ÖAW AUSTRIAN ACADEMY OF SCIENCES

IQOQI – INSTITUTE FOR QUANTUM OPTICS AND QUANTUM INFORMATION VIENNA

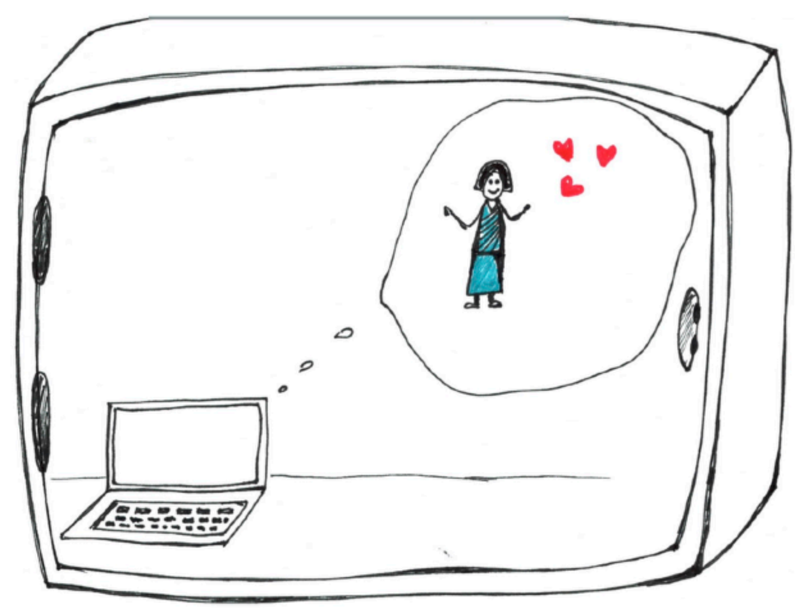
Computational irreducibility

and notions of simulation for Turing machines

Marius Krumm and Markus P. Müller

Institute for Quantum Optics and Quantum Information (IQOQI), Vienna

Perimeter Institute for Theoretical Physics (PI), Waterloo, Canada



FQXi FOUNDATIONAL QUESTIONS INSTITUTE

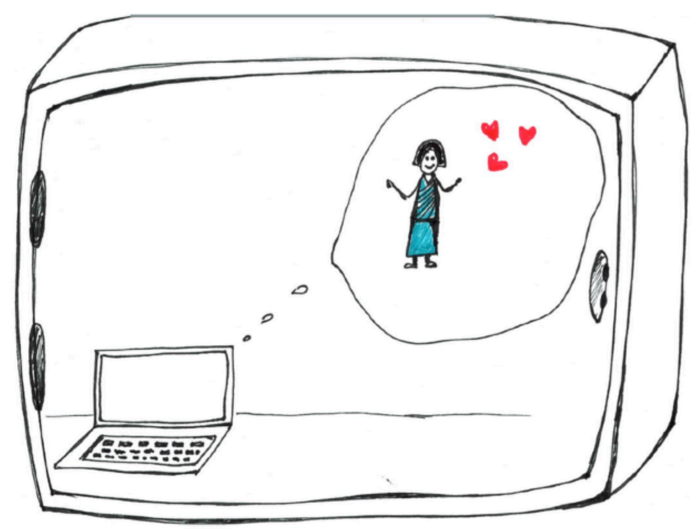
PERIMETER INSTITUTE

Overview

1. **Wolfram's** computational irreducibility and free will

inadequacy of that approach

2. John the cook and **computational sourcehood**



motivates the main technical question

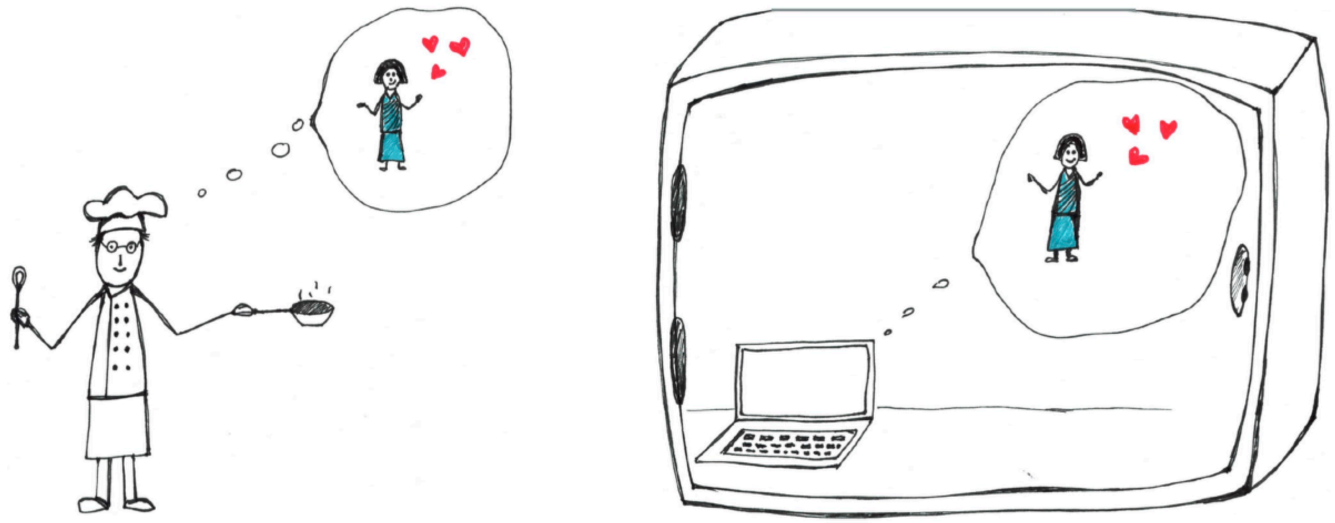
3. Universality and a **simulation preorder for TMs**?

Overview

1. **Wolfram's** computational irreducibility and free will

inadequacy of that approach

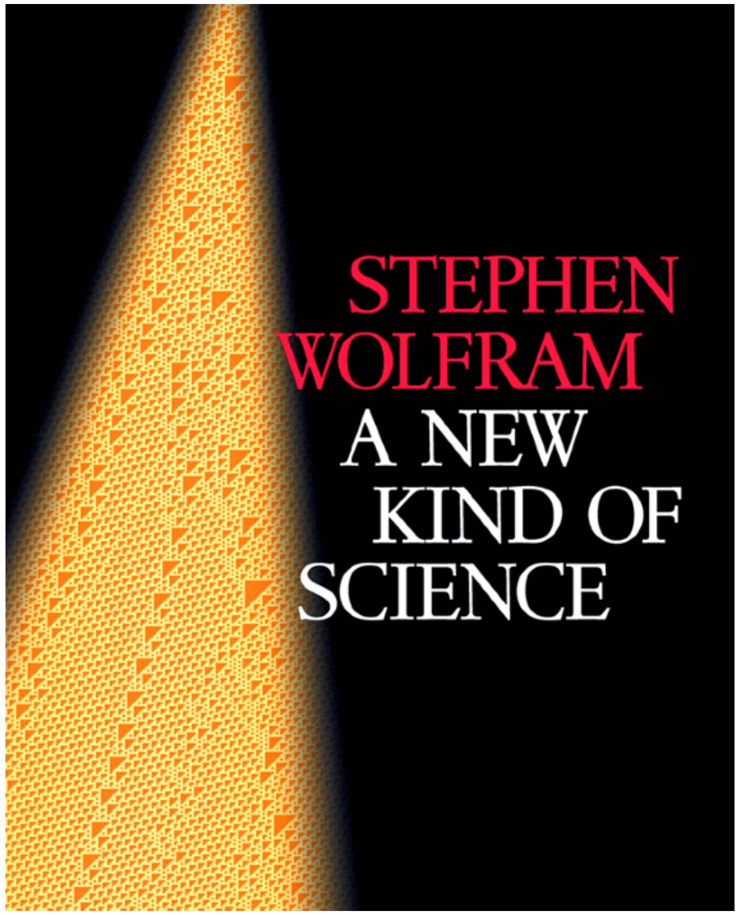
2. John the cook and **computational sourcehood**



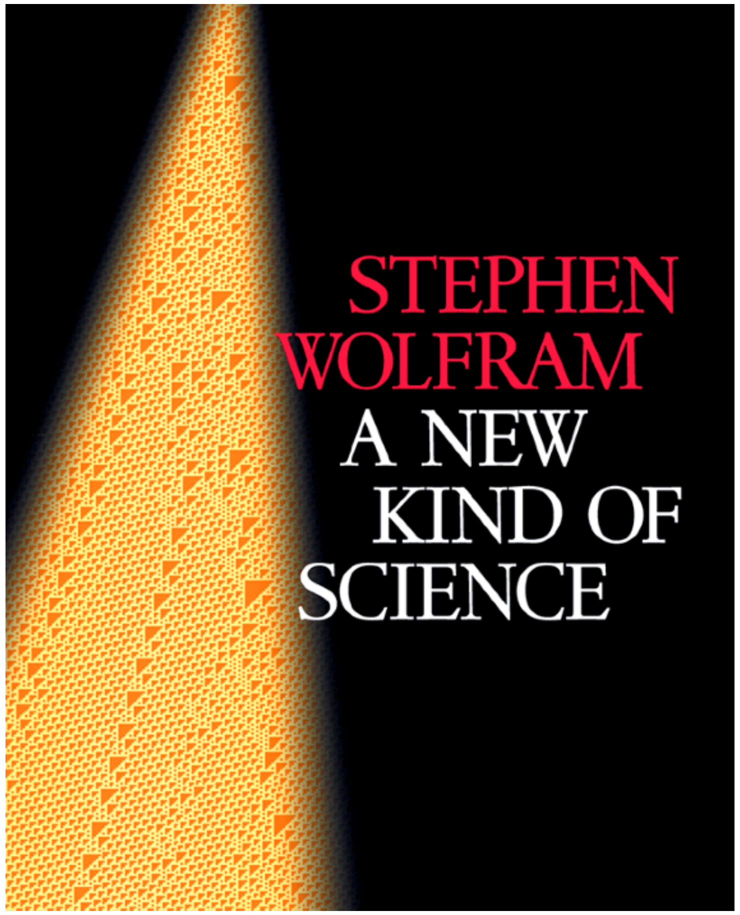
motivates the main technical question

3. Universality and a **simulation preorder for TMs**?

Wolfram’s computational irreducibility

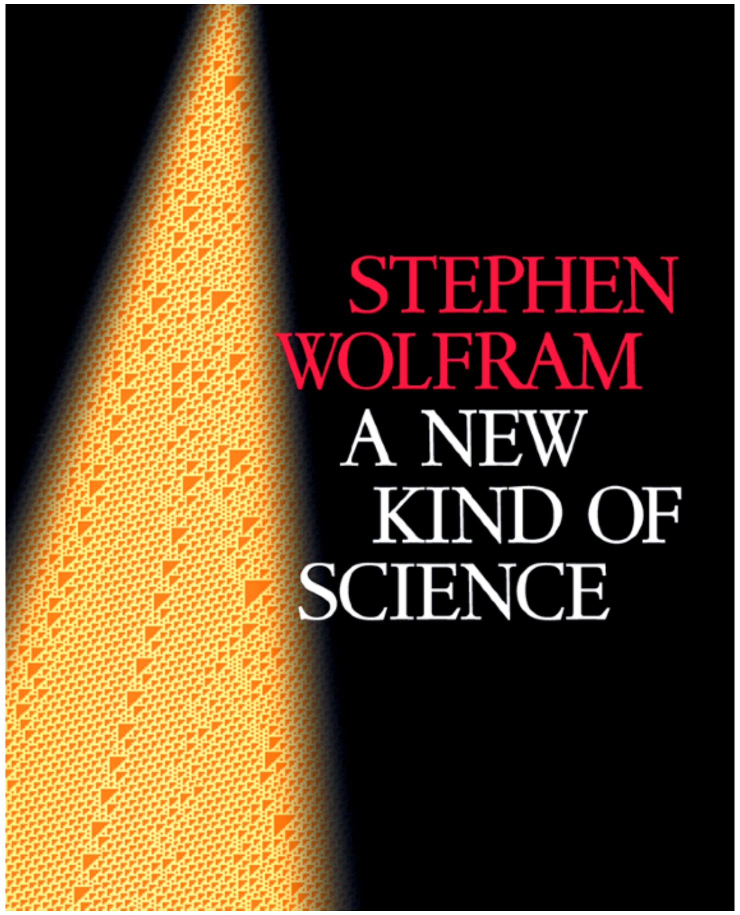


Wolfram's computational irreducibility



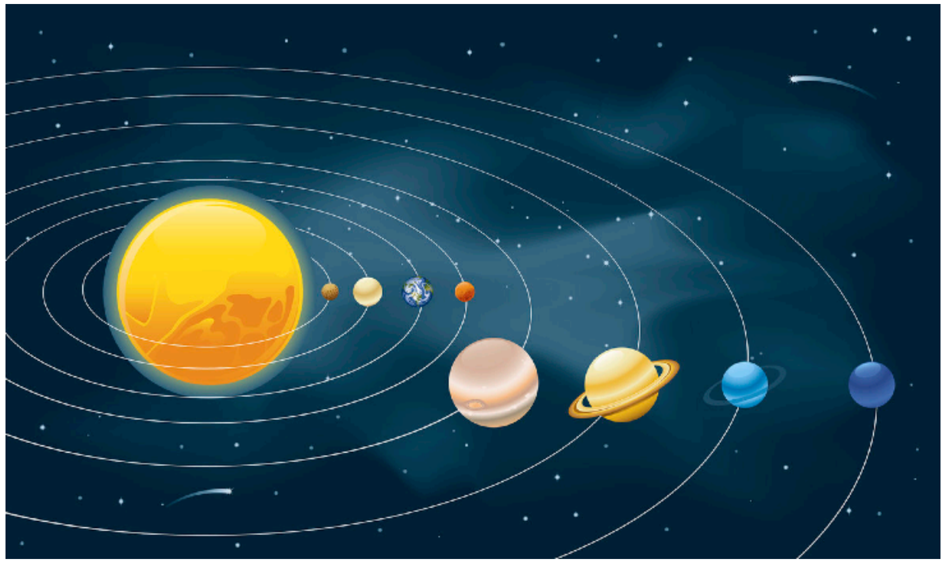
Some systems in nature are **comp. reducible:** we can predict their future behavior with simple equations / theories / algorithms.

Wolfram's computational irreducibility

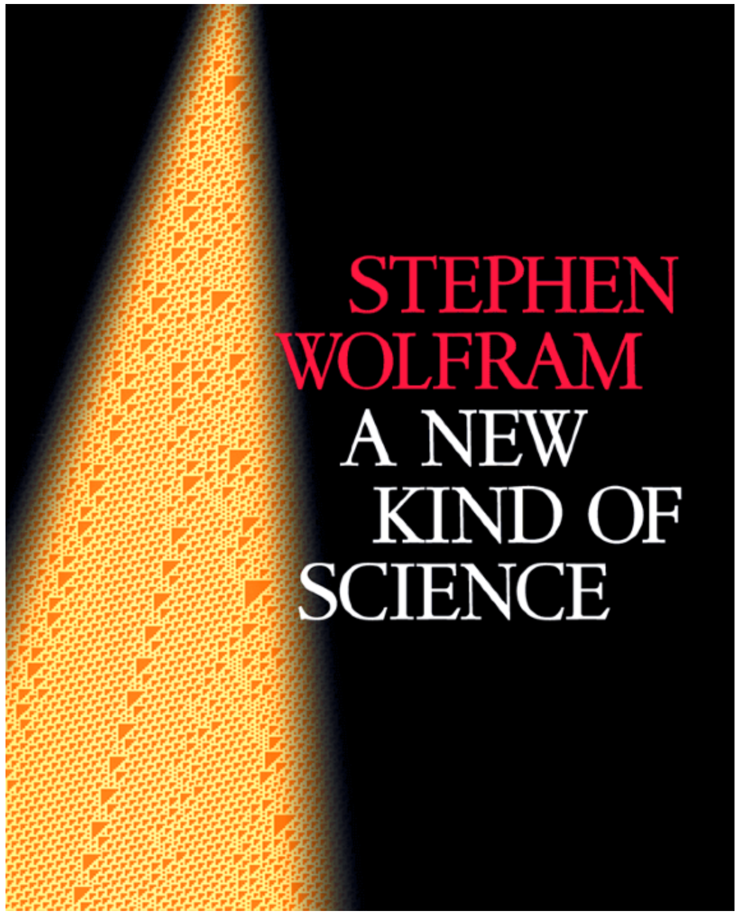


Some systems in nature are **comp. reducible:** we can predict their future behavior with simple equations / theories / algorithms.

Examples:
Position of **Jupiter** on Jan. 31, 2520;

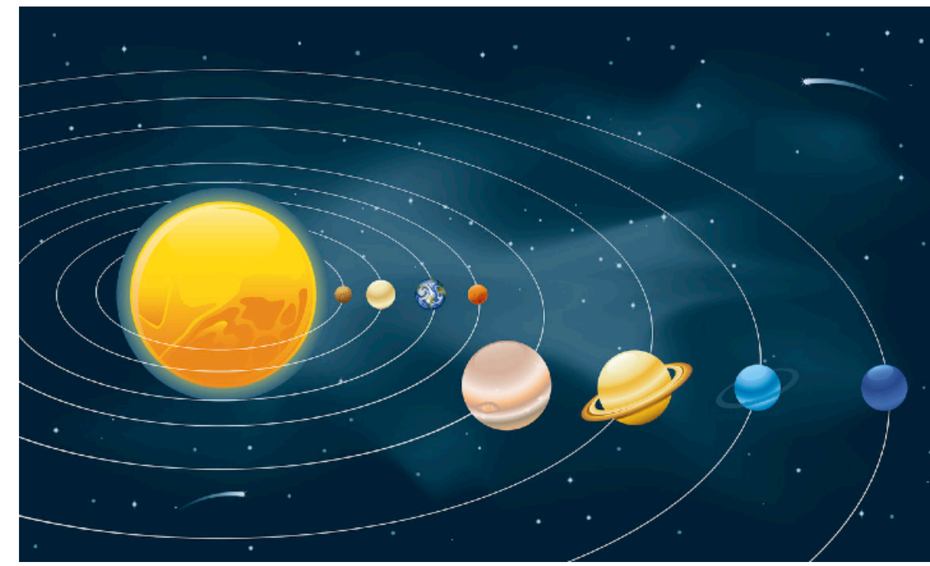


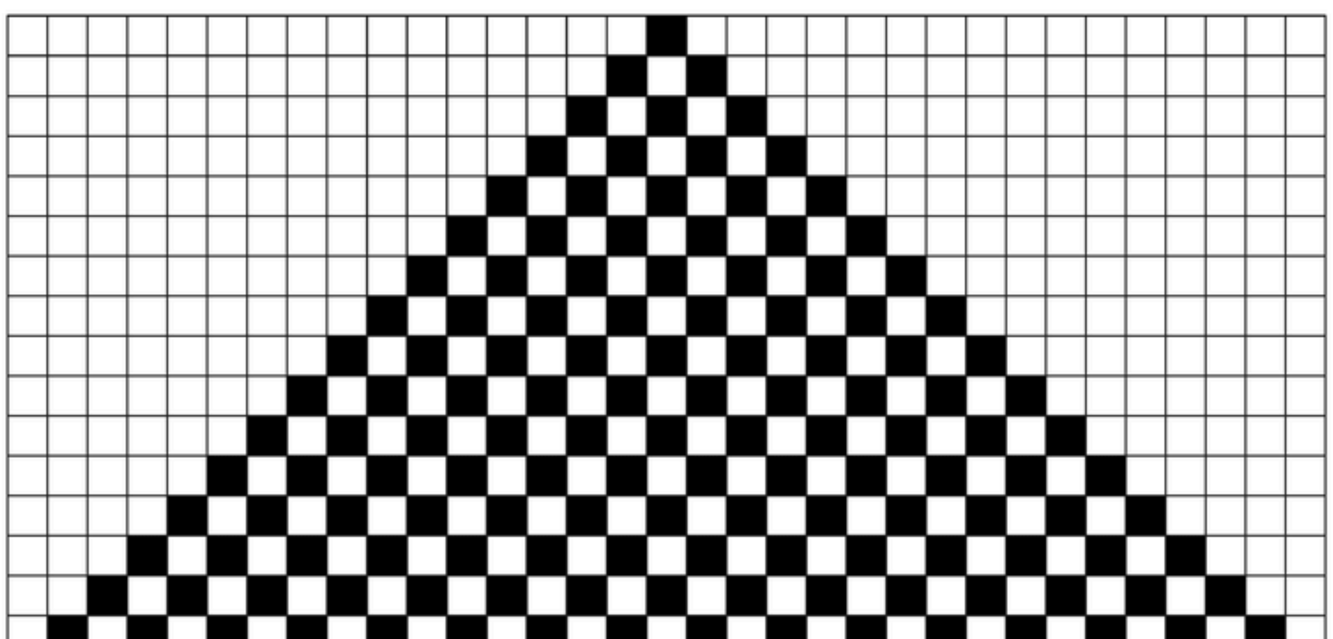
Wolfram's computational irreducibility

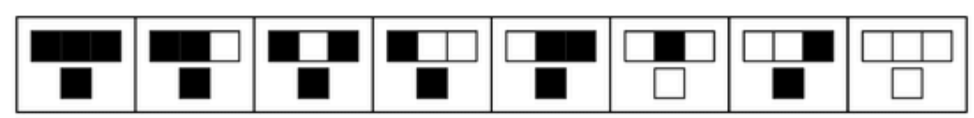


Some systems in nature are **comp. reducible:** we can predict their future behavior with simple equations / theories / algorithms.

Examples:
Position of **Jupiter** on Jan. 31, 2520;
some **cellular automata**.



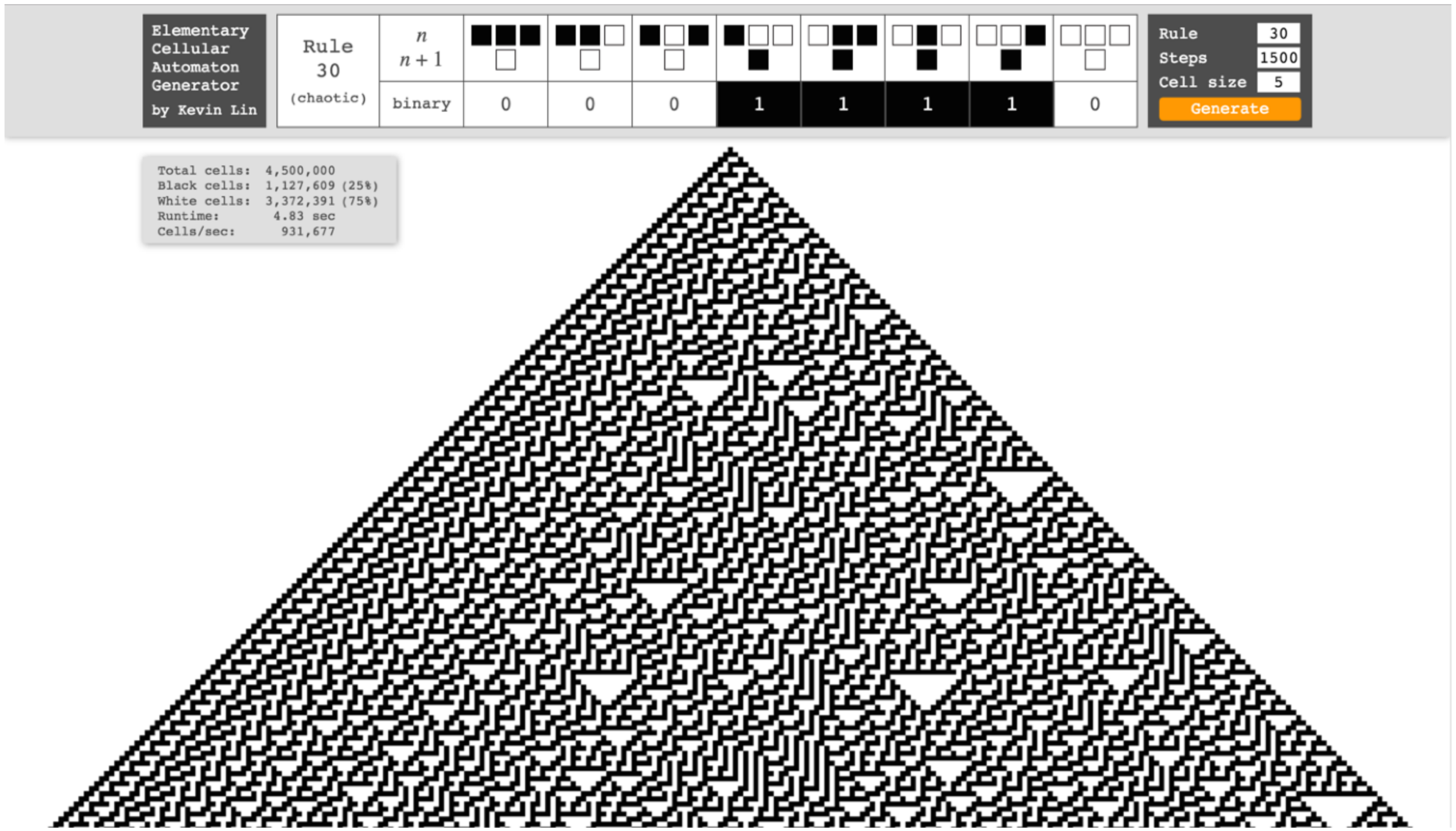




A cellular automaton with a slightly different rule. The rule makes a particular cell black if either of its neighbors was black on the step before, and makes the cell white if both its neighbors were white. Starting from a single black cell, this rule leads to a checkerboard pattern. In the numbering scheme of Chapter 3, this is cellular automaton rule 250.

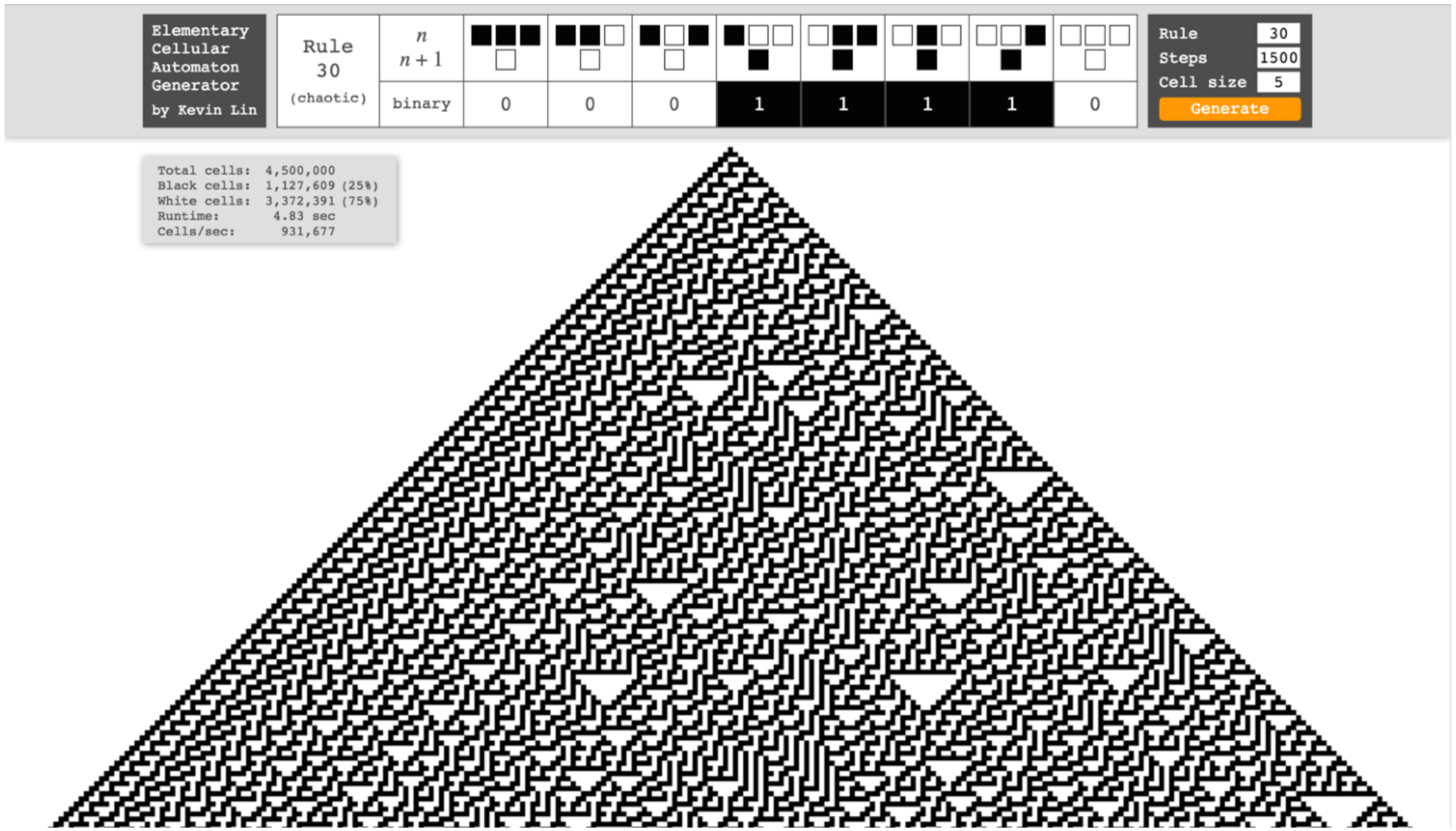
Wolfram’s computational irreducibility

Some systems admit no such shortcuts: **computationally irreducible**.



Wolfram's computational irreducibility

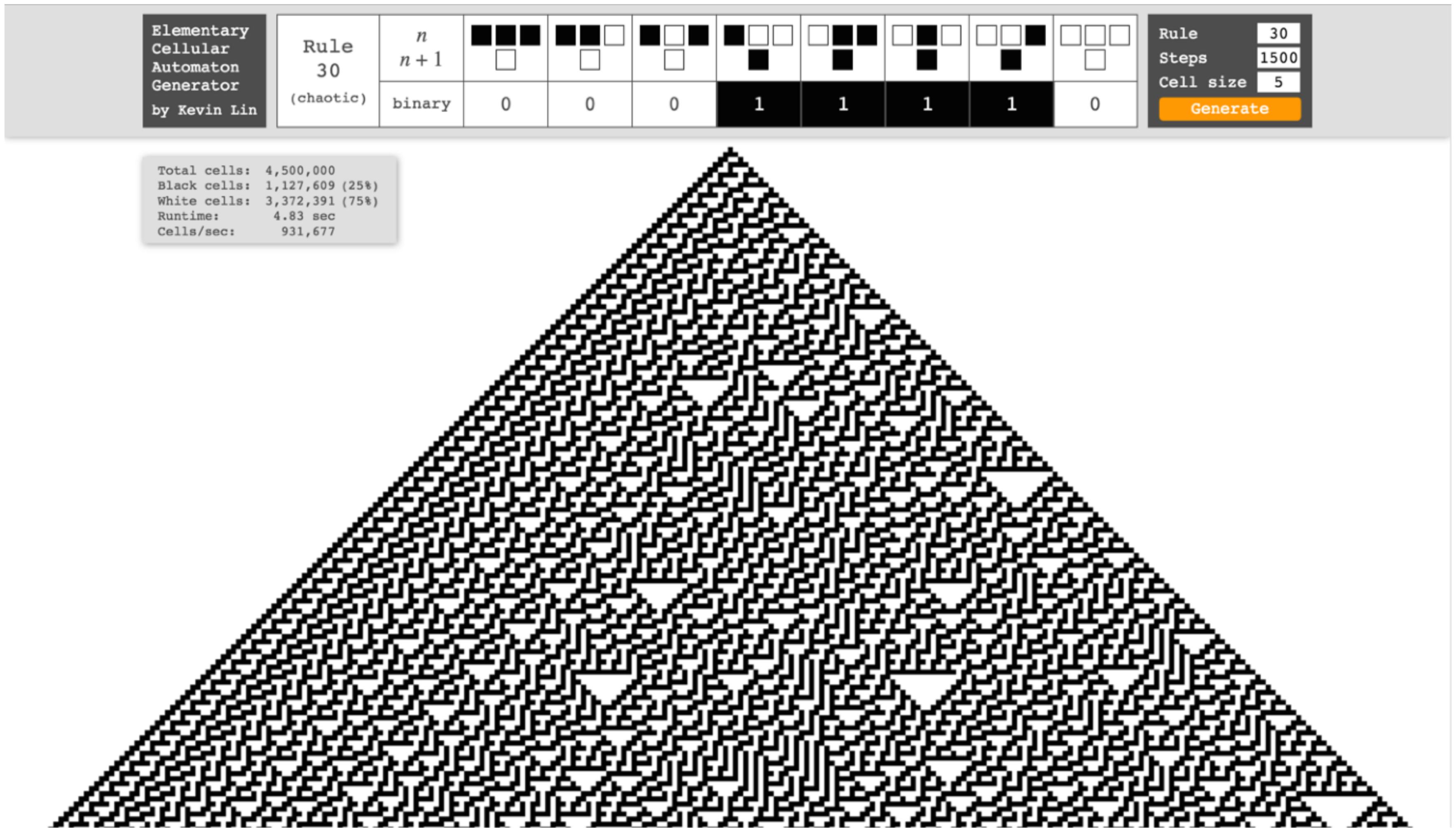
Some systems admit no such shortcuts: **computationally irreducible**.



- To predict the behavior of a CI system, we have to **emulate it exactly.**
- Happens as soon as **computational universality** is reached.
- Wolfram's claim: except for the simplest systems, this is the **typical behavior**.

Wolfram’s computational irreducibility

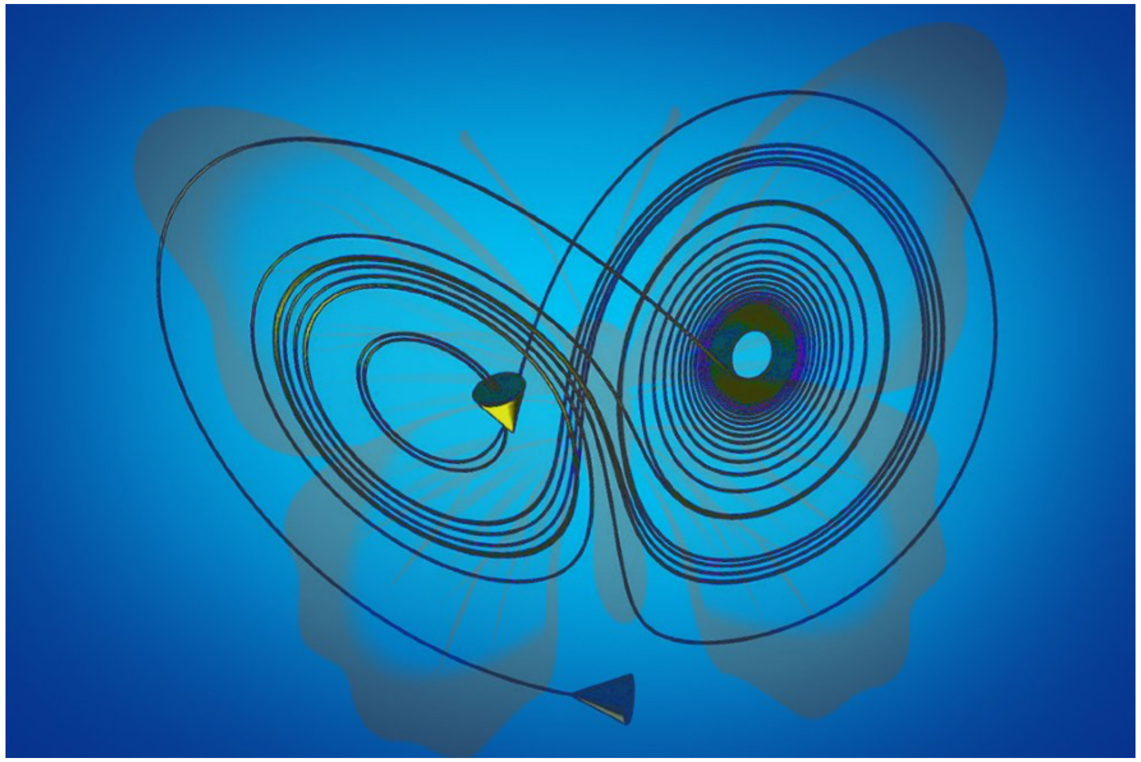
Some systems admit no such shortcuts: **computationally irreducible**.



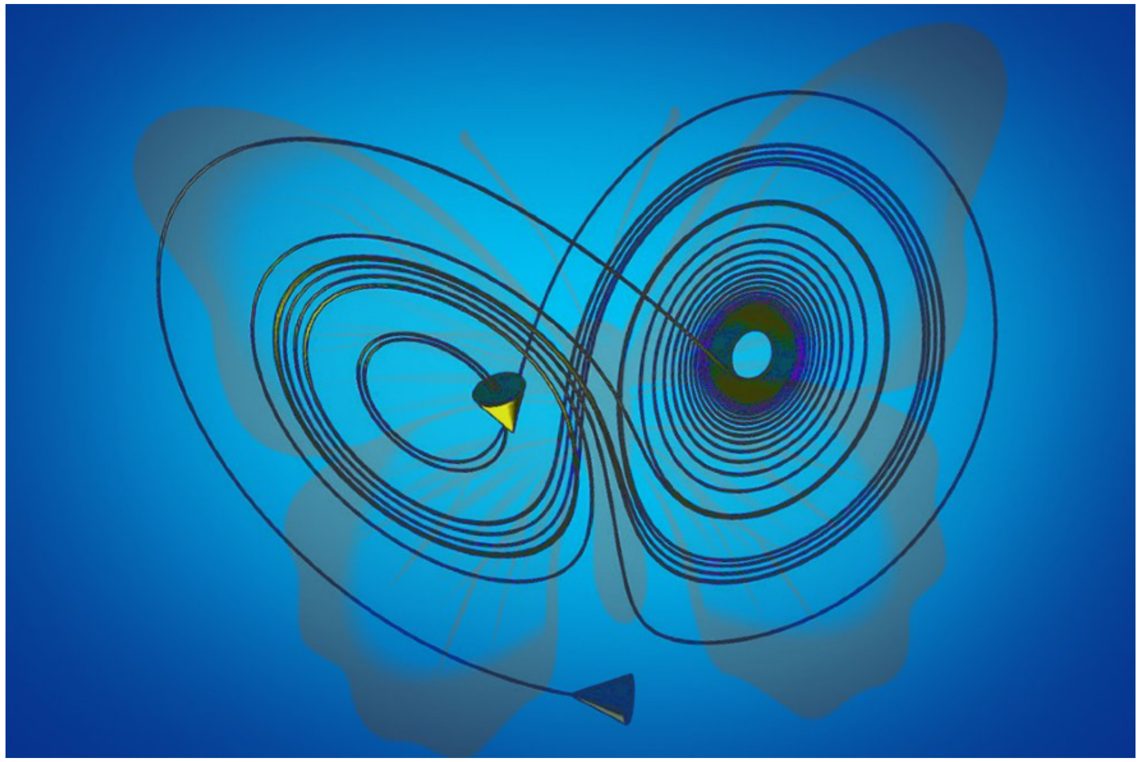
- To predict the behavior of a CI system, we have to **emulate it exactly.**
- Happens as soon as **computational universality** is reached.
- Wolfram’s claim: except for the simplest systems, this is the **typical behavior**.

Problem: there is **no formal definition** of computational irreducibility!

Computational irreducibility ≠ chaos



Computational irreducibility ≠ chaos



What is the relation of your theory with chaos and complexity theory. When I try to explain what you discover in your book to someone else they say, "Ah, chaos theory."

Chaos theory is really about a very specific phenomenon: that **sensitive dependence on initial conditions** can lead to randomness. And what one finds in the end is that the only way to get randomness out of this phenomenon is just to put randomness in, in the initial conditions. What I've found is that simple programs can actually produce randomness—and complexity—without it ever being put it. It's a much more powerful phenomenon.

Computational irreducibility and **free will**?

Computational irreducibility and **free will**?

Wolfram (2002):

“And it is this, I believe, that is the ultimate origin of the **apparent freedom** *of human will. For even though all the components of our brains presumably follow definite laws, I strongly suspect that their overall behavior corresponds to an irreducible computation whose outcome can never in effect be found by reasonable laws.”*

Computational irreducibility and **free will**?

Wolfram (2002):

"And it is this, I believe, that is the ultimate origin of the **apparent freedom** *of human will. For even though all the components of our brains presumably follow definite laws, I strongly suspect that their overall behavior corresponds to an irreducible computation whose outcome can never in effect be found by reasonable laws."*

S. Bringsjord, Free will and a new kind of science (2013):

"If someone's will is apparently free, it hardly follows that that will is in fact free. Nowhere in ANKS [his book] does Wolfram even intimate that he maintains that our decisions are in fact free."

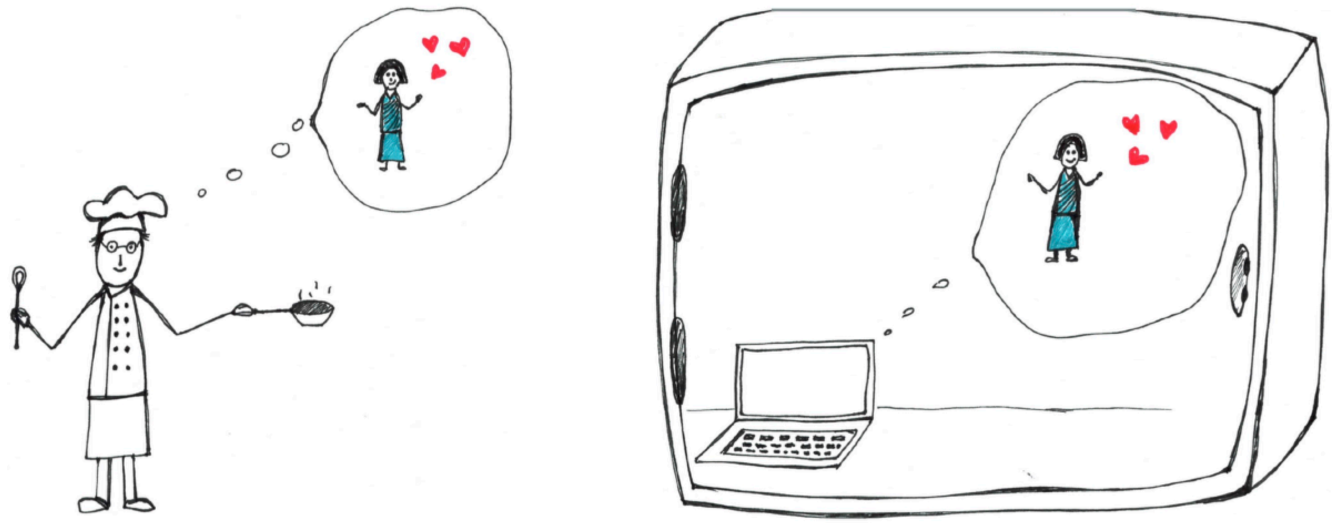
➡ Wolfram is *"epistemologically correct"*, but *"metaphysically wrong"*.

Overview

1. **Wolfram's** computational irreducibility and free will

inadequacy of that approach

2. John the cook and **computational sourcehood**



motivates the main technical question

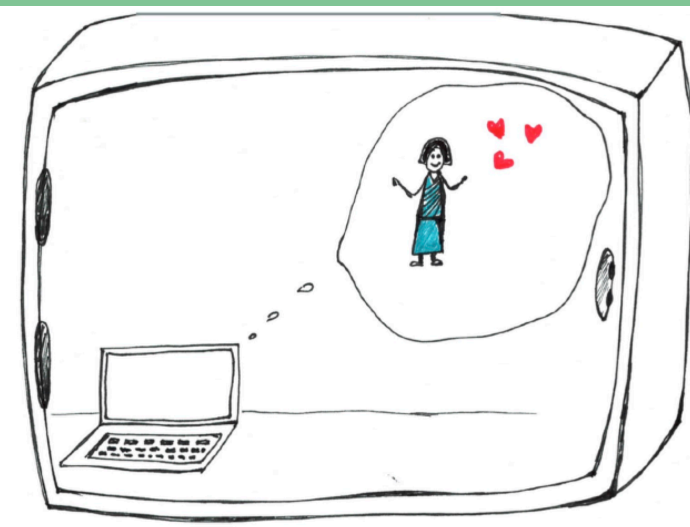
3. Universality and a **simulation preorder for TMs**?

Overview

1. **Wolfram's** computational irreducibility and free will

inadequacy of that approach

2. John the cook and **computational sourcehood**



motivates the main technical question

3. Universality and a **simulation preorder for TMs**?

Free will, compatibilism, and sourcehood

Philosophers argue for (one of) the following underpinnings of free will:

- **The freedom to do otherwise.**
  But what does that exactly mean?

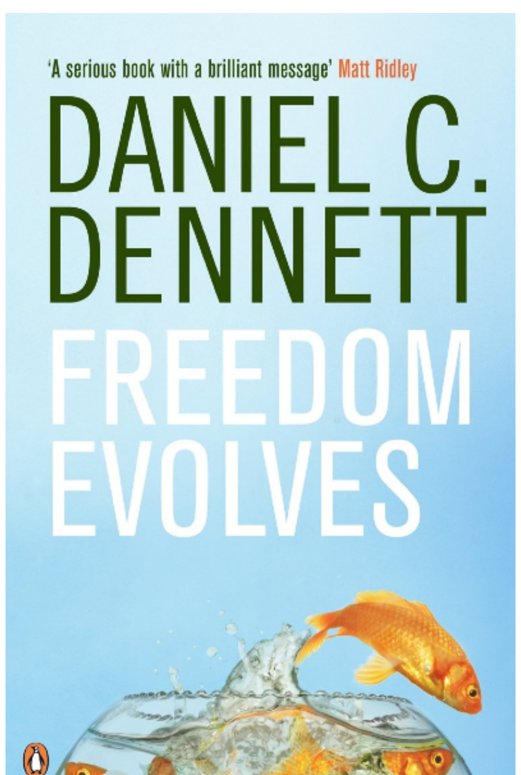
- **Sourcehood.**
  What matters for an agent's freedom and responsibility
  is the source of her action—how her action was brought about.

Free will, compatibilism, and sourcehood

Philosophers argue for (one of) the following underpinnings of free will:

- ~~**The freedom to do otherwise.**
  But what does that exactly mean?~~

- **Sourcehood.**
  What matters for an agent's freedom and responsibility
  is the source of her action—how her action was brought about.



Like the **compatibilists**, I will focus on **sourcehood**:
it is a notion of free will that is compatible even with
a fully deterministic and digital world.
However, I will argue that computational irreducibility
is **not** the correct notion to study.

John the (emotional) cook

Worst-case assumptions: fully deterministic and digital world.

- Every morning, John prepares one of *N* breakfasts, where *N* is large.
- E.g., he thinks of his late Canadian wife, and then prepares omelette with Maple syrup.

John the (emotional) cook

Worst-case assumptions: fully deterministic and digital world.

- Every morning, John prepares one of *N* breakfasts, where *N* is large.
- E.g., he thinks of his late Canadian wife, and then prepares omelette with Maple syrup.

Let's try to convince John that he doesn't have free will: scan him + apartment in the evening → computer.

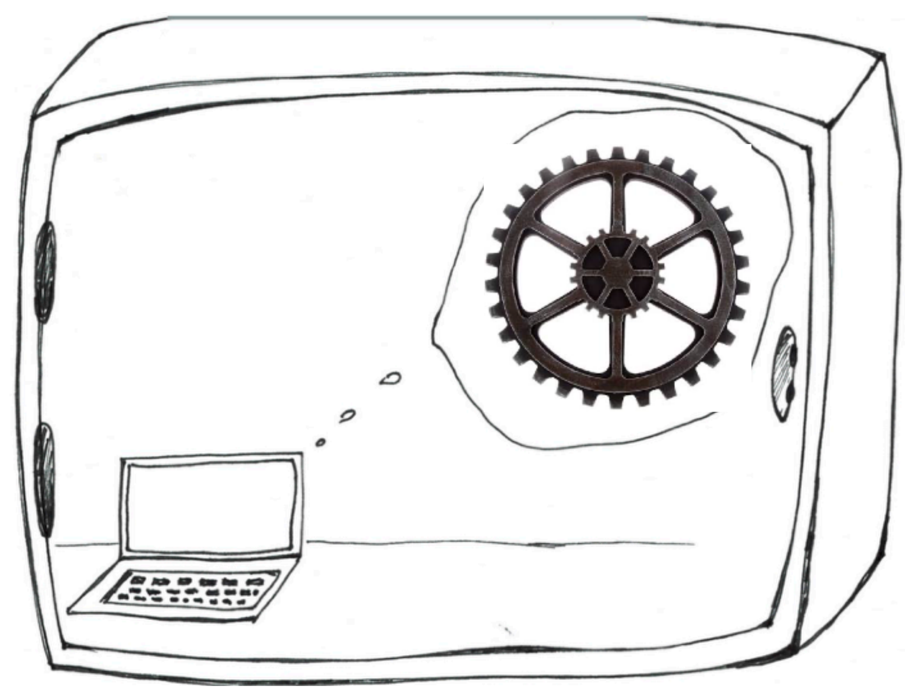
John the (emotional) cook

Worst-case assumptions: fully deterministic and digital world.

- Every morning, John prepares one of *N* breakfasts, where *N* is large.
- E.g., he thinks of his late Canadian wife, and then prepares omelette with Maple syrup.

Let's try to convince John that he doesn't have free will: scan him + apartment in the evening → computer.

Computation time T might be small enough to finish before breakfast, or it might finish later. → Put computer in a secure **safe**. Confront John with result **after** the breakfast.

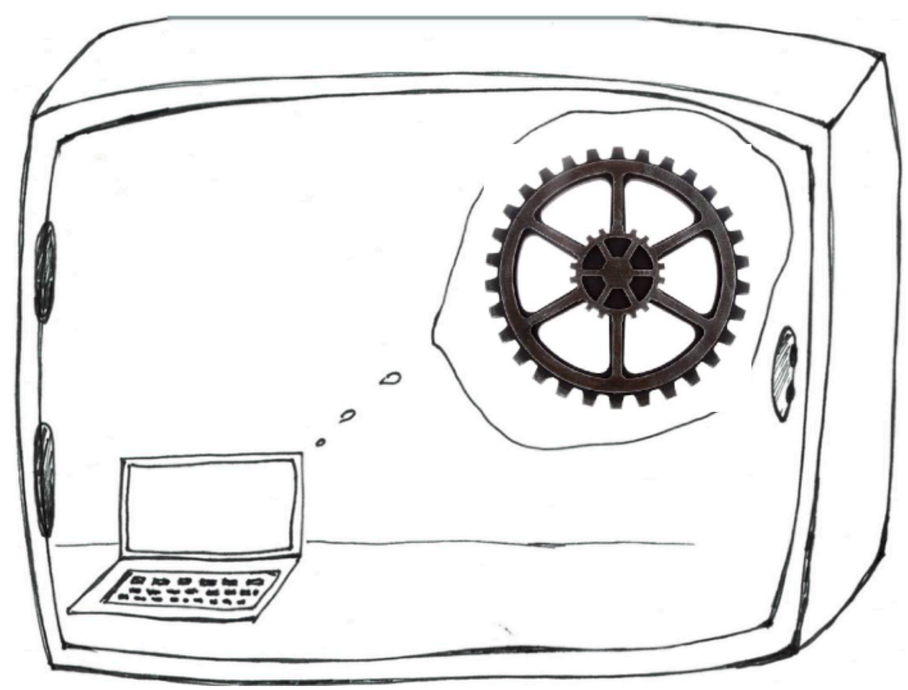


John the (emotional) cook

- If T is small enough (shortcut):

See, John? You have decided to prepare the Canadian omelette. Ha, this is exactly what our computer has predicted half an hour earlier, as several witnesses can testify — well before you have thought about your Canadian wife!"

Computation time T might be small enough to finish before breakfast, or it might finish later. ➡ Put computer in a secure **safe**. Confront John with result **after** the breakfast.



John the (emotional) cook

- If T is small enough (shortcut):

See, John? You have decided to prepare the Canadian omelette. Ha, this is exactly what our computer has predicted half an hour earlier, as several witnesses can testify — well before you have thought about your Canadian wife!”

- If T is large (no shortcut):

See, John? You think your emotions have decided to prepare the Canadian omelette, but what happened in the safe was only determined by your physical state and apartment yesterday night. Your thoughts of your late wife this morning had no impact on the decision whatsoever!”

John the (emotional) cook

- If T is small enough (shortcut):

See, John? You have decided to prepare the Canadian omelette. Ha, this is exactly what our computer has predicted half an hour earlier, as several witnesses can testify — well before you have thought about your Canadian wife!"

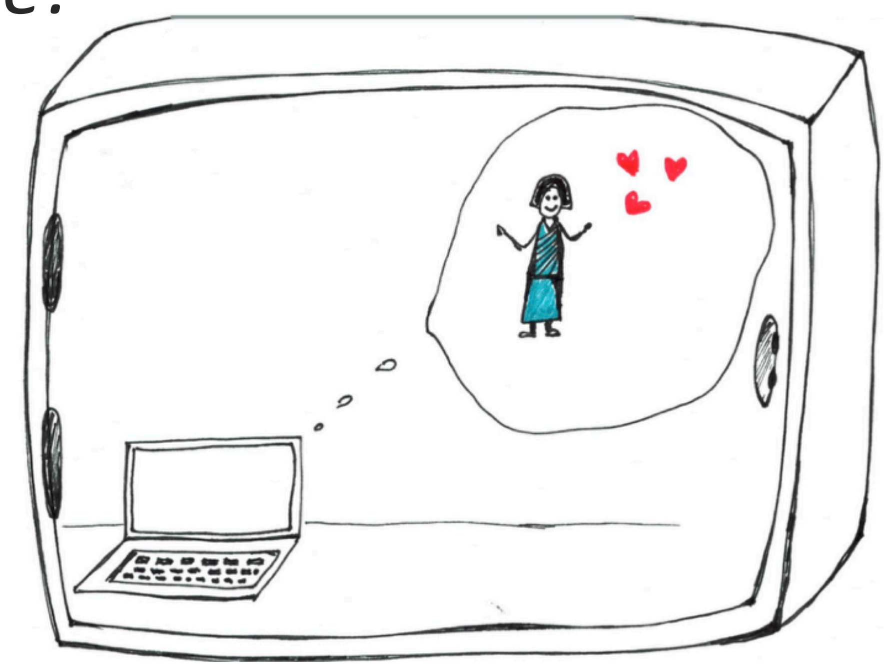
- If T is large (no shortcut):

See, John? You think your emotions have decided to prepare the Canadian omelette, but what happened in the safe was only determined by your physical state and apartment yesterday night. Your thoughts of your late wife this morning had no impact on the decision whatsoever!"

In **both** cases, sourcefulness of John's emotions is equally contested. **"Shortcut or not" is an irrelevant question.** Insofar as computational irreducibility is understood as "no shortcut", it is not the relevant notion.

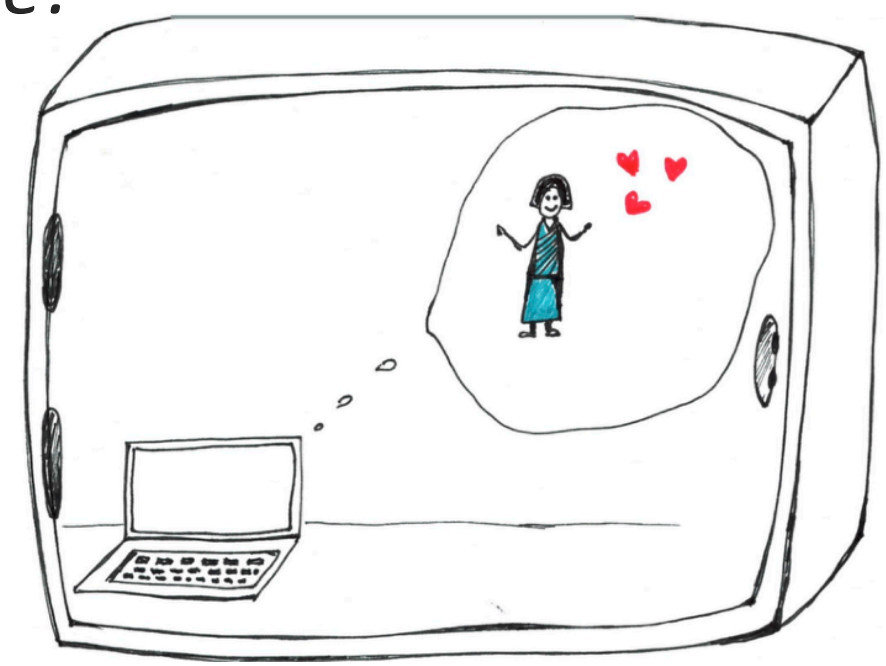
John argues back

But look at your computer: it has performed an exact simulation of my thought processes. In particular, there have been **algorithmic correlates** *of my emotions in the safe!*



John argues back

But look at your computer: it has performed an exact simulation of my thought processes. In particular, there have been **algorithmic correlates** *of my emotions in the safe!*

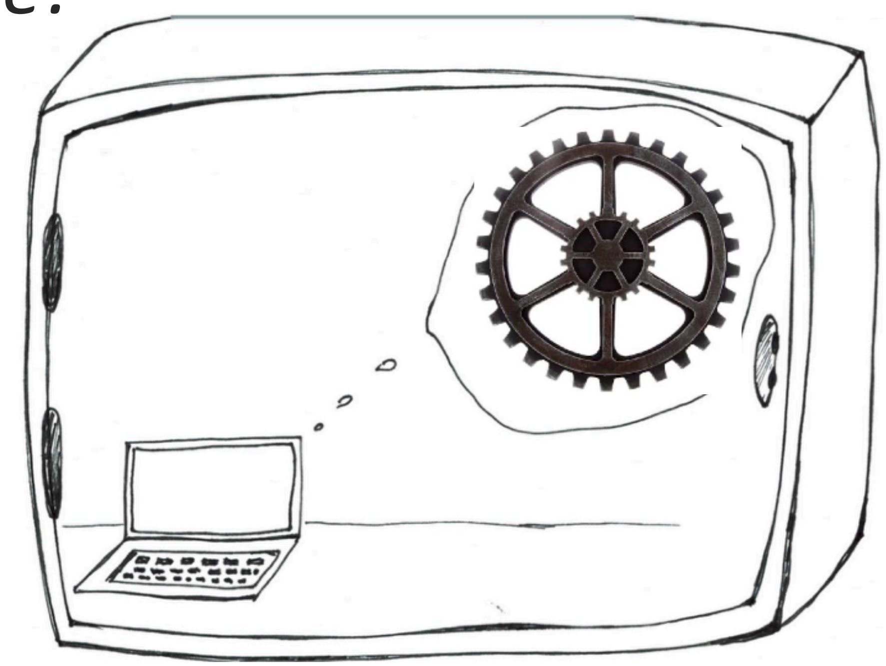


I am not that material body, but the computational process that is represented by it. You have just manufactured another representation of the same process in the safe.

Hence, it was me who has made the decision!

John argues back

But look at your computer: it has performed an exact simulation of my thought processes. In particular, there have been **algorithmic correlates** *of my emotions in the safe!*

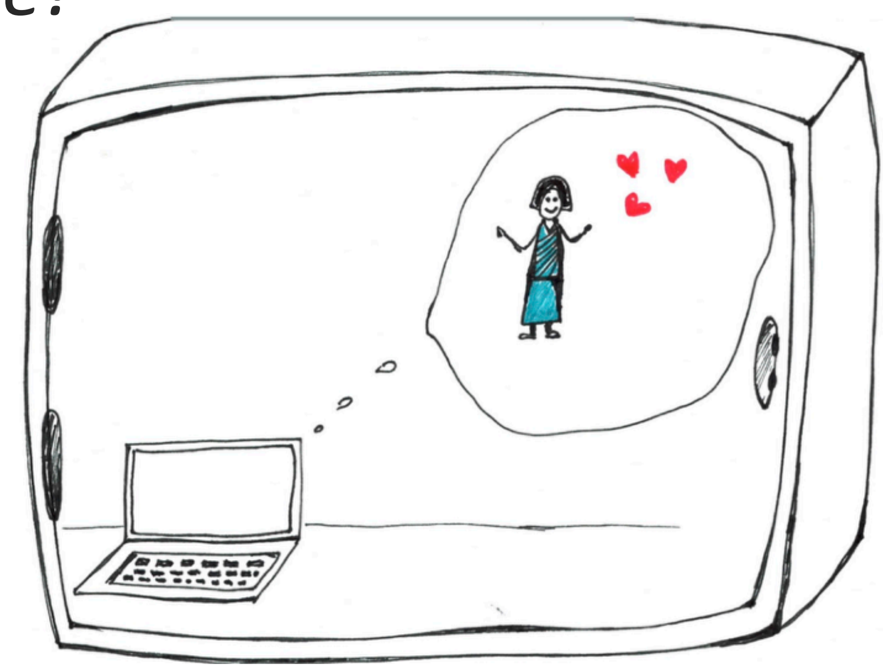


I am not that material body, but the computational process that is represented by it. You have just manufactured another representation of the same process in the safe. **Hence, it was me who has made the decision!**

Clearly, John *would* be very worried if the computation inside the safe had *nothing to do* with the processes in his brain.

John argues back

But look at your computer: it has performed an exact simulation of my thought processes. In particular, there have been **algorithmic correlates** *of my emotions in the safe!*



I am not that material body, but the computational process that is represented by it. You have just manufactured another representation of the same process in the safe.
Hence, it was me who has made the decision!

Clearly, John *would* be very worried if the computation inside the safe had *nothing to do* with the processes in his brain.

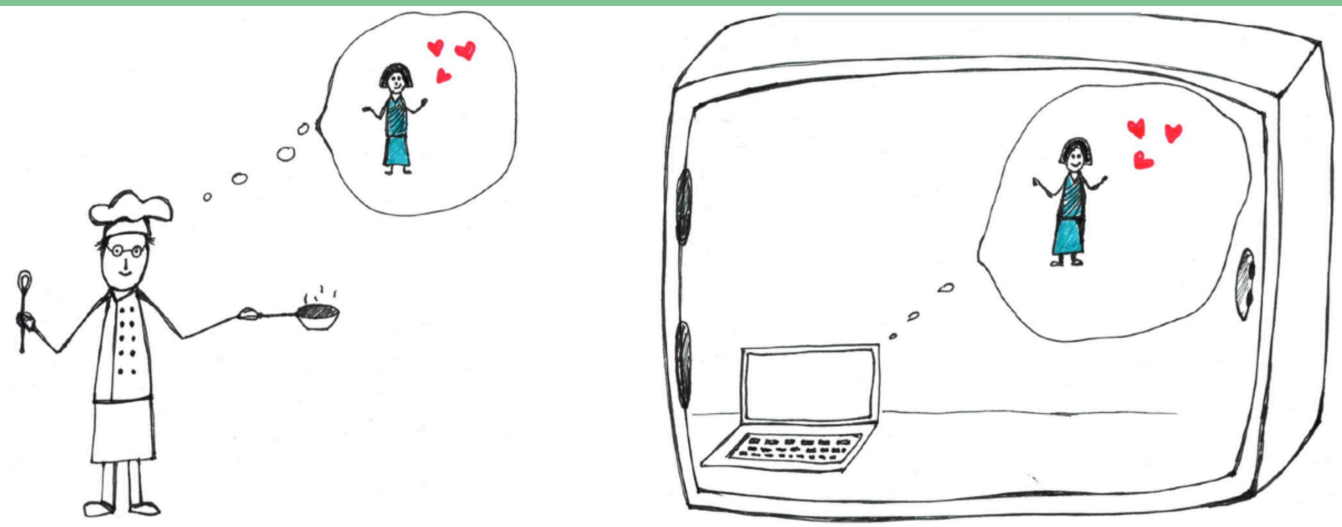
Computational sourcehood: To predict John's decision, the simulation has to contain representations of all of John's instantaneous states.

Overview

1. **Wolfram's** computational irreducibility and free will

inadequacy of that approach

2. John the cook and **computational sourcehood**



motivates the main technical question

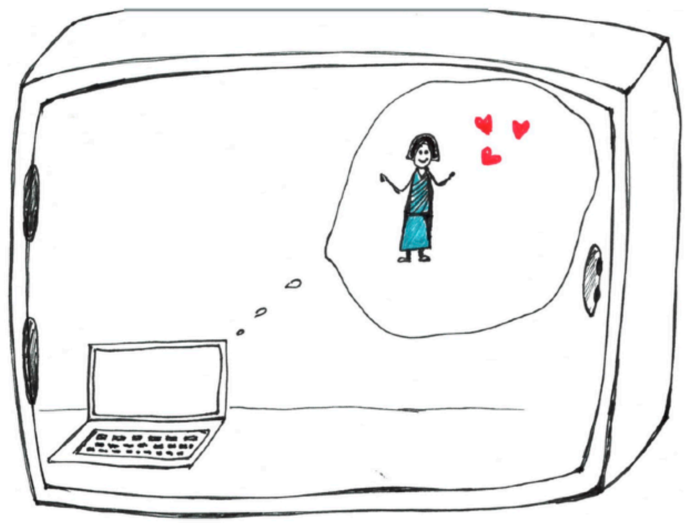
3. Universality and a **simulation preorder for TMs**?

Overview

1. **Wolfram's** computational irreducibility and free will

inadequacy of that approach

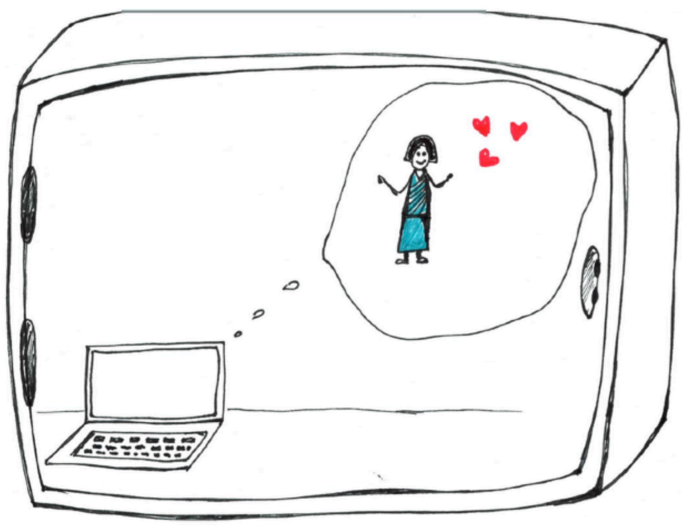
2. John the cook and **computational sourcehood**



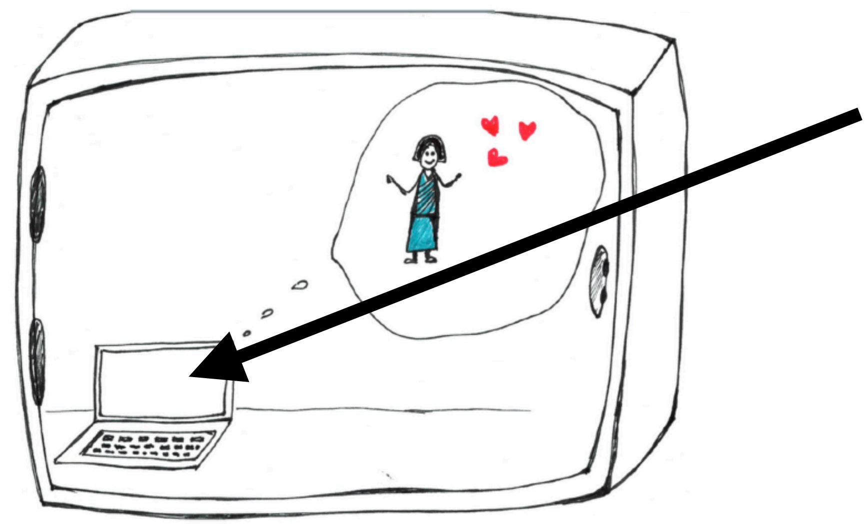
motivates the main technical question

3. Universality and a **simulation preorder for TMs**?

Computational sourcehood and Turing machines



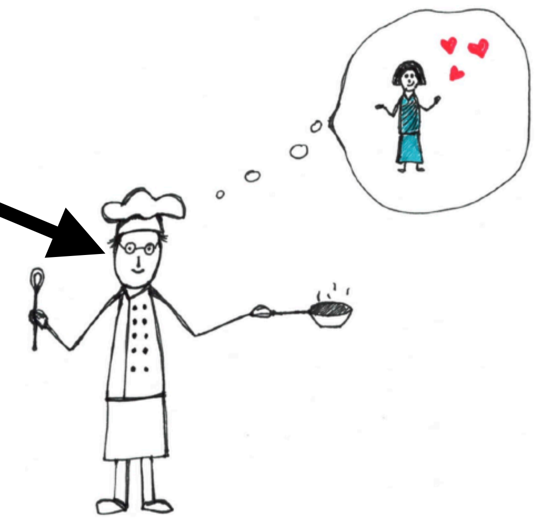
Computational sourcehood and Turing machines

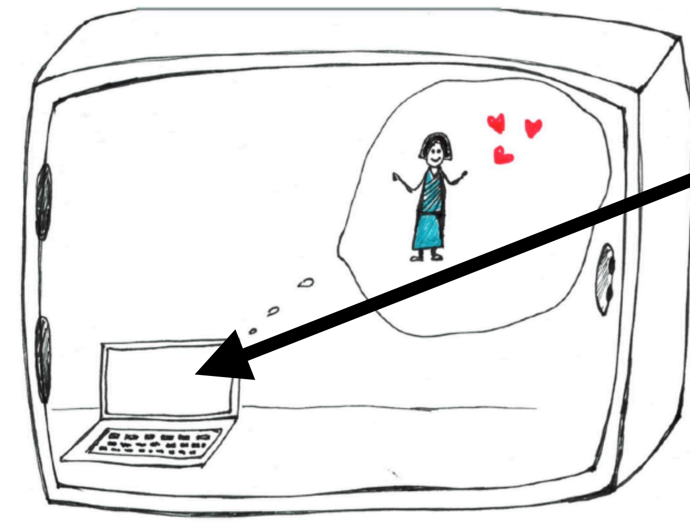


universal
TM *U*

Computational sourcehood and Turing machines

TM *T*

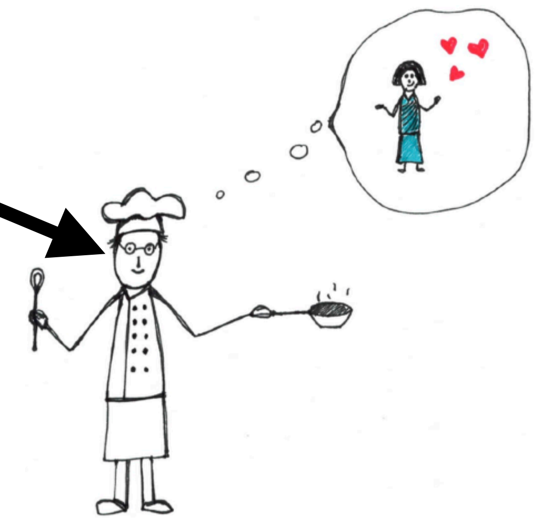


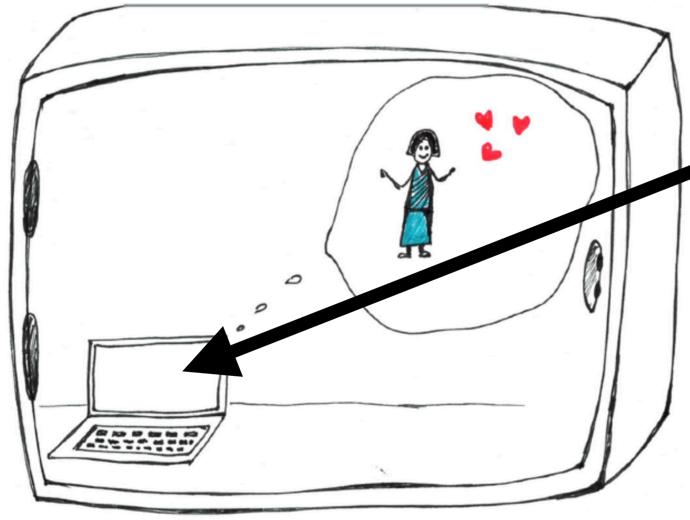


universal
TM *U*

Computational sourcehood and Turing machines

TM T





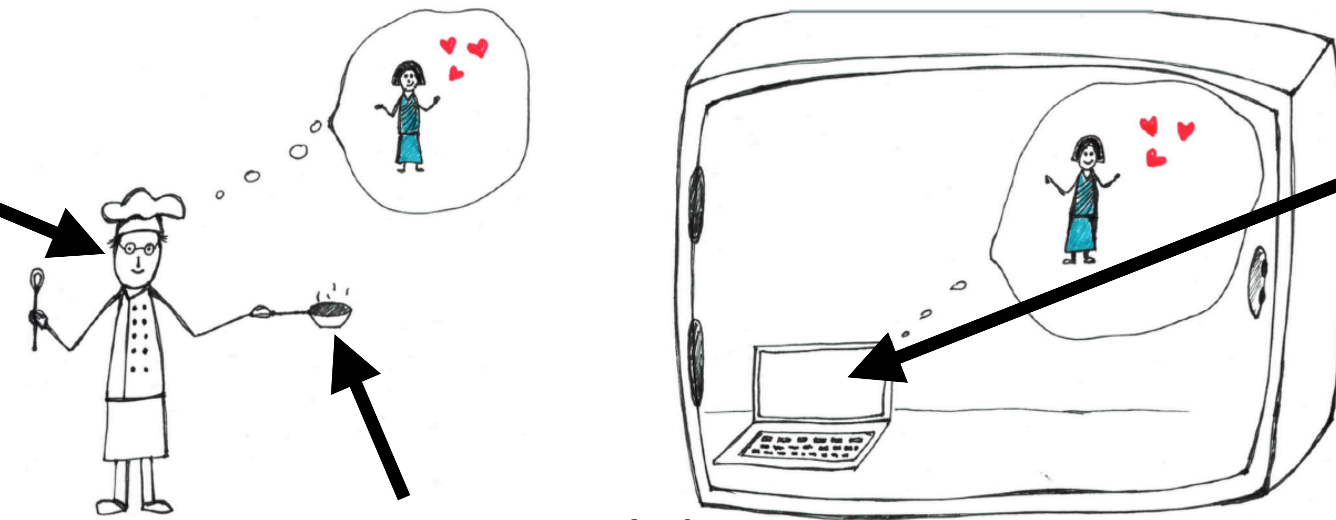
universal TM U

- x: description of John's (+apartment's) state the evening before

$$x \in \{0,1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$$

Computational sourcehood and Turing machines

TM *T*



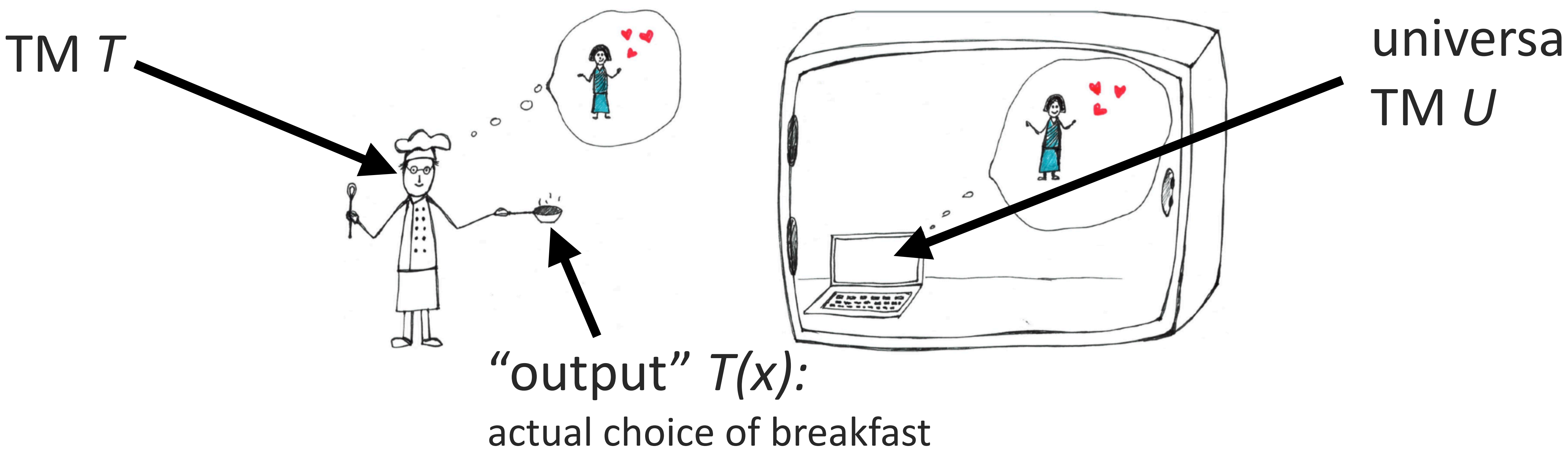
universal TM *U*

“output” *T(x)*:
actual choice of breakfast

- *x*: description of John’s (+apartment’s) state the evening before

$$x \in \{0,1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$$

Computational sourcehood and Turing machines

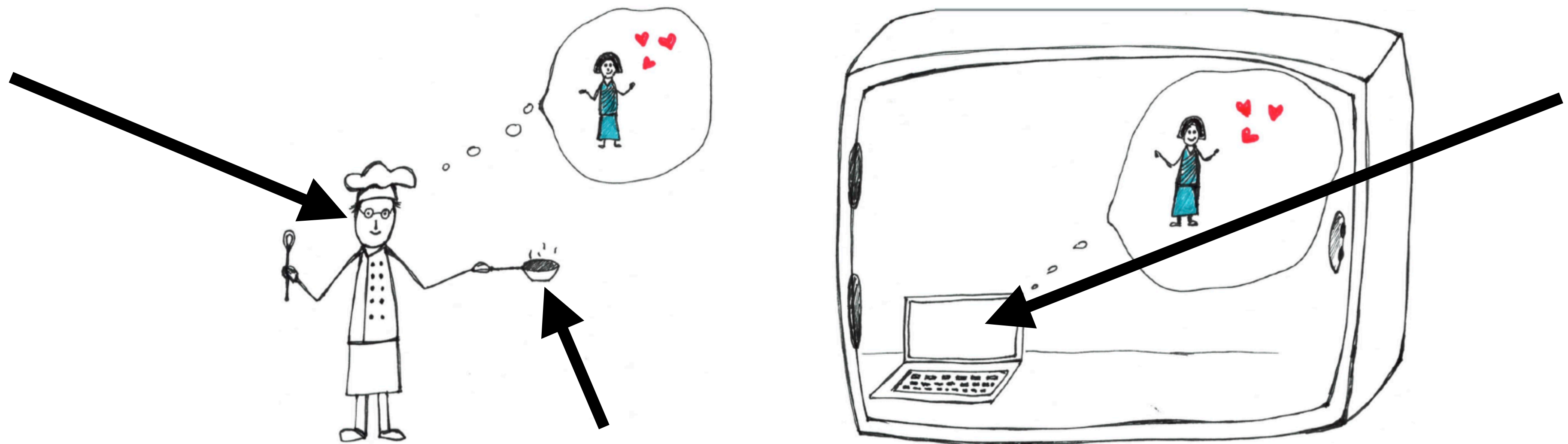


- x: description of John’s (+apartment’s) state the evening before

$$x \in \{0,1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$$

- p_T: description of T (i.e. of “John as a process”)

Computational sourcehood and Turing machines



- x: description of John's (+apartment's) state the evening before
  $x \in \{0,1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$
- p_T: description of T (i.e. of "John as a process")

$$U(p_T x) = T(x)$$

Same outputs, but

does that mean that U must simulate T step by step?

A conjecture on universal Turing machines

Conjecture. Suppose that a TM *U* is **universal** in the sense that it reproduces the **outputs** of any other TM: that is,

$$U(p_T x) = T(x)$$

for every TM *T* and every input *x* on which *T* halts. Then, for "most" *T*, the universal TM *U* will generate its output by means of some form of **step-by-step simulation** of *T*'s computation.

In this sense, *T* is the "source" of its outputs (**computational sourcehood**).

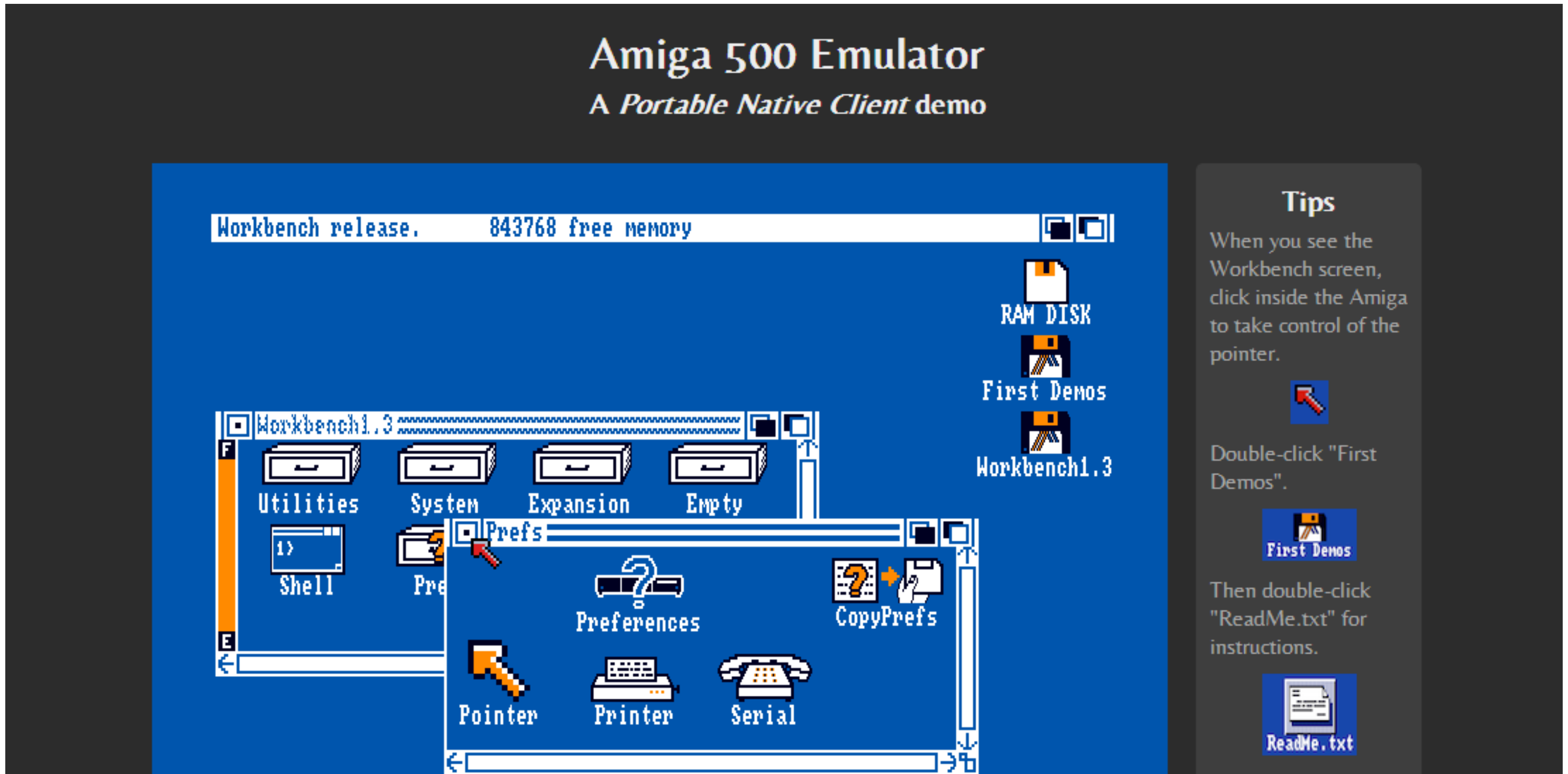
A conjecture on universal Turing machines

Conjecture. Suppose that a TM U is **universal** in the sense that it reproduces the **outputs** of any other TM: that is,

$$U(p_T x) = T(x)$$

for every TM T and every input x on which T halts. Then, for "most" T, the universal TM U will generate its output by means of some form of **step-by-step simulation** of T's computation.
In this sense, T is the "source" of its outputs (**computational sourcehood**).

Standard emulators do this!



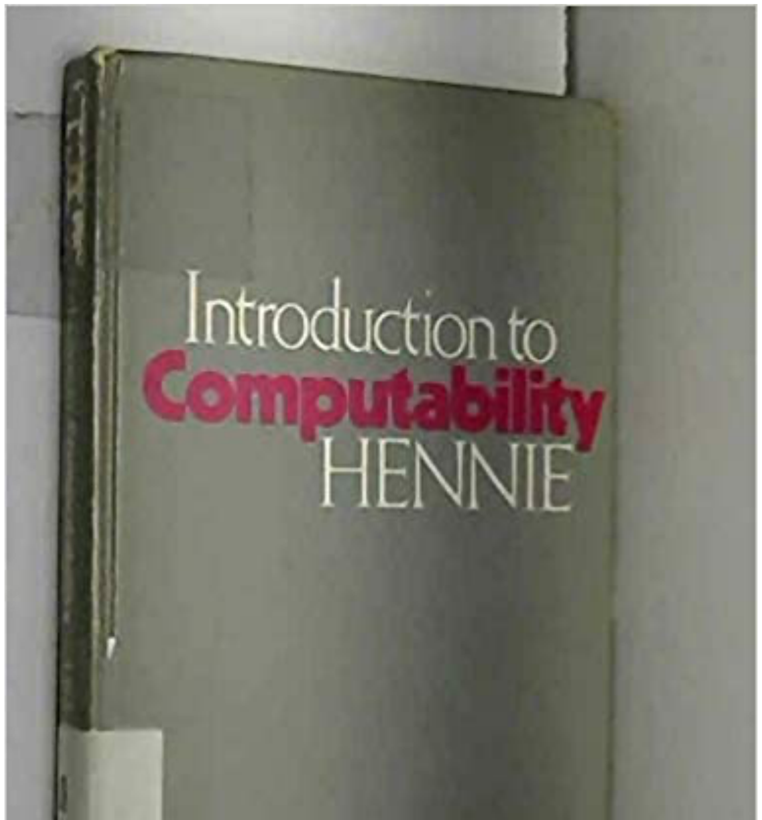
A conjecture on universal Turing machines

Conjecture. Suppose that a TM *U* is **universal** in the sense that it reproduces the **outputs** of any other TM: that is,

$$U(p_T x) = T(x)$$

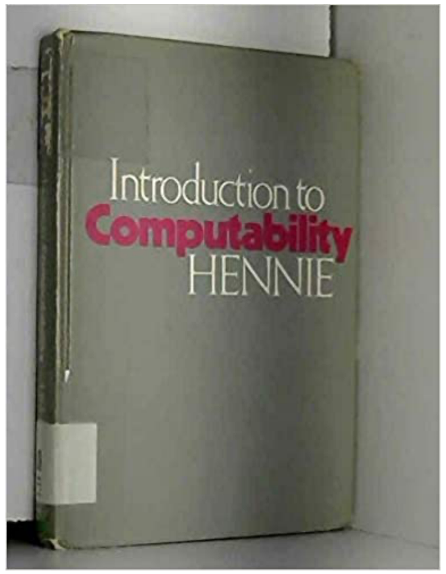
for every TM *T* and every input *x* on which *T* halts. Then, for "most" *T*, the universal TM *U* will generate its output by means of some form of **step-by-step simulation** of *T*'s computation.
In this sense, *T* is the "source" of its outputs (**computational sourcehood**).

Textbook universal TMs do this too…
… and motivate attempts to **formalize** the conjecture rigorously.



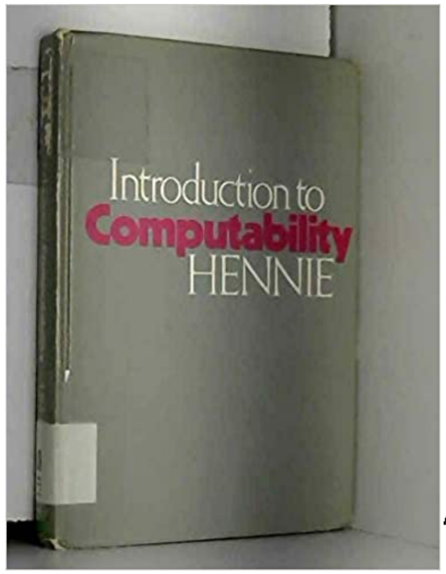
Goal: find a rigorous formulation of the conjecture that has a chance to be true.

A conjecture on universal Turing machines



Hennie: encode the tape contents of *T* on the tape of the universal machine *U*.

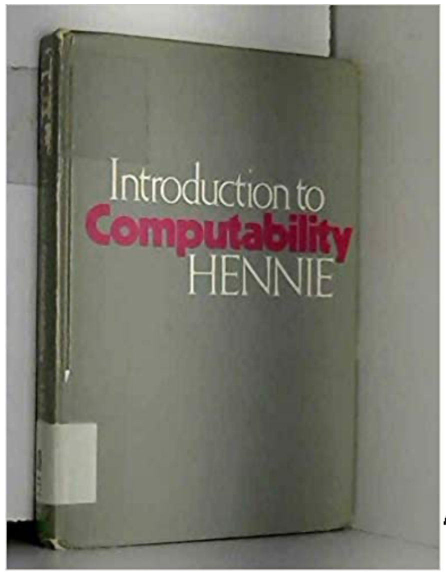
A conjecture on universal Turing machines



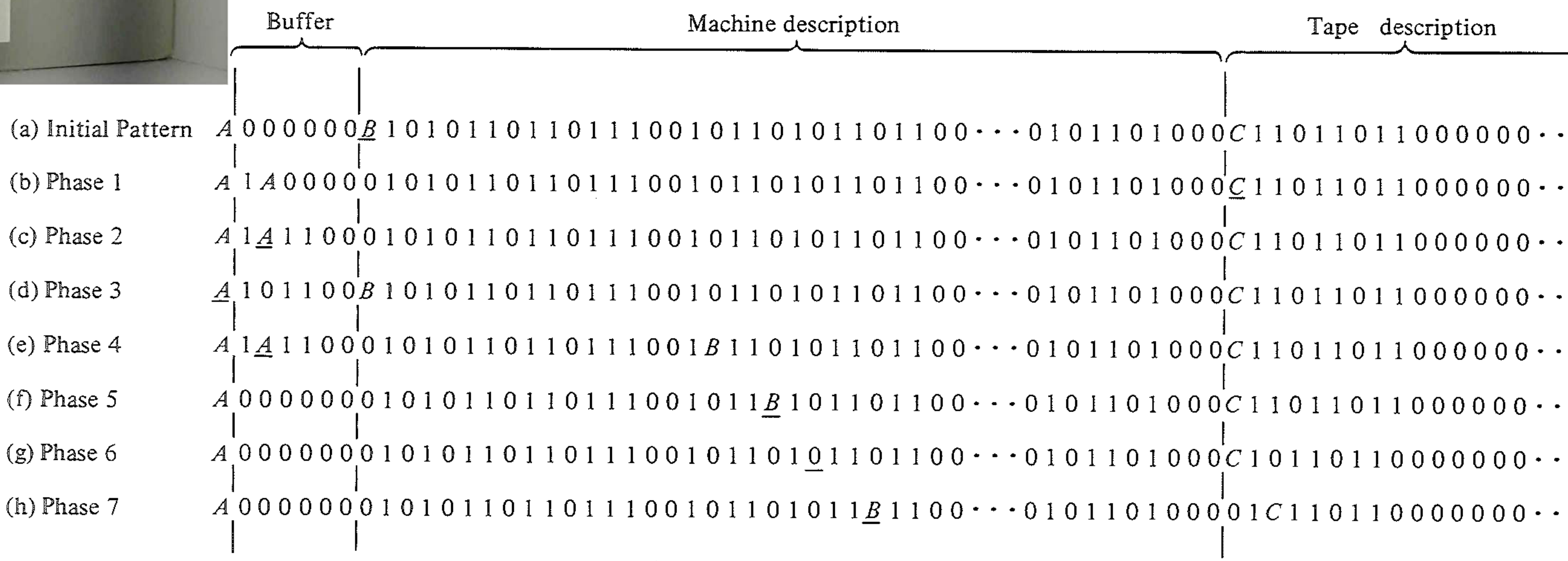
Hennie: encode the tape contents of *T* on the tape of the universal machine *U*.

	Buffer	Machine description	Tape description
(a) Initial Pattern	*A* 0 0 0 0 0 0	$\underline{B}$ 1 0 1 0 1 1 0 1 1 0 1 1 1 0 0 1 0 1 1 0 1 0 1 1 0 1 1 0 0 · · · 0 1 0 1 1 0 1 0 0 0	*C* 1 1 0 1 1 0 1 1 0 0 0 0 0 0 · · ·
(b) Phase 1	*A* 1 *A* 0 0 0 0 0	0 1 0 1 0 1 1 0 1 1 0 1 1 1 0 0 1 0 1 1 0 1 0 1 1 0 1 1 0 0 · · · 0 1 0 1 1 0 1 0 0 0	$\underline{C}$ 1 1 0 1 1 0 1 1 0 0 0 0 0 0 · · ·
(c) Phase 2	*A* 1 $\underline{A}$ 1 1 0 0	0 1 0 1 0 1 1 0 1 1 0 1 1 1 0 0 1 0 1 1 0 1 0 1 1 0 1 1 0 0 · · · 0 1 0 1 1 0 1 0 0 0	*C* 1 1 0 1 1 0 1 1 0 0 0 0 0 0 · · ·
(d) Phase 3	$\underline{A}$ 1 0 1 1 0 0	*B* 1 0 1 0 1 1 0 1 1 0 1 1 1 0 0 1 0 1 1 0 1 0 1 1 0 1 1 0 0 · · · 0 1 0 1 1 0 1 0 0 0	*C* 1 1 0 1 1 0 1 1 0 0 0 0 0 0 · · ·
(e) Phase 4	*A* 1 $\underline{A}$ 1 1 0 0	0 1 0 1 0 1 1 0 1 1 0 1 1 1 0 0 1 *B* 1 1 0 1 0 1 1 0 1 1 0 0 · · · 0 1 0 1 1 0 1 0 0 0	*C* 1 1 0 1 1 0 1 1 0 0 0 0 0 0 · · ·
(f) Phase 5	*A* 0 0 0 0 0 0	0 1 0 1 0 1 1 0 1 1 0 1 1 1 0 0 1 0 1 1 $\underline{B}$ 1 0 1 1 0 1 1 0 0 · · · 0 1 0 1 1 0 1 0 0 0	*C* 1 1 0 1 1 0 1 1 0 0 0 0 0 0 · · ·
(g) Phase 6	*A* 0 0 0 0 0 0	0 1 0 1 0 1 1 0 1 1 0 1 1 1 0 0 1 0 1 1 0 1 0 1 $\underline{0}$ 1 1 0 1 1 0 0 · · · 0 1 0 1 1 0 1 0 0 0	*C* 1 0 1 1 0 1 1 0 0 0 0 0 0 0 · · ·
(h) Phase 7	*A* 0 0 0 0 0 0	0 1 0 1 0 1 1 0 1 1 0 1 1 1 0 0 1 0 1 1 0 1 0 1 1 $\underline{B}$ 1 1 0 0 · · · 0 1 0 1 1 0 1 0 0 0	0 1 *C* 1 1 0 1 1 0 0 0 0 0 0 0 · · ·

A conjecture on universal Turing machines



Hennie: encode the tape contents of *T* on the tape of the universal machine *U*.



Instantaneous configuration of *U* (tape contents, head position, state) contains a complete image of the instantaneous configuration of *T*. One step for *T* corresponds to several steps for *U*.

A conjecture on universal Turing machines

$\mathcal{C}_T(x,t) :=$ configuration of TM *T* on input *x* after *t* steps.

Let $\mathcal{S}$ be a set of "simple functions", containing the identity *(just which set to choose best will be the main question in the following)*.

Definition. Let *T* and *T'* be TMs, and suppose there is some simple function $\varphi \in \mathcal{S}$ that maps the sequence of configurations

$$\mathcal{C}_{T'}(x,0), \mathcal{C}_{T'}(x,1), \mathcal{C}_{T'}(x,2), \ldots, \mathcal{C}_{T'}(x,t'_H)$$

to

$$\mathcal{C}_T(x,0), \mathcal{C}_T(x,1), \mathcal{C}_T(x,2), \ldots, \mathcal{C}_T(x,t_H).$$

Then we write $T \preceq_{\mathcal{S}} T'$ ("**simulation preorder**").

A conjecture on universal Turing machines

$\mathcal{C}_T(x,t) :=$ configuration of TM *T* on input *x* after *t* steps.

Let $\mathcal{S}$ be a set of "simple functions", containing the identity *(just which set to choose best will be the main question in the following)*.

Definition. Let *T* and *T'* be TMs, and suppose there is some simple function $\varphi \in \mathcal{S}$ that maps the sequence of configurations

$$\mathcal{C}_{T'}(x,0), \mathcal{C}_{T'}(x,1), \mathcal{C}_{T'}(x,2), \ldots, \mathcal{C}_{T'}(x,t'_H)$$

to

$$\mathcal{C}_T(x,0), \mathcal{C}_T(x,1), \mathcal{C}_T(x,2), \ldots, \mathcal{C}_T(x,t_H).$$

Then we write $T \preceq_{\mathcal{S}} T'$ ("**simulation preorder**").

A conjecture on universal Turing machines

$\mathcal{C}_T(x,t) :=$ configuration of TM *T* on input *x* after *t* steps.

Let $\mathcal{S}$ be a set of "simple functions", containing the identity *(just which set to choose best will be the main question in the following)*.

Definition. Let *T* and *T'* be TMs, and suppose there is some simple function $\varphi \in \mathcal{S}$ that maps the sequence of configurations

$$\mathcal{C}_{T'}(x,0), \mathcal{C}_{T'}(x,1), \mathcal{C}_{T'}(x,2), \ldots, \mathcal{C}_{T'}(x,t'_H)$$

to

$$\mathcal{C}_T(x,0), \mathcal{C}_T(x,1), \mathcal{C}_T(x,2), \ldots, \mathcal{C}_T(x,t_H).$$

Then we write $T \preceq_{\mathcal{S}} T'$ ("**simulation preorder**").

Textbook universal TMs *U* satisfy $T \preceq_{\mathcal{S}} U(p_T \bullet)$
if $\mathcal{S}$ contains a function that decodes *T*'s configuration from *U*'s.
Can this be true for **all** *U*?

A conjecture on universal Turing machines

Textbook universal TMs *U* satisfy $T \preceq_{\mathcal{S}} U(p_T \bullet)$

if $\mathcal{S}$ contains a function that decodes *T*'s configuration from *U*'s.

Can this be true for **all** *U*?

A conjecture on universal Turing machines

Textbook universal TMs *U* satisfy $T \preceq_{\mathcal{S}} U(p_T \bullet)$
if $\mathcal{S}$ contains a function that decodes *T*'s configuration from *U*'s.

Can this be true for **all** *U*? **No.**

Counterexample: Modify a textbook universal TM *U* such that it begins its operation with "**bullshit detection**".
It detects codes for a subclass of "bullshit TMs" *T* that perform a complicated calculation and then output 0.
U then just outputs 0 and halts, without simulating *T*.

A conjecture on universal Turing machines

Textbook universal TMs *U* satisfy $T \preceq_{\mathcal{S}} U(p_T \bullet)$
if $\mathcal{S}$ contains a function that decodes *T*'s configuration from *U*'s.

Can this be true for **all** *U*? **No.**

Counterexample: Modify a textbook universal TM *U* such that it begins its operation with "**bullshit detection**". It detects codes for a subclass of "bullshit TMs" *T* that perform a complicated calculation and then output 0. *U* then just outputs 0 and halts, without simulating *T*.

Conjecture. For every universal TM *U*, we have

$$T \preceq_{\mathcal{S}} U(p_T \bullet)$$

for an infinite (and "sufficiently diverse") set of TMs *T*.

How **not** to choose the set of simple functions $\mathcal{S}$

Definition. A **clock TM** is a TM that ignores its input and counts integer time steps $t \in \mathbb{N}$ on its work tape indefinitely.

How **not** to choose the set of simple functions $\mathcal{S}$

Definition. A **clock TM** is a TM that ignores its input and counts integer time steps $t \in \mathbb{N}$ on its work tape indefinitely.

Lemma. Let *C* be a clock TM. If we define $\mathcal{S}$ to be the set of **all total computable functions** on the configurations, then

$$T \preceq_{\mathcal{S}} C \quad \text{for all TMs } T.$$

That is, the clock TM *C* will formally be considered to simulate all other TMs step by step. :-(

How **not** to choose the set of simple functions $\mathcal{S}$

Definition. A **clock TM** is a TM that ignores its input and counts integer time steps $t \in \mathbb{N}$ on its work tape indefinitely.

Lemma. Let *C* be a clock TM. If we define $\mathcal{S}$ to be the set of **all total computable functions** on the configurations, then

$$T \preceq_{\mathcal{S}} C \quad \text{for all TMs } T.$$

That is, the clock TM *C* will formally be considered to simulate all other TMs step by step. :-(

Proof idea. There will be a function $\varphi \in \mathcal{S}$ that reads *x* and *t* from $\mathcal{C}_C(x,t)$ and simply **recomputes** $\mathcal{C}_T(x,t)$.

How **not** to choose the set of simple functions $\mathcal{S}$

Clearly, the total computable functions are not intuitively “simple”.
We need a much smaller set of much simpler functions. **But wait:**

How **not** to choose the set of simple functions $\mathcal{S}$

Clearly, the total computable functions are not intuitively “simple”. We need a much smaller set of much simpler functions. **But wait:**

Lemma. Let *C* be a clock TM. If we define $\mathcal{S}$ to be the set of all functions on the configurations with **linear run time**, then

$$T \preceq_{\mathcal{S}} C \quad \text{for all TMs } T.$$

That is, the clock TM *C* will formally be considered to simulate all other TMs step by step. :-(

How **not** to choose the set of simple functions $\mathcal{S}$

Clearly, the total computable functions are not intuitively “simple”. We need a much smaller set of much simpler functions. **But wait:**

Lemma. Let *C* be a clock TM. If we define $\mathcal{S}$ to be the set of all functions on the configurations with **linear run time**, then

$$T \preceq_{\mathcal{S}} C \quad \text{for all TMs } T.$$

That is, the clock TM *C* will formally be considered to simulate all other TMs step by step. **:-(**

Proof idea. Recomputation takes only linear time. **:-(**

How **not** to choose the set of simple functions $\mathcal{S}$

Clearly, the total computable functions are not intuitively "simple".
We need a much smaller set of much simpler functions. **But wait:**

Lemma. Let *C* be a clock TM. If we define $\mathcal{S}$ to be the set of all functions on the configurations with **linear run time**, then

$$T \preceq_{\mathcal{S}} C \quad \text{for all TMs } T.$$

That is, the clock TM *C* will formally be considered to simulate all other TMs step by step. **:-(**

Proof idea. Recomputation takes only linear time. **:-(**

Shall we go even simpler than linear time?
Wait a minute…

The set of simple functions $\mathcal{S}$ must contain very **complex** functions

The set of simple functions $\mathcal{S}$ must contain very **complex** functions

Start with a textbook universal TM *U*, and modify it as follows: After every step, *U* makes a **brute-force encryption** of all the tape cells that are not relevant for the next computation step, and **decrypts** the cells that *are* relevant.

Details cumbersome; see paper.

Then *U* will be extremely slow, but still **universal**.

The set of simple functions $\mathcal{S}$ must contain very **complex** functions

Start with a textbook universal TM *U*, and modify it as follows: After every step, *U* makes a **brute-force encryption** of all the tape cells that are not relevant for the next computation step, and **decrypts** the cells that *are* relevant.

Details cumbersome; see paper.

Then *U* will be extremely slow, but still **universal**.

To decode the simulated TM configuration, $\varphi \in \mathcal{S}$ must perform an immensely complex decryption.

The set of simple functions $\mathcal{S}$ must contain very **complex** functions

Start with a textbook universal TM *U*, and modify it as follows:
After every step, *U* makes a **brute-force encryption** of all the tape cells that are not relevant for the next computation step, and **decrypts** the cells that *are* relevant.

Details cumbersome; see paper.

Then *U* will be extremely slow, but still **universal**.

To decode the simulated TM configuration, $\varphi \in \mathcal{S}$ must perform an immensely complex decryption.

To have any chance that our conjecture is true,

Conjecture. For every universal TM *U*, we have

$$T \preceq_{\mathcal{S}} U(p_T \bullet)$$

for an infinite (and "sufficiently diverse") set of TMs *T*.

the set $\mathcal{S}$ must be characterized by something else than simplicity.

From simplicity to preservation of structure

From simplicity to preservation of structure

Attempt of **Definition:** A function φ on TM configurations is **structure-preserving** if for every pair of configurations c, c' such that

$$\mathcal{D}(c, c') \text{ is small,}$$

there is a pair of configurations C, C' with $\varphi(C) = c,\ \ \varphi(C') = c'$ such that

$$\mathcal{D}(C, C') \text{ is not too large.}$$

From simplicity to preservation of structure

Attempt of **Definition:** A function φ on TM configurations is **structure-preserving** if for every pair of configurations c, c' such that

$$\mathcal{D}(c, c') \text{ is small,}$$

there is a pair of configurations C, C' with $\varphi(C) = c, \ \varphi(C') = c'$ such that

$$\mathcal{D}(C, C') \text{ is not too large.}$$

This assumes the choice of some distance function $\mathcal{D}(c, c')$, e.g. the sum of the Hamming distances of the tape contents.

From simplicity to preservation of structure

Attempt of **Definition:** A function φ on TM configurations is **structure-preserving** if for every pair of configurations c, c' such that

$$\mathcal{D}(c, c') \text{ is small,}$$

there is a pair of configurations C, C' with $\varphi(C) = c$, $\varphi(C') = c'$ such that

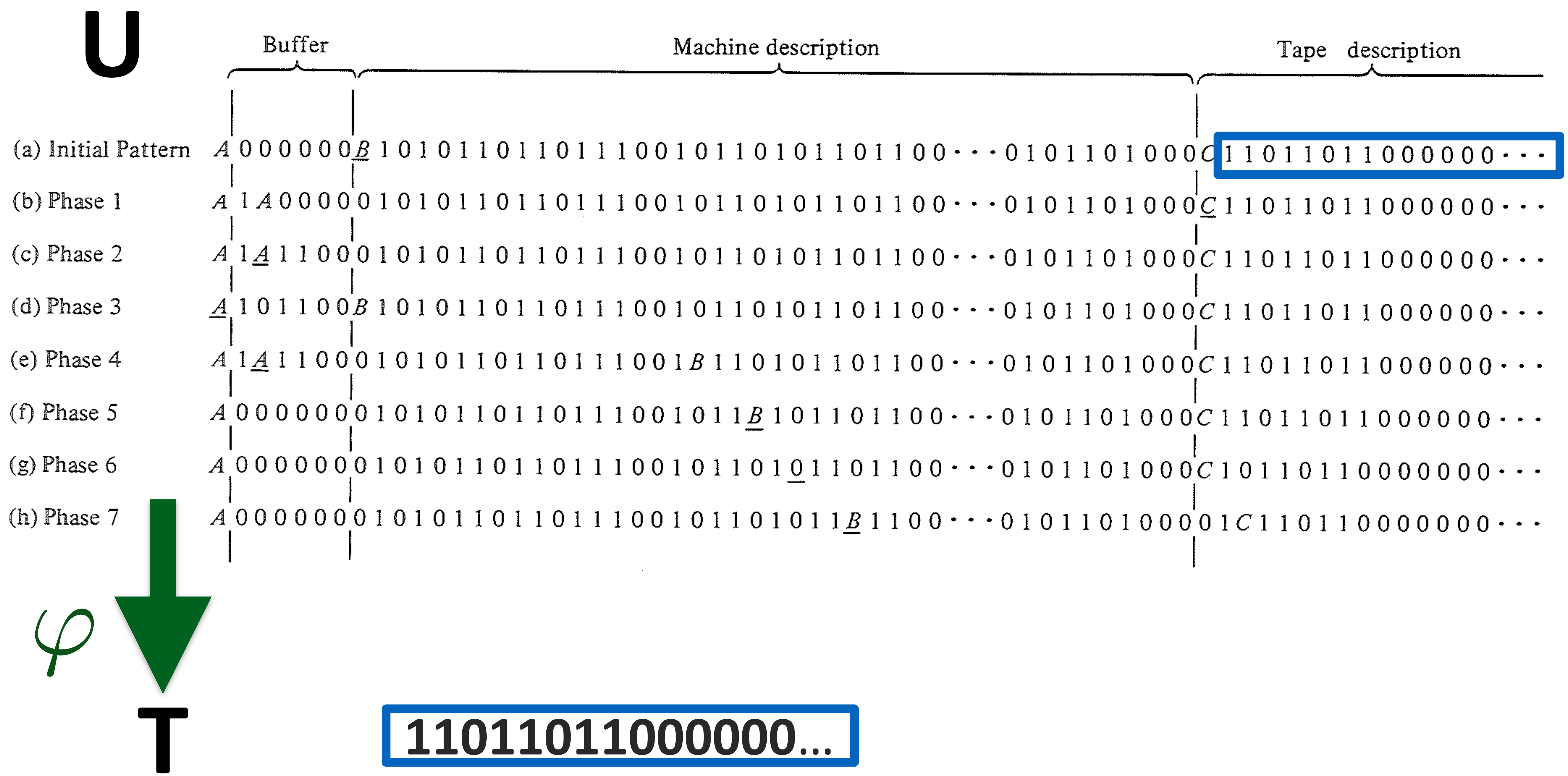
$$\mathcal{D}(C, C') \text{ is not too large.}$$

This assumes the choice of some distance function $\mathcal{D}(c, c')$, e.g. the sum of the Hamming distances of the tape contents.

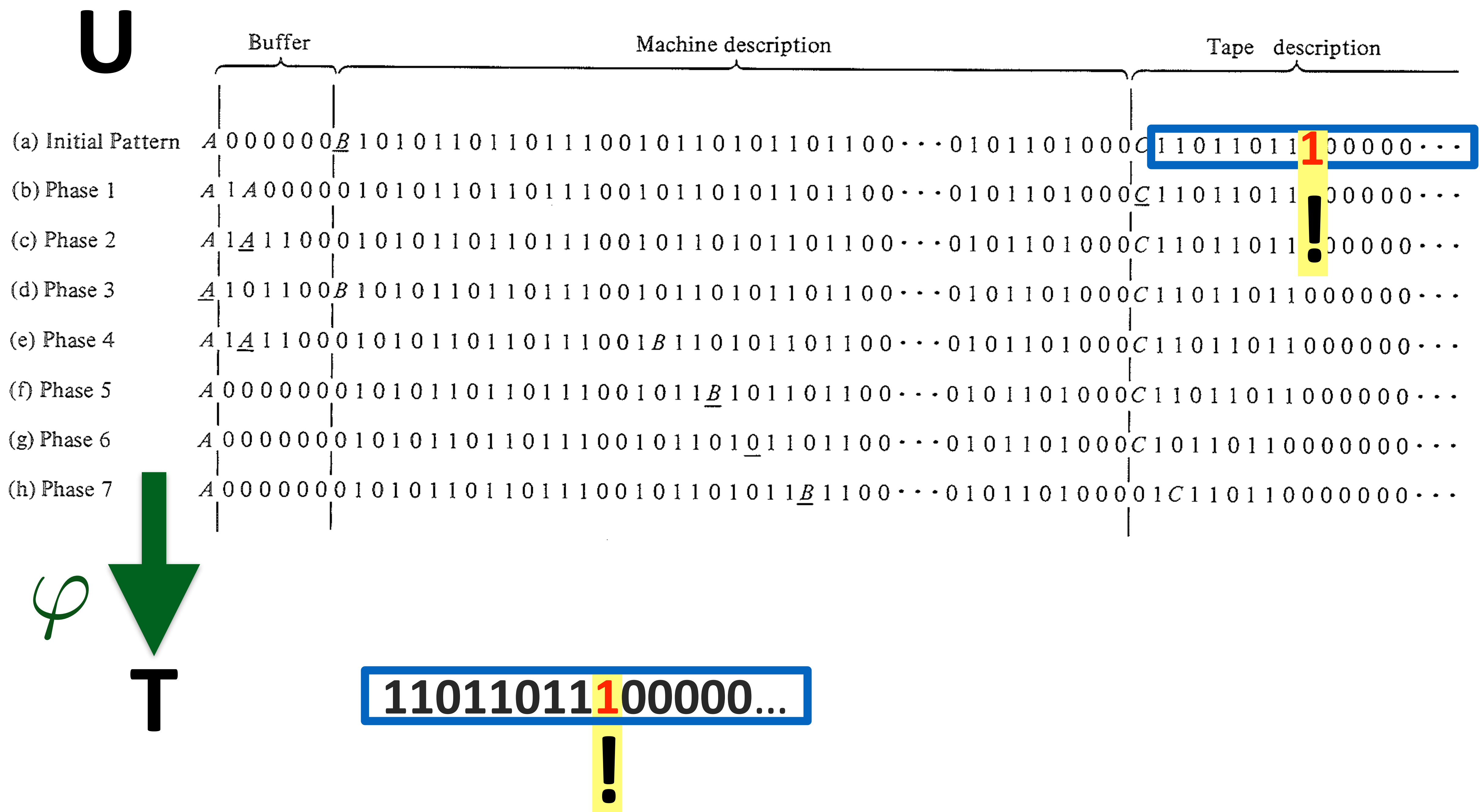
Decoding functions for **standard textbook universal TMs** *are* structure-preserving in this sense!

→

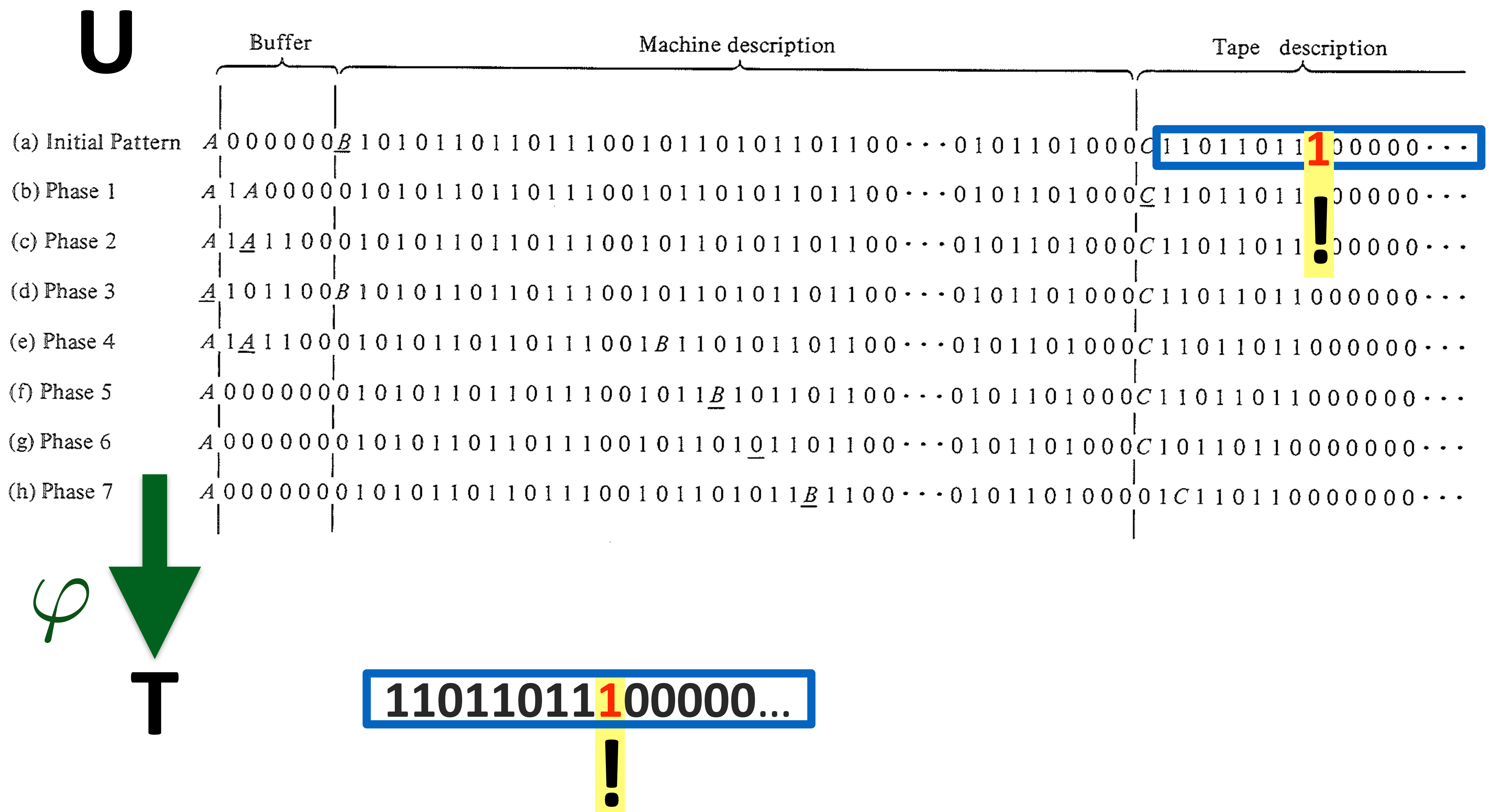
Hennie's *U* and structure-preserving decoding



Hennie's *U* and structure-preserving decoding



Hennie's *U* and structure-preserving decoding



For these two configurations c_T, c'_T with $\mathcal{D}(c_T, c'_T) = 1$,

we find C_U, C'_U with $\varphi(C_U) = c_T, \varphi(C'_U) = c'_T$ and

$$\mathcal{D}(C_U, C'_U) = 1.$$

Clock TMs violate this condition

Clock TMs violate this condition

Recall the clock TM *C* and the "cheating" function φ.

It reads *x* and *t* from $\mathcal{C}_C(x,t)$ and **recomputes** $\mathcal{C}_T(x,t)$.

Clock TMs violate this condition

Recall the clock TM *C* and the "cheating" function φ.
It reads *x* and *t* from $\mathcal{C}_C(x,t)$ and **recomputes** $\mathcal{C}_T(x,t)$.

Such φ should not be allowed. Because otherwise,
our definition would claim that *C* simulates *T* step-by-step.

Clock TMs violate this condition

Recall the clock TM *C* and the “cheating” function φ.
It reads *x* and *t* from $\mathcal{C}_C(x,t)$ and **recomputes** $\mathcal{C}_T(x,t)$.

Such φ should not be allowed. Because otherwise,
our definition would claim that *C* simulates *T* step-by-step.

Indeed, φ is **not structure-preserving**: for every *N,*
there are configurations c_T, c'_T with $\mathcal{D}(c_T, c'_T) = 1$

such that **all** C_U, C'_U with $\varphi(C_U) = c_T, \varphi(C'_U) = c'_T$

have Hamming distance $\mathcal{D}(C_U, C'_U) > N.$

Clock TMs violate this condition

Recall the clock TM *C* and the "cheating" function φ.
It reads *x* and *t* from $\mathcal{C}_C(x,t)$ and **recomputes** $\mathcal{C}_T(x,t)$.

Such φ should not be allowed. Because otherwise, our definition would claim that *C* simulates *T* step-by-step.

Indeed, φ is **not structure-preserving**: for every *N*, there are configurations c_T, c'_T with $\mathcal{D}(c_T, c'_T) = 1$ such that **all** C_U, C'_U with $\varphi(C_U) = c_T, \varphi(C'_U) = c'_T$ have Hamming distance $\mathcal{D}(C_U, C'_U) > N$.

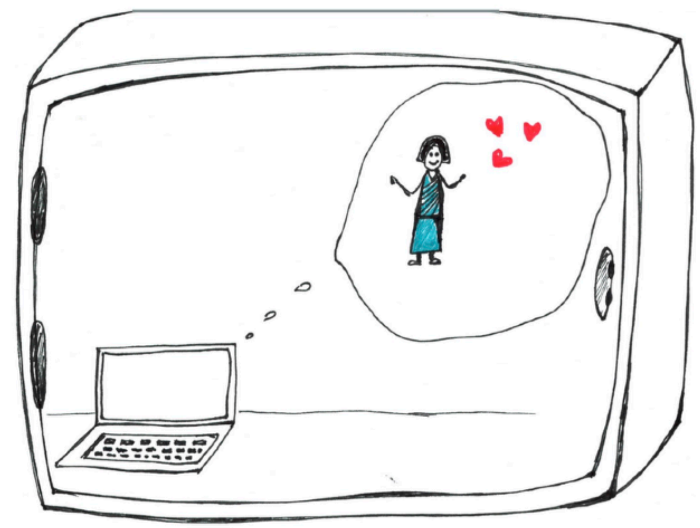
Amended conjecture. For every universal TM *U*, we have

$$T \preceq_{\mathcal{S}} U(p_T \bullet)$$

for an infinite and diverse set of TMs *T*, where $\mathcal{S}$ is a natural set of structure-preserving functions on TM configurations.

Conclusions

- Wolfram's **Computational Irreducibility** and apparent free will
- **Computational sourcehood** as an attempt at defining an aspect of *actual* free will. Motivation: thought experiment of John the cook

- Conjecture on **universal TMs**: they must typically simulate step-by-step.
- **Attempt at formalization** via ~~simple~~ structure-preserving functions

M. Krumm and M. P. Müller, **arXiv:2101.12033**
(To be updated soon)

Thank you!