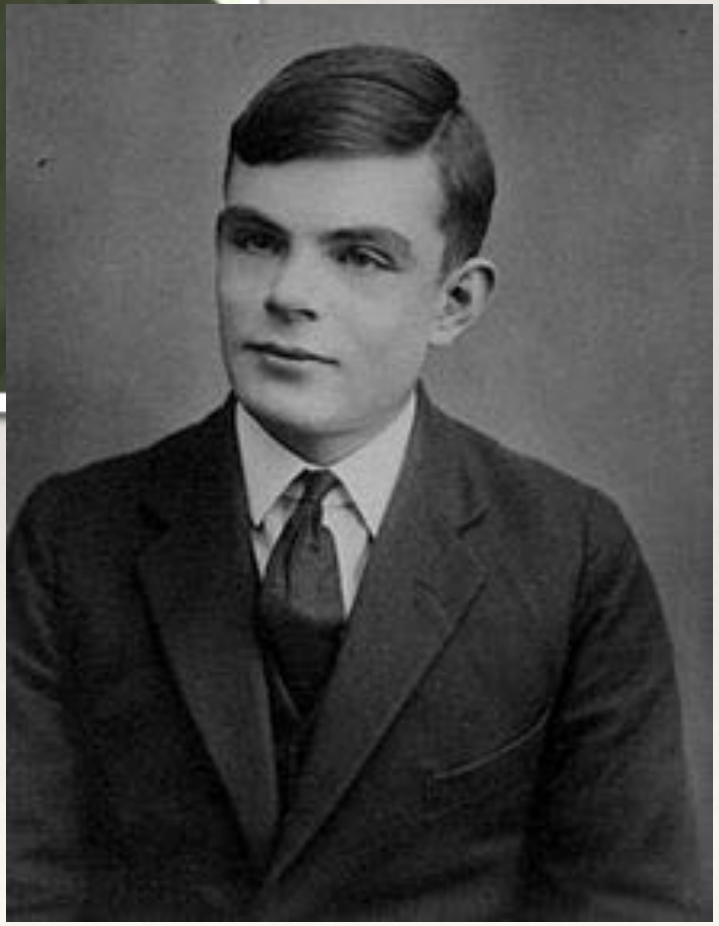
Physics Worldview

- Quantum mechanics implies the existence of an efficient (polynomial) algorithm for factoring integers (Shor)
- RSA is not secure.



Computer Science Worldview

- Extended Church-Turing Thesis: Everything that is computable in Nature is computable by a von Neumann machine, and is not exponentially faster.
- Despite intense interest and effort, no one knows how to factor integers efficiently in that model.
- RSA is secure.



Something is Wrong

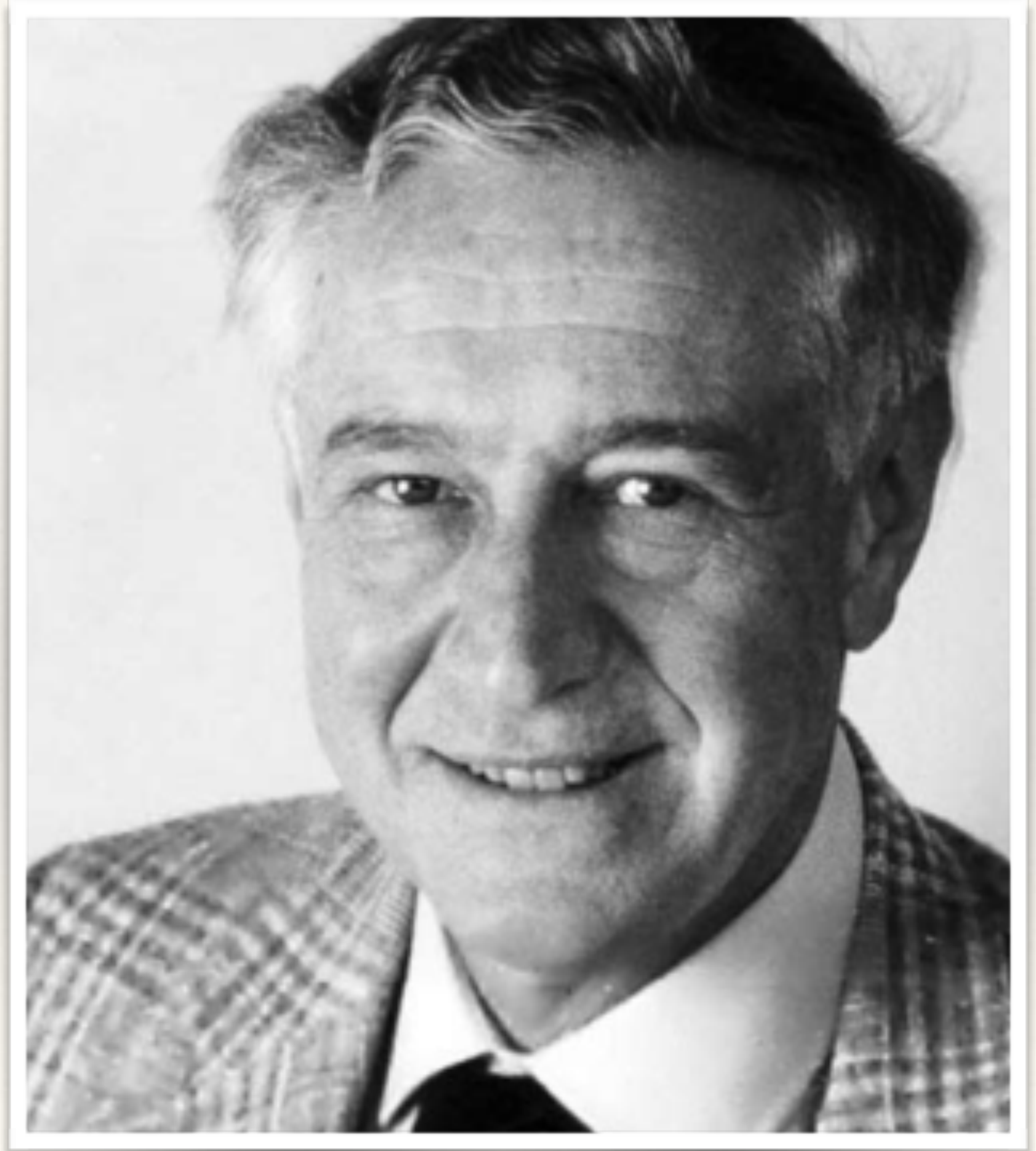
- Either Shor's algorithm is not "natural". Textbook quantum mechanics is wrong;
- or, the von Neumann architecture is not universal. There are other "natural" computing models that are exponentially faster;
- or, computer scientists have not been clever enough to find an efficient factoring algorithm;
- or, everybody is wrong

Possibility I: Revise quantum mechanics

The mathematician's vision of an unlimited sequence of totally reliable operations is unlikely to be implementable in this real universe.

But the real world is unlikely to supply us with unlimited memory or unlimited Turing machine tapes. Therefore, continuum mathematics is not executable, and physical laws which invoke that can not really be satisfactory. They are references to illusionary procedures.

(Landauer 1996 and 1999)



Possibility I: Revise quantum mechanics

I want to talk about the possibility that there is to be an exact simulation, that the computer will do exactly the same as nature. If this is to be proved and the type of computer is as I've already explained, then it's going to be necessary that everything that happens in a finite volume of space and time would have to be exactly analyzable with a finite number of logical operations. The present theory of physics is not that way, apparently. It allows space to go down into infinitesimal distances, wavelengths to get infinitely great, terms to be summed in infinite order, and so forth; and therefore, if this proposition is right, physical law is wrong.

(Feynman 1981)

Possibility II: Revise Computer Science

Another thing that had been suggested early was that natural laws are reversible, but that computer rules are not. But this turned out to be false; the computer rules can be reversible, and it has been a very, very useful thing to notice and to discover that. This is a place where the relationship of physics and computation has turned itself the other way and told us something about the possibilities of computation. So this is an interesting subject because it tells us something about computer rules…

(Feynman 1981)

Possibility II: Revise Computer Science

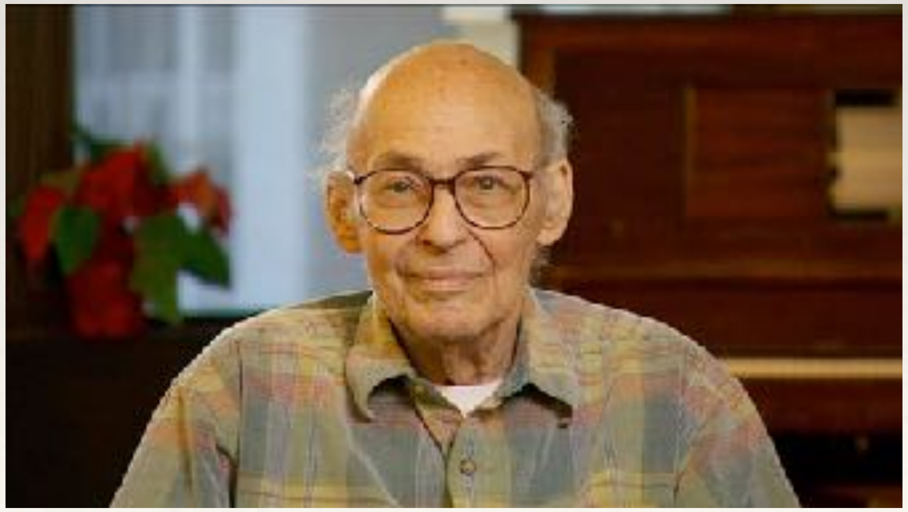
Turing hoped that his abstracted-paper-tape model was so simple, so transparent and well defined, that it would not depend on any assumptions about physics that could conceivably be falsified, and therefore that it could become the basis of an abstract theory of computation that was independent of the underlying physics. 'He thought,' as Feynman once put it, 'that he understood paper.' But he was mistaken. Real, quantum-mechanical paper is wildly different from the abstract stuff that the Turing machine uses. The Turing machine is entirely classical, and does not allow for the possibility the paper might have different symbols written on it in different universes, and that those might interfere with one another.

(Deutsch 1985)

Possibility II: Revise Computer Science

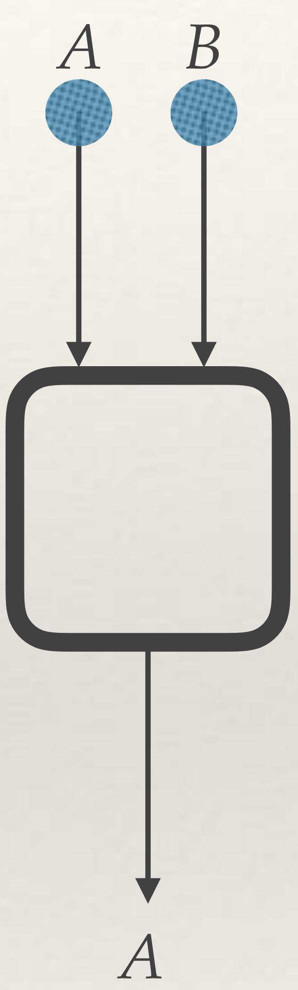
Ed Fredkin pursued the idea that information must be finite in density. One day, he announced that things must be even more simple than that. He said that he was going to assume that information itself is conserved. "You're out of you mind, Ed." I pronounced. "That's completely ridiculous. Nothing could happen in such a world. There couldn't even be logical gates. No decisions could ever be made." But when Fredkin gets one of his ideas, he's quite immune to objections like that; indeed, they fuel him with energy. Soon he went on to assume that information processing must also be reversible — and invented what's now called the Fredkin gate.

(Minsky 1999)



Conservation of Information

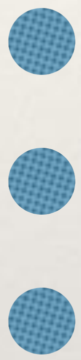
- None of the common foundational models of computation is based on conservation of information
  - Circuit model has AND, OR, etc gates that lose information;
  - Turing Machine allows one to overwrite a cell losing information;
  - λ-calculus allows functions that throw away their arguments losing information;
  - etc etc etc



Laundauer's principle:
Erasure of information generates heat!!!

Fredkin Gate

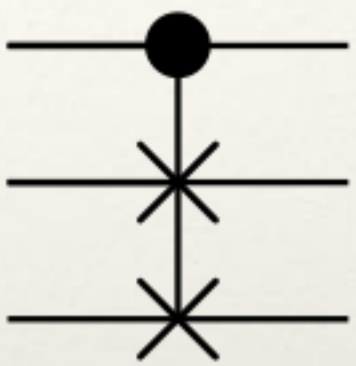
$c =$ *false* *true*



c C

x X

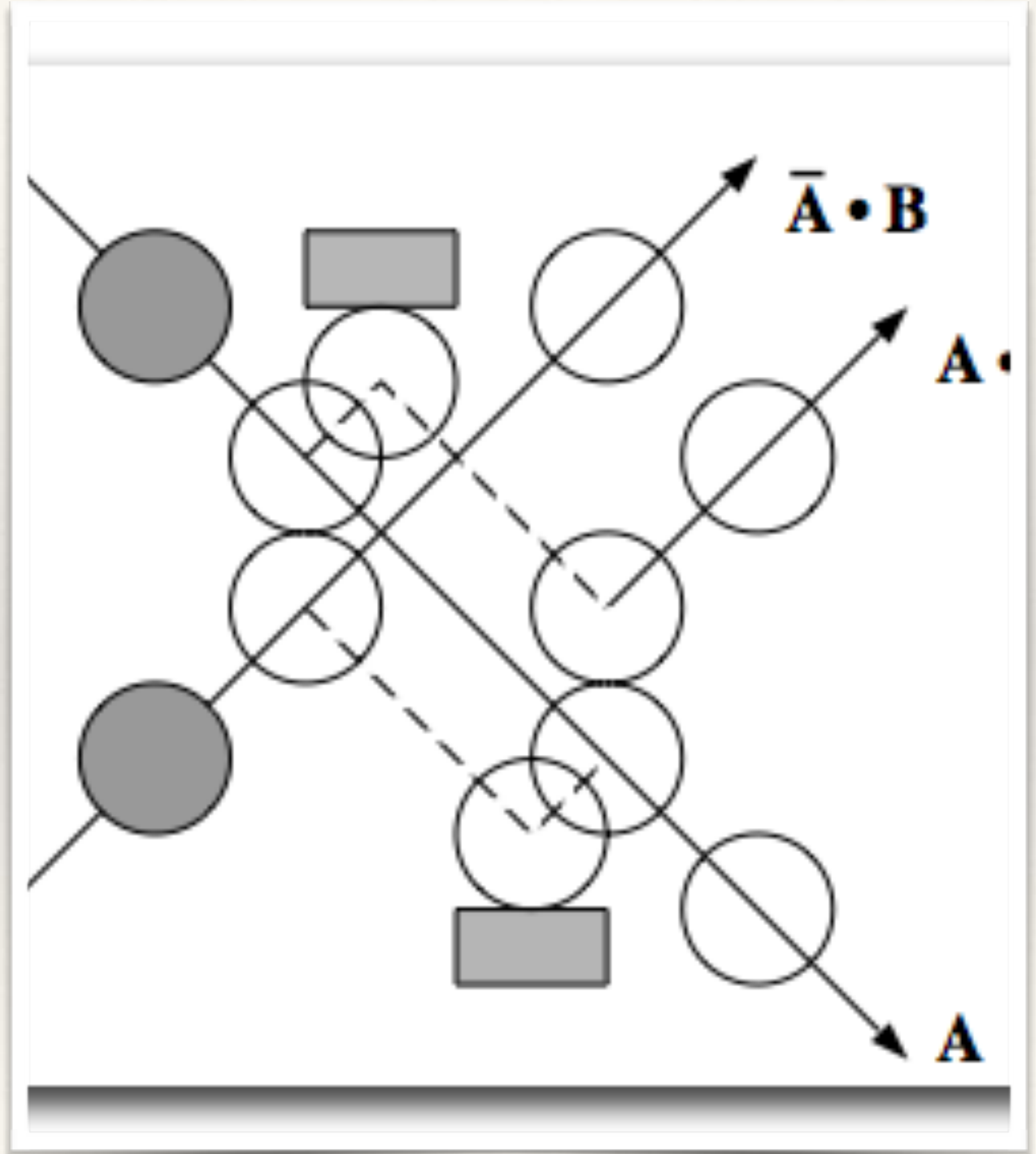
y Y



If $y = \mathit{false}$, then $Y = c$ AND x
If $x = \mathit{false}$, $y = \mathit{true}$, then $y =$ NOT c
If $y = \mathit{true}$, then $X = c$ OR x

Conservation of Information

- Is it just a low-level concern, a curiosity, an esoteric model, or is it really foundational?
- We argue that, if taken seriously, it revolutionizes the theory and practice of computation, and even its logical foundations.
- This perspective has far reaching implications in many high-level applications



Logic

~200 years ago

An Investigation of the Laws of Thought, on which are Founded the Mathematical Theories of Logic and Probabilities, G. Boole, 1853.

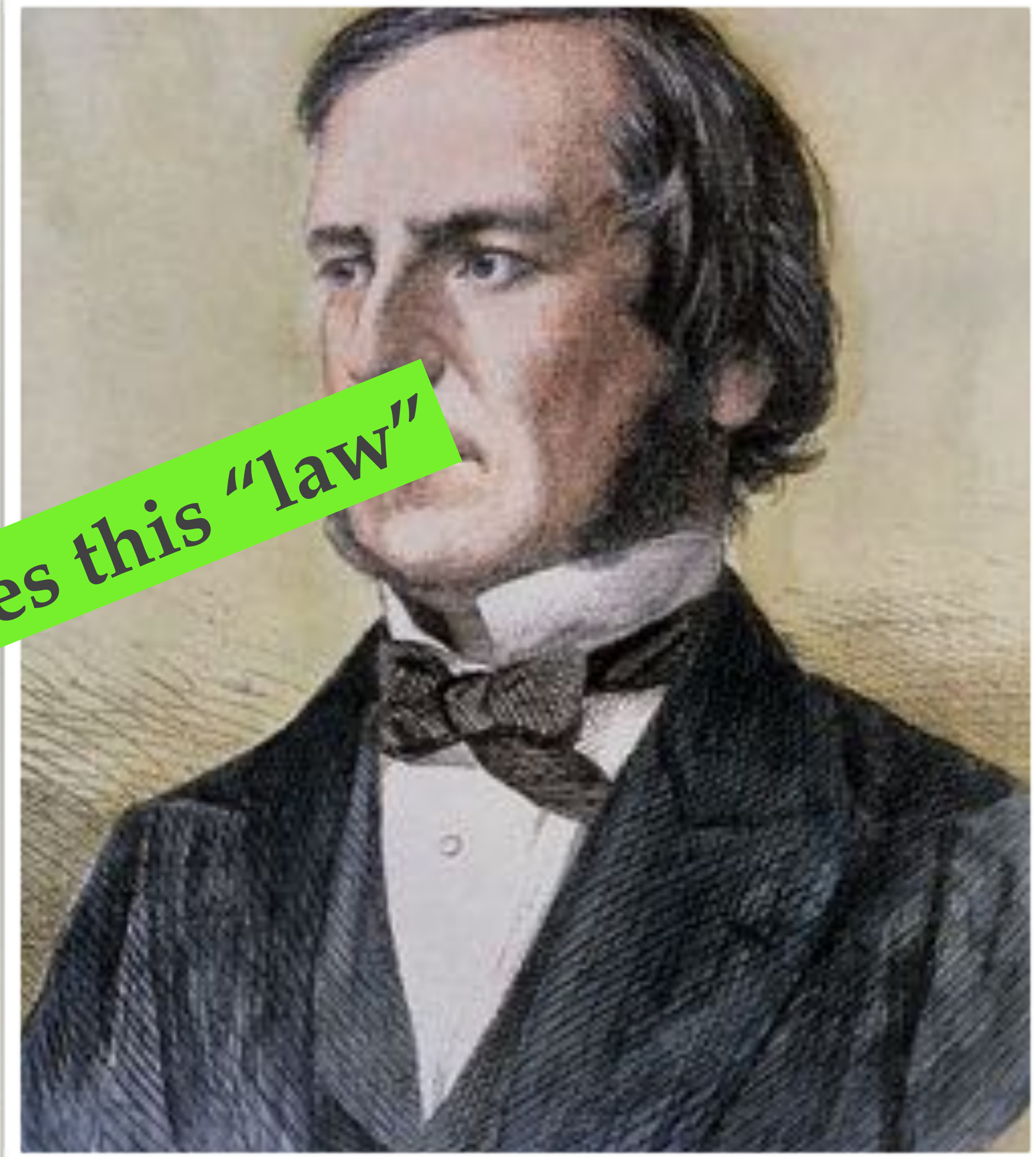
Opening sentence of Chapter 1:

The design of the following treatise is to investigate the fundamental laws of those operations of the mind by which reasoning is performed; . . .

A few chapters later:

Proposition IV. That axiom of [illegible]ans which is termed the pri[illegible]ntradiction, and which affir[illegible]mpossible for any being to [illegible]uality, and at the same ti[illegible] possess it, is a consequence of the fu[illegible]mental law of thought, whose expression is $x^2 = x$.

Quantum superposition violates this "law"



~2300 years ago

- Stoics invent the propositional calculus
- Re-invented in the 12th century
- Re-re-invented by Leibniz in the 17-18th century
- Re-re-re-invented by Boole and de-Morgan in the 19th century
- Improved by Gentzen and Łukasiewicz in the 20th century

Modus Ponens

$$\frac{P \longrightarrow Q \qquad\qquad P}{Q}$$

X is a man implies X is mortal
Socrates is a man
Hence Socrates is moral.

Obvious logical inference rule. Isn't it?

Does it have anything to do with physics?

The Curry-Howard Isomorphism

The proposition $P \rightarrow Q$ is isomorphic to a function that takes a value of type P as an argument and returns a value of type Q as a result.

$$\frac{P \longrightarrow Q \qquad P}{Q}$$

Modus Ponens is *function application*. In programming terms *this is a computational physical process!*

Logic can be revisited

In other terms, what is so good in logic that quantum physics should obey? Can't we imagine that our conceptions about logic are wrong, so wrong that they are unable to cope with the quantum miracle? […] Instead of teaching logic to nature, it is more reasonable to learn from her. Instead of interpreting quantum into logic, we shall interpret logic into quantum.

(Girard 2007)

Linear Logic

When a function $P \longrightarrow Q$ is applied to a value to type P, it consumes the value of type P.

The value of type P cannot be used again; it is gone!

$SodaMachine : Money \longrightarrow Candy$

After we say *SodaMachine(€1.50)* our money has been used; it is gone.

Logic, Physics, Computation

No matter where you start, you are led to these principles:

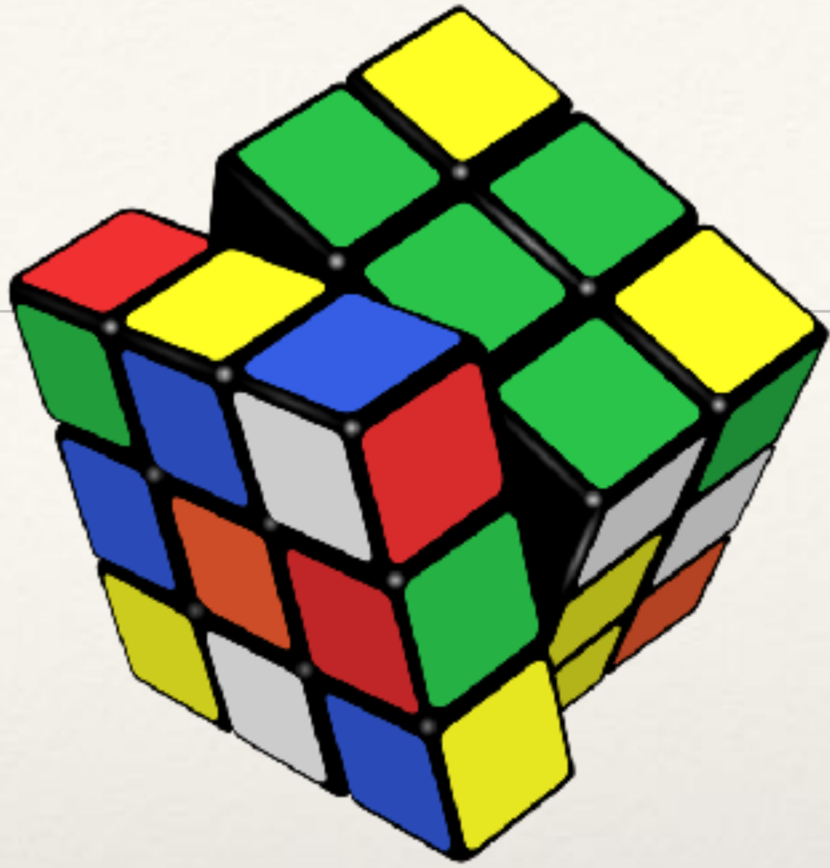
No creation of information

No duplication of information

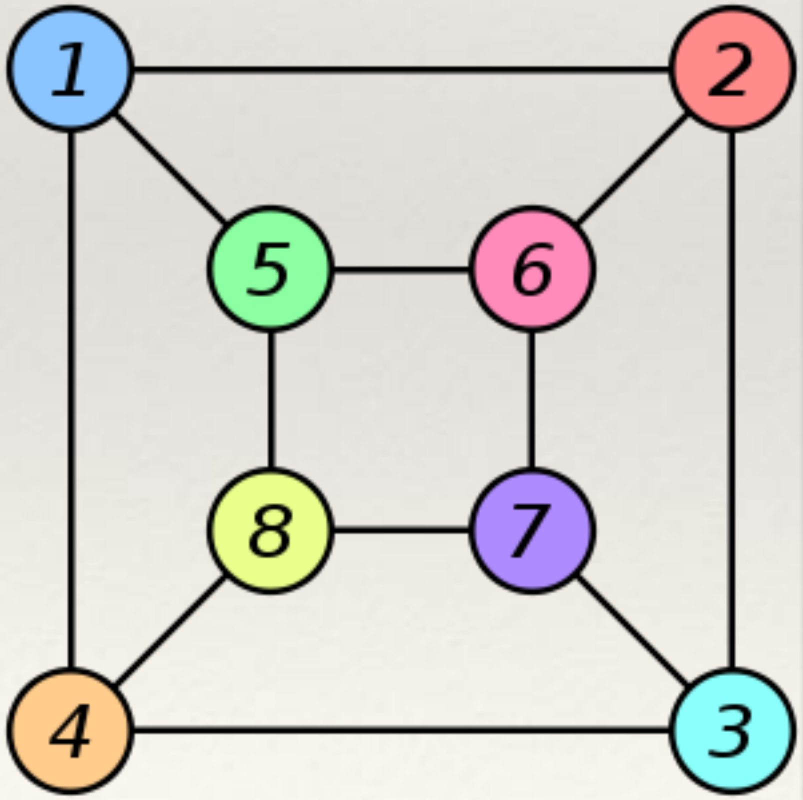
Conservation of information

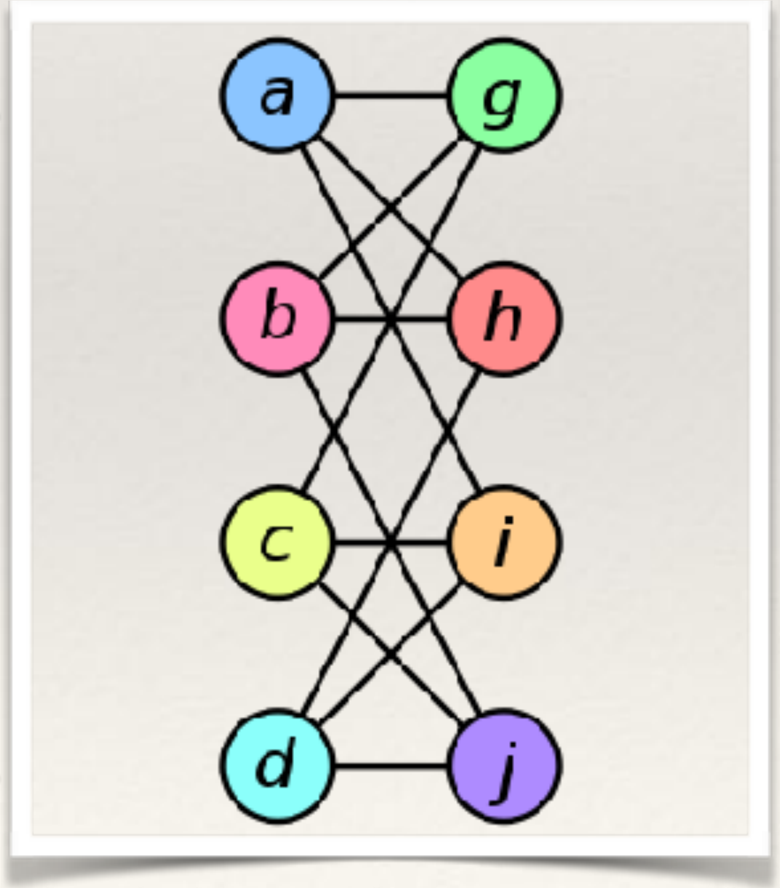
A Reversible Model of Computation and a Reversible Programming Language and their Reversible Logic

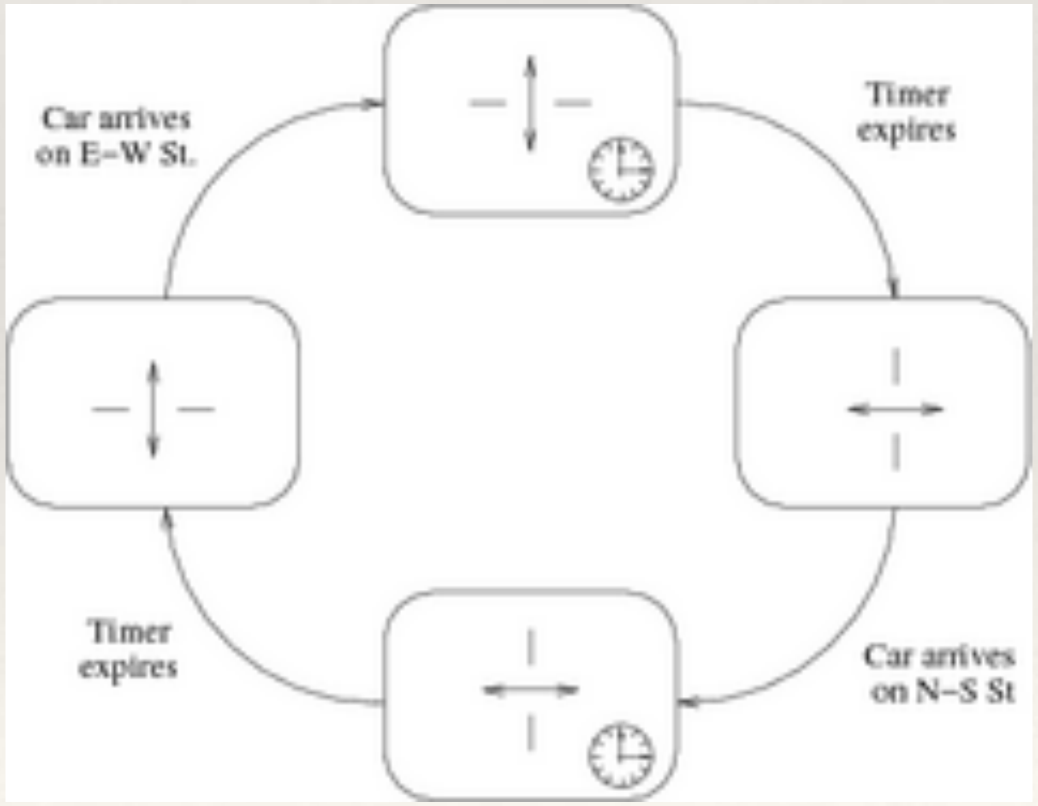
Key Idea



- All programs/proofs/ deductions are isomorphisms / equivalences







From Irreversible to Reversible

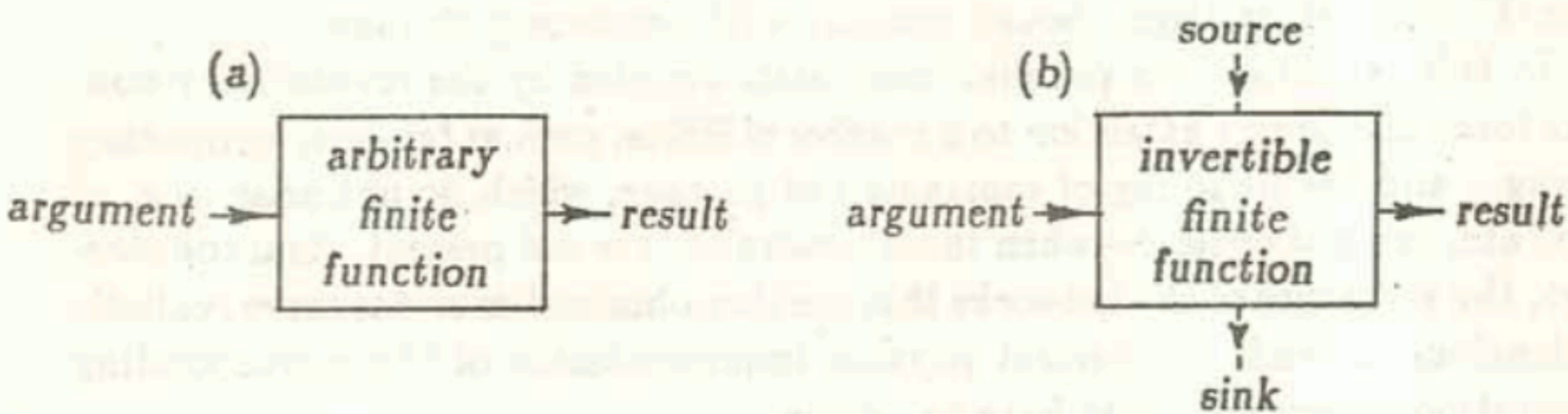
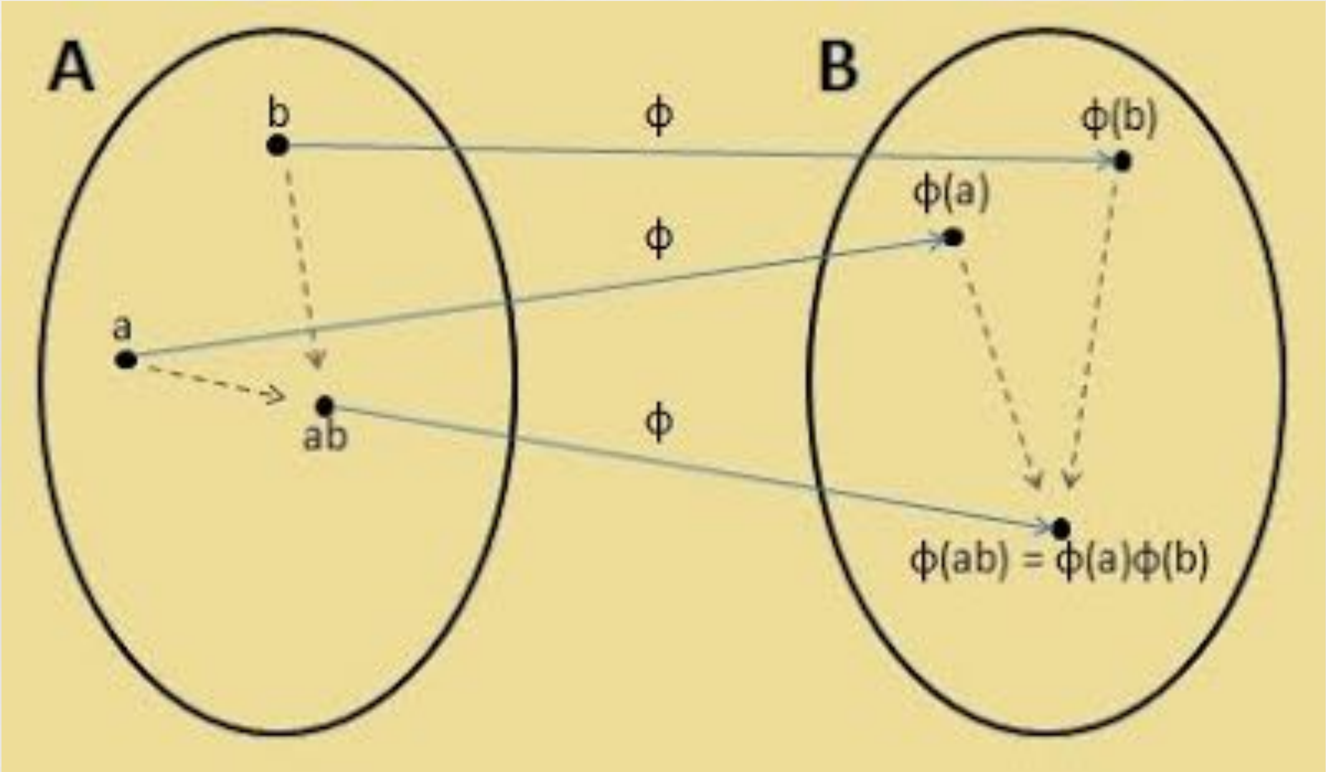


Fig. 4.2 Any finite function (a) can be realized as an invertible finite function (b) having a number of auxiliary input lines which are fed with constants and a number of auxiliary output lines whose values are disregarded.

(Toffoli 1980)

Type Isomorphisms

- Some problems are naturally reversible.
- Those that are not can be embedded in reversible problems.
- When it comes to writing programs to solve these problems, all is needed is a language for expressing equivalences.
- Technically a language that is sound and complete with respect to isomorphisms between types.



Sound and Complete Type Isomorphisms (for finite types)

$$
\begin{array}{rcll}
0 + b & \rightleftharpoons & b & \textit{identity for } + \\
b_1 + b_2 & \rightleftharpoons & b_2 + b_1 & \textit{commutativity for } + \\
b_1 + (b_2 + b_3) & \rightleftharpoons & (b_1 + b_2) + b_3 & \textit{associativity for } +
\end{array}
$$

$$
\begin{array}{rcll}
1 \times b & \rightleftharpoons & b & \textit{identity for } \times \\
b_1 \times b_2 & \rightleftharpoons & b_2 \times b_1 & \textit{commutativity for } \times \\
b_1 \times (b_2 \times b_3) & \rightleftharpoons & (b_1 \times b_2) \times b_3 & \textit{associativity for } \times
\end{array}
$$

$$
\begin{array}{rcll}
0 \times b & \rightleftharpoons & 0 & \textit{distribute over } 0 \\
(b_1 + b_2) \times b_3 & \rightleftharpoons & (b_1 \times b_3) + (b_2 \times b_3) & \textit{distribute over } +
\end{array}
$$

$$
\frac{}{b_1 \rightleftharpoons b_1} \qquad \frac{b_1 \rightleftharpoons b_2}{b_2 \rightleftharpoons b_1} \qquad \frac{b_1 \rightleftharpoons b_2 \quad b_2 \rightleftharpoons b_3}{b_1 \rightleftharpoons b_3}
$$

$$
\frac{b_1 \rightleftharpoons b_3 \quad b_2 \rightleftharpoons b_4}{(b_1 + b_2) \rightleftharpoons (b_3 + b_4)} \qquad \frac{b_1 \rightleftharpoons b_3 \quad b_2 \rightleftharpoons b_4}{(b_1 \times b_2) \rightleftharpoons (b_3 \times b_4)}
$$

Name the Isomorphisms

$$
\begin{array}{rrcll}
\mathit{zeroe}: & 0 + b & \rightleftharpoons & b & : \mathit{zeroi} \\
\mathit{swap}^{+}: & b_1 + b_2 & \rightleftharpoons & b_2 + b_1 & : \mathit{swap}^{+} \\
\mathit{assocl}^{+}: & b_1 + (b_2 + b_3) & \rightleftharpoons & (b_1 + b_2) + b_3 & : \mathit{assocr}^{+} \\
\\
\mathit{unite}: & 1 \times b & \rightleftharpoons & b & : \mathit{uniti} \\
\mathit{swap}^{\times}: & b_1 \times b_2 & \rightleftharpoons & b_2 \times b_1 & : \mathit{swap}^{\times} \\
\mathit{assocl}^{\times}: & b_1 \times (b_2 \times b_3) & \rightleftharpoons & (b_1 \times b_2) \times b_3 & : \mathit{assocr}^{\times} \\
\\
\mathit{distrib}_0: & 0 \times b & \rightleftharpoons & 0 & : \mathit{factor}_0 \\
\mathit{distrib}: & (b_1 + b_2) \times b_3 & \rightleftharpoons & (b_1 \times b_3) + (b_2 \times b_3) & : \mathit{factor}
\end{array}
$$

$$
\frac{}{\mathit{id} : b \rightleftharpoons b} \qquad \frac{c : b_1 \rightleftharpoons b_2}{\mathit{sym}\ c : b_2 \rightleftharpoons b_1} \qquad \frac{c_1 : b_1 \rightleftharpoons b_2 \quad c_2 : b_2 \rightleftharpoons b_3}{(c_1 \fatsemi c_2) : b_1 \rightleftharpoons b_3}
$$

$$
\frac{c_1 : b_1 \rightleftharpoons b_3 \quad c_2 : b_2 \rightleftharpoons b_4}{(c_1 + c_2) : (b_1 + b_2) \rightleftharpoons (b_3 + b_4)} \qquad \frac{c_1 : b_1 \rightleftharpoons b_3 \quad c_2 : b_2 \rightleftharpoons b_4}{(c_1 \times c_2) : (b_1 \times b_2) \rightleftharpoons (b_3 \times b_4)}
$$

Example

The type isomorphism:

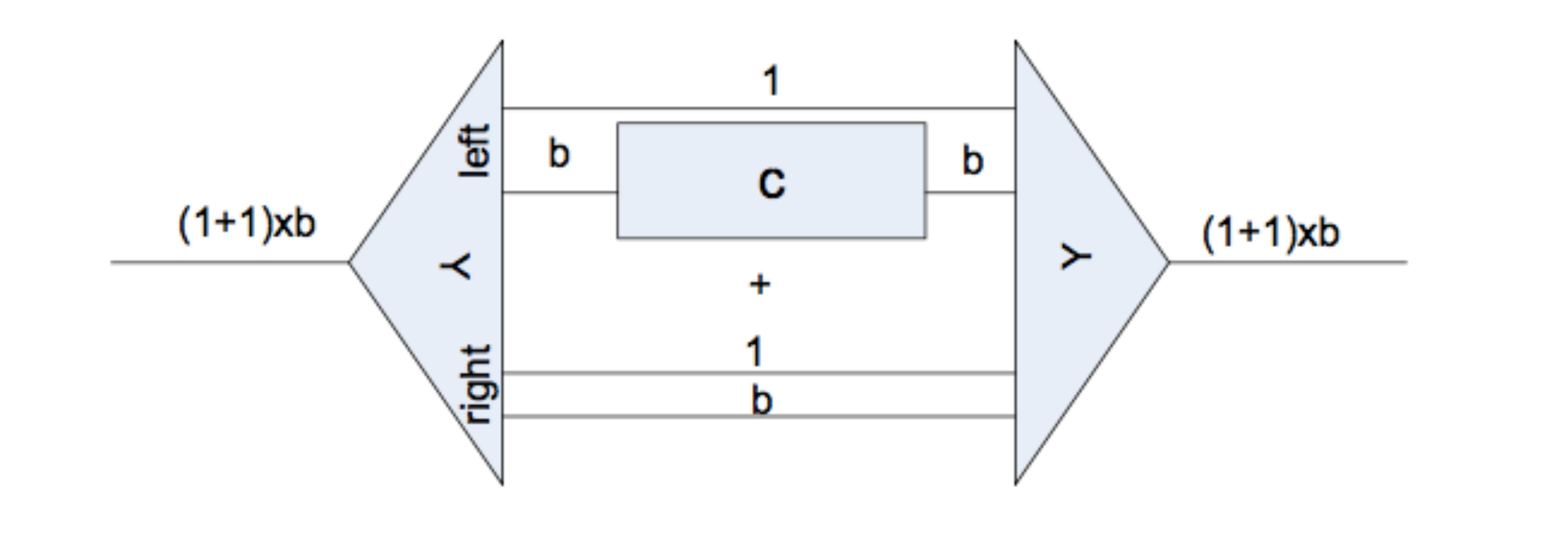
$$
\begin{array}{ll}
 & (1+1)\times((1+1)\times b) \\
= & (1+1)\times((1\times b)+(1\times b)) \\
= & (1+1)\times(b+b) \\
= & (1\times(b+b))+(1\times(b+b)) \\
= & (b+b)+(b+b)
\end{array}
$$

corresponds to the program:

$$(id \times (distrib \mathbin{;} (unite \times unite))) \mathbin{;} (distrib \mathbin{;} (unite \times unite))$$

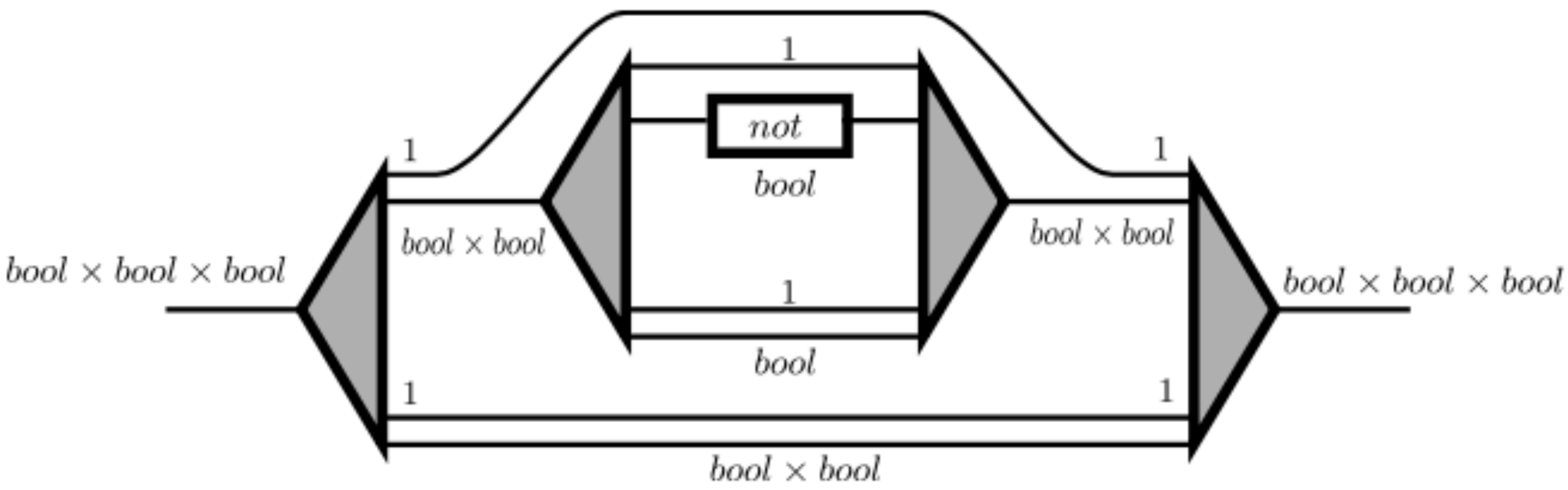
Conditionals

$$\mathbf{if}_c : bool \times b \rightleftharpoons bool \times b$$
$$\mathbf{if}_c = distrib \fatsemi ((id \times c) + id) \fatsemi factor$$



Toffoli and Fredkin Gates

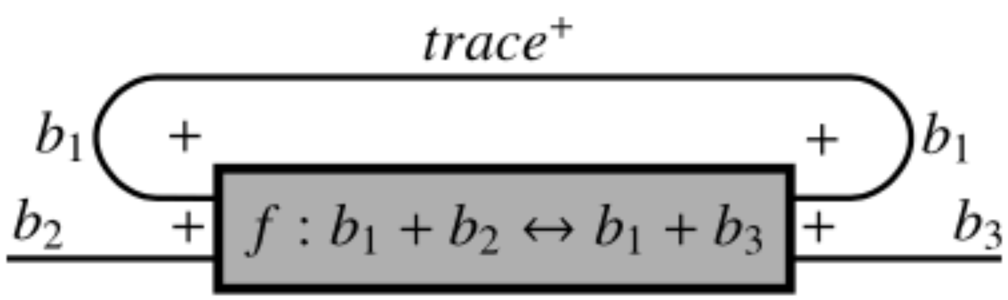
- Fredkin gate: $\mathbf{if}_{swap^{\times}}$
- Toffoli gate: $\mathbf{if}_{\mathbf{if}_{swap^{+}}}$



Recursion

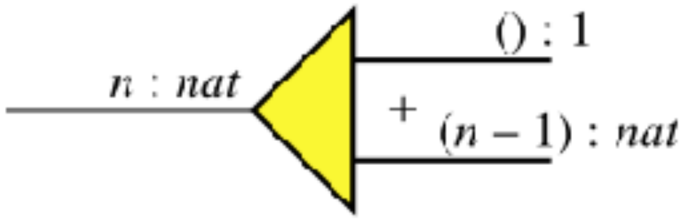
- Add recursive types (so that we can now define natural numbers, lists, trees, etc.)

- Add categorical trace (the categorical abstraction of feedback, looping, and recursion)

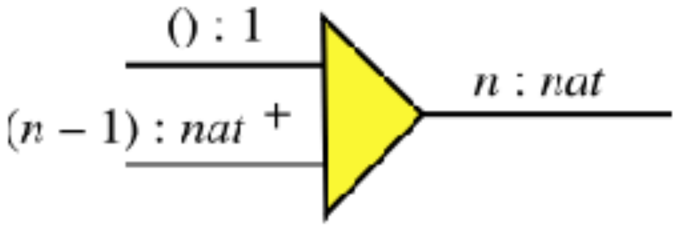


Numbers

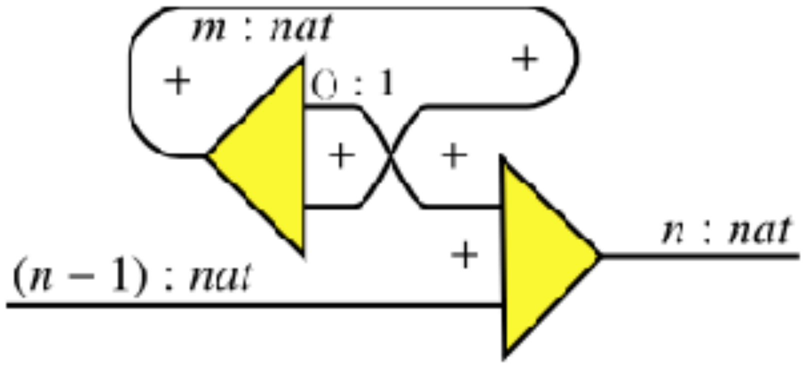
- Unfolding a natural number:



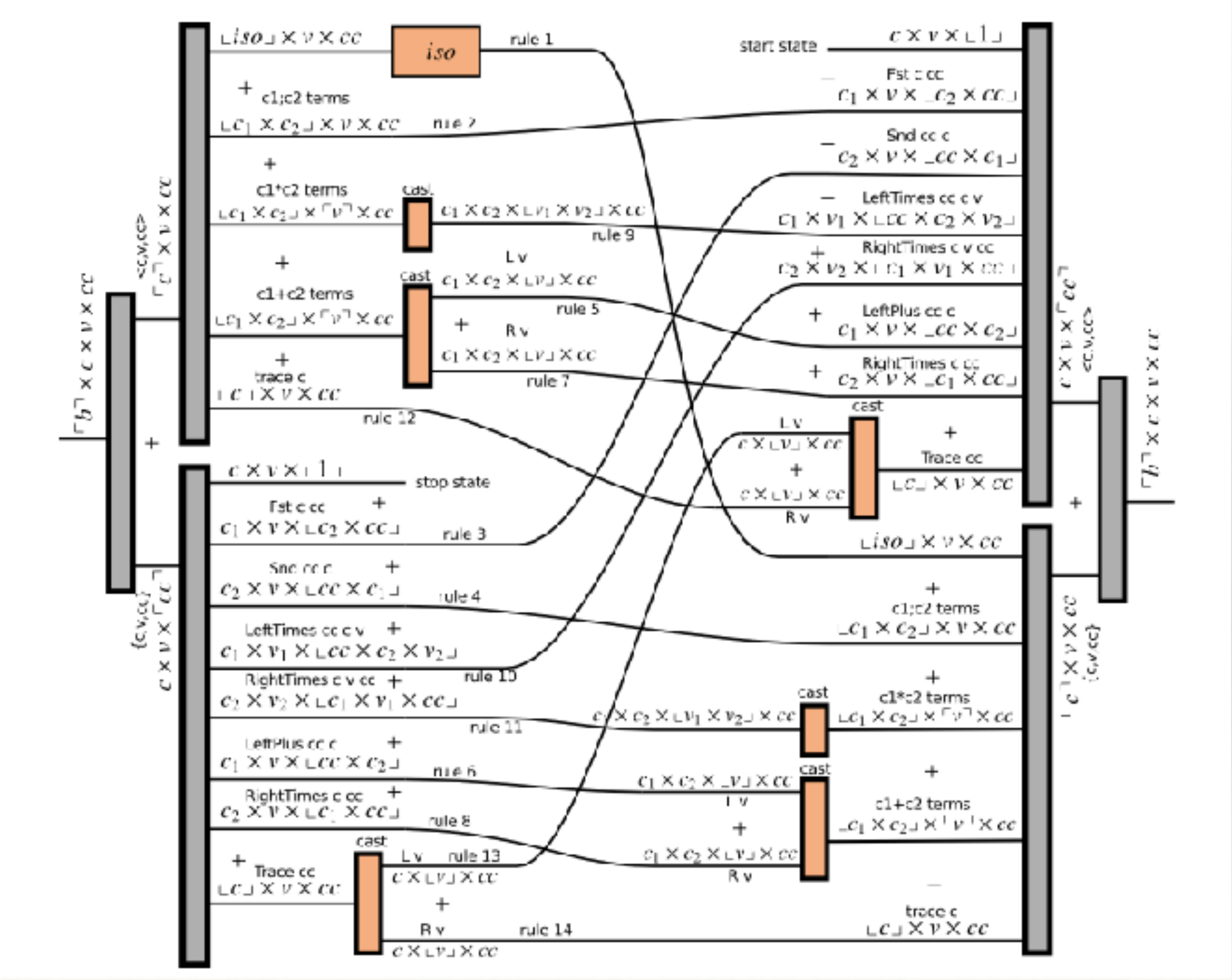
- Folding a natural number:



- add1/sub1 (partial isomorphisms)



A meta-circular interpreter



Closed vs. Open Systems

- So far we have modeled closed systems in which information is preserved;
- An open system may have interactions with its environment;
- We will show that these interactions can be modeled with two new operations that interact with the environment via side effects:
  - *create* : $1 \rightharpoonup b$
  - *erase* : $b \rightharpoonup 1$
- The effects are isolated using an arrow metalanguage which is a standard way of modeling computational effects.
- A compiler can take a conventional language with implicit effects and produce a version in which the effects are explicit (POPL 2012)

Password Checker again

$nand(b_1, b_2) = \textbf{if } b_1 \textbf{ then } not\ b_2 \textbf{ else } true$

The most optimal implementation of *nand* must erase at least 1.2 bits i.e. at least two *bools* must be erased.

$$
\begin{aligned}
&nand : bool \times bool \rightharpoonup bool \\
&nand = distrib \ggg (not \oplus (erase_{bool} \ggg create_{true})) \\
&\qquad \ggg factor \ggg (erase_{bool} \otimes id) \ggg arr\ unite
\end{aligned}
$$

This is our password checker!

Physical Perspective on Computation

- Computer science abstractions are based on old physics
- Computer applications are more and more "physical"
- Physical principles such as conservation of information should be part of our foundational abstractions
- Other principles?

Conclusions

Collaborators and References