Church's Theorem*

Selmer Bringsjord
(w TM slides by Naveen Sundar G)

Rensselaer AI & Reasoning (RAIR) Lab
Department of Cognitive Science
Department of Computer Science
Lally School of Management & Technology
Rensselaer Polytechnic Institute (RPI)
Troy, New York 12180 USA

Intermediate Formal Logic & AI (IFLAI2)
9/13/2021 ver 4

*Many thx to Can Mekik for corrections in the featured Turing machine.

Mechanics …

Mechanics …

`ChisholmsParadox` released.

Mechanics …

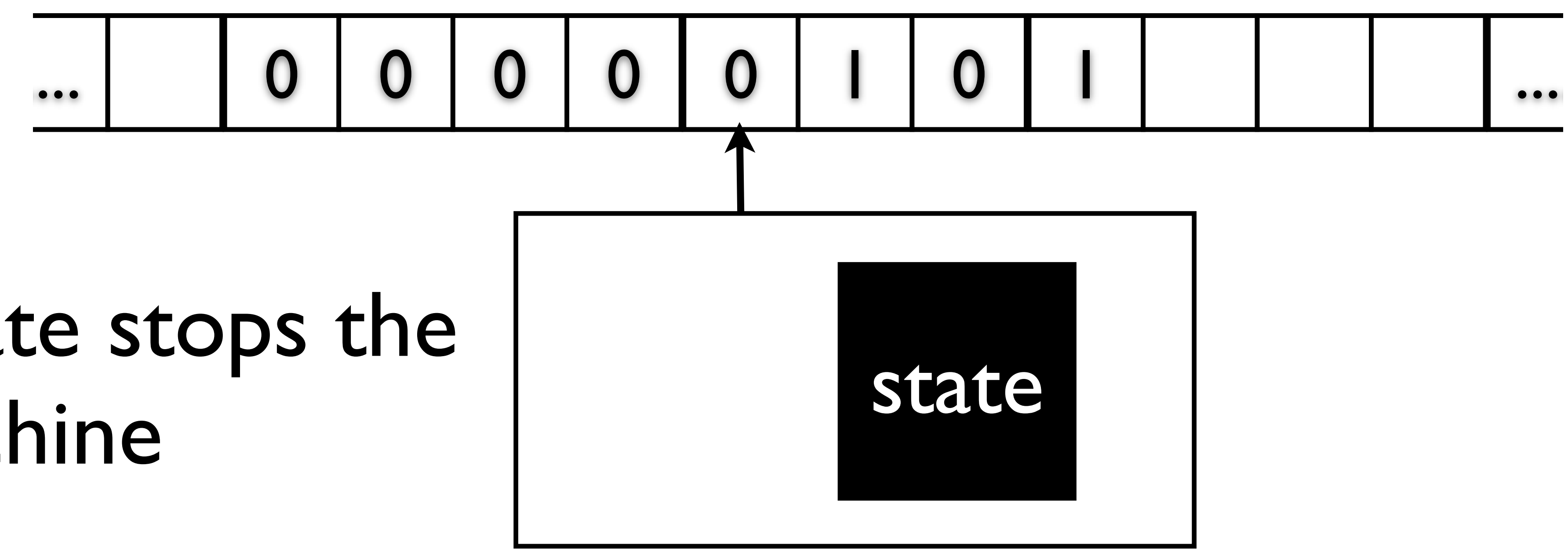
`ChisholmsParadox` released.

Questions? …

Turing-decidability/computability

. . .

Turing Machines



a special state stops the machine

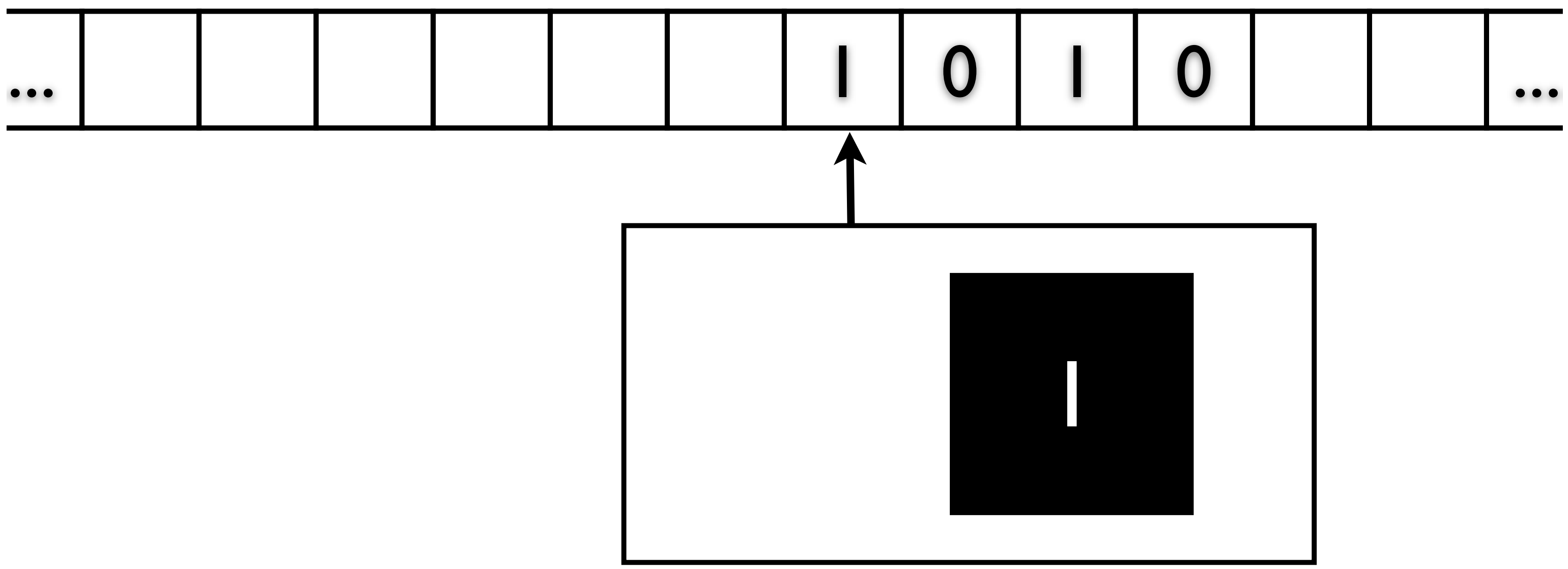
Program

current state	current symbol	next state	next symbol	direction

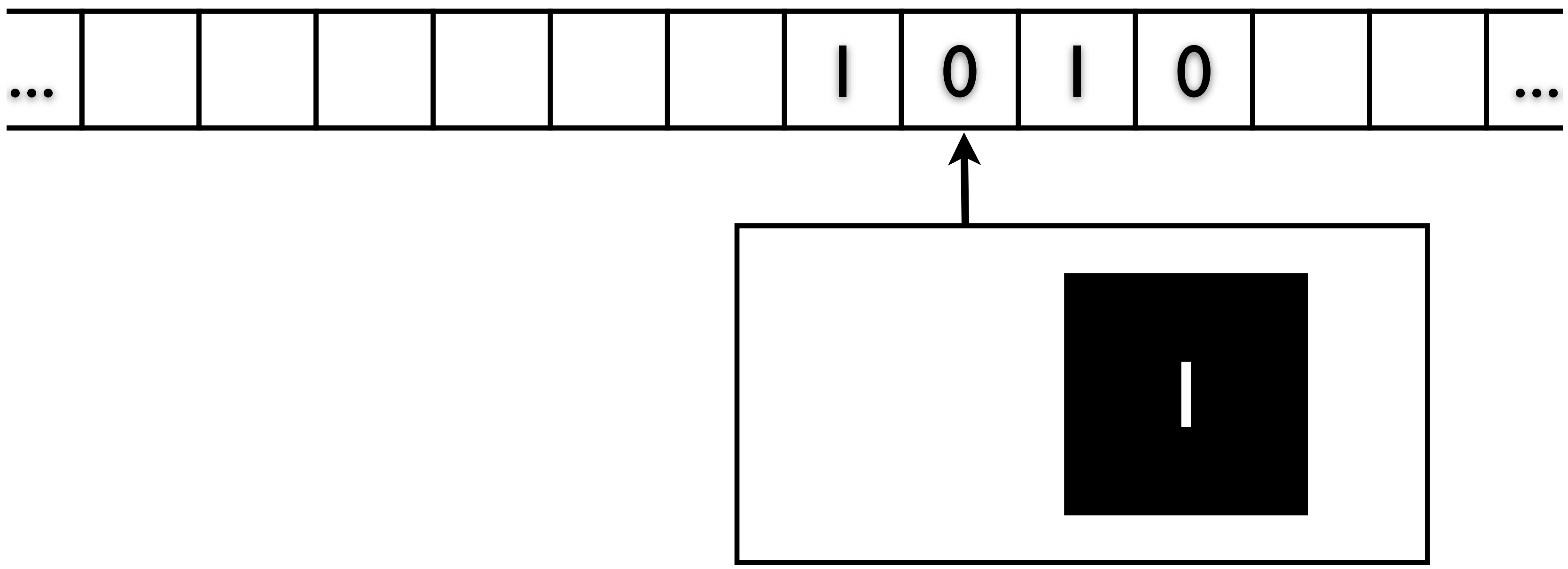
Even Number Function

- $f(n) = 1$ if n is even; else $f(n) = 0$

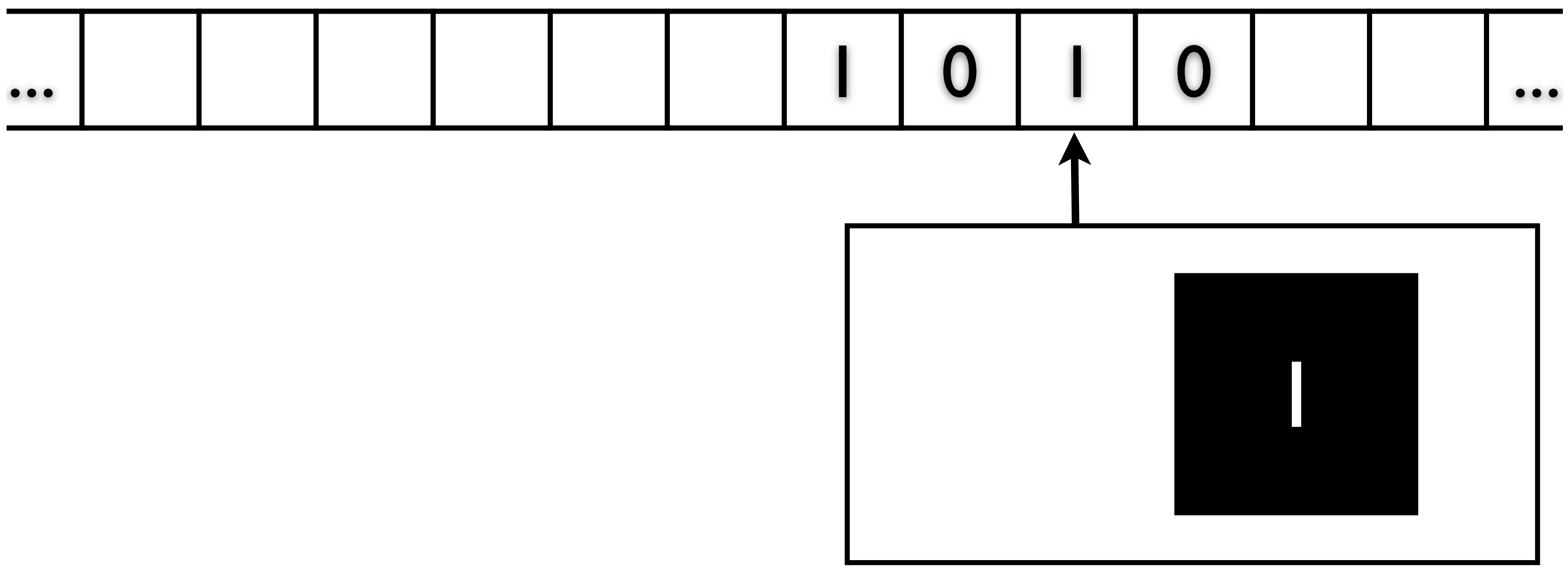
current state	current symbol	next state	next symbol	direction
3	0	3	blank	Left
3	1	3	blank	Left
3	blank	stop	0	Same
2	0	2	blank	Left
2	1	2	blank	Left
2	blank	stop	1	Same
1	1	1	1	Right
1	0	1	0	Right
1	blank	4	blank	Left
4	0	2	0	Same
4	1	3	1	Left



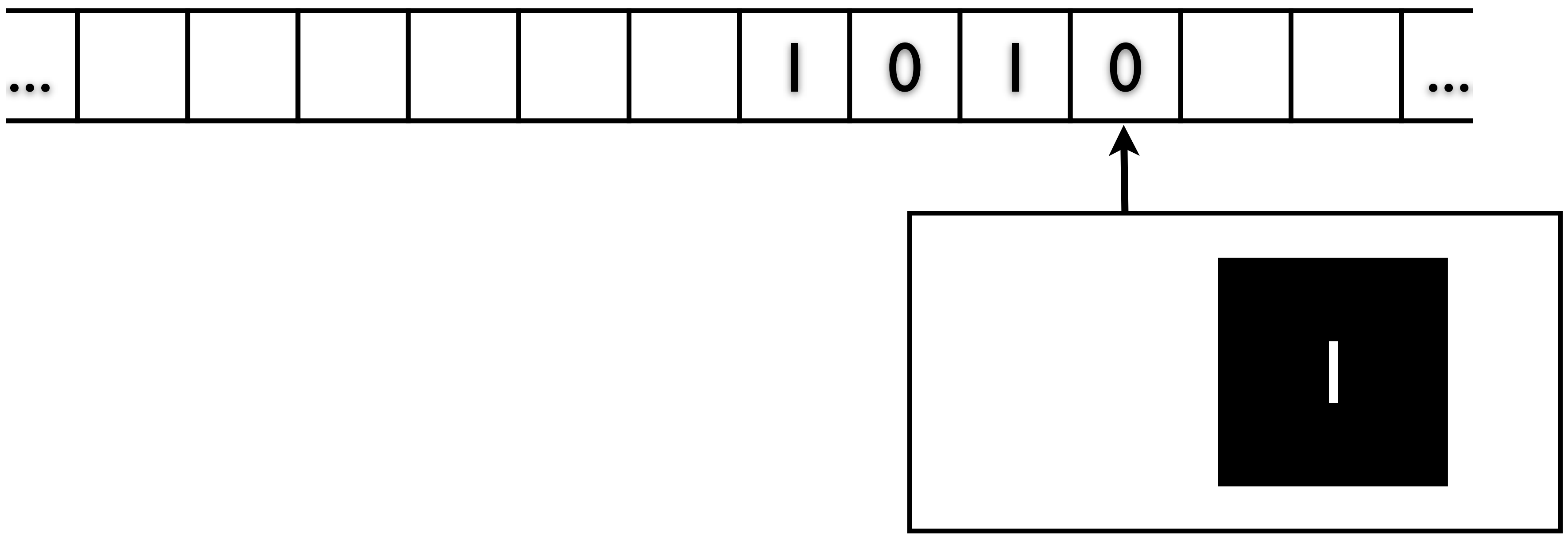
current state	current symbol	next state	next symbol	direction
3	0	3	blank	Left
3	1	3	blank	Left
3	blank	stop	0	Same
2	0	2	blank	Left
2	1	2	blank	Left
2	blank	stop	1	Same
1	1	1	1	Right
1	0	1	0	Right
1	blank	4	blank	Left
4	0	2	0	Same
4	1	3	1	Left



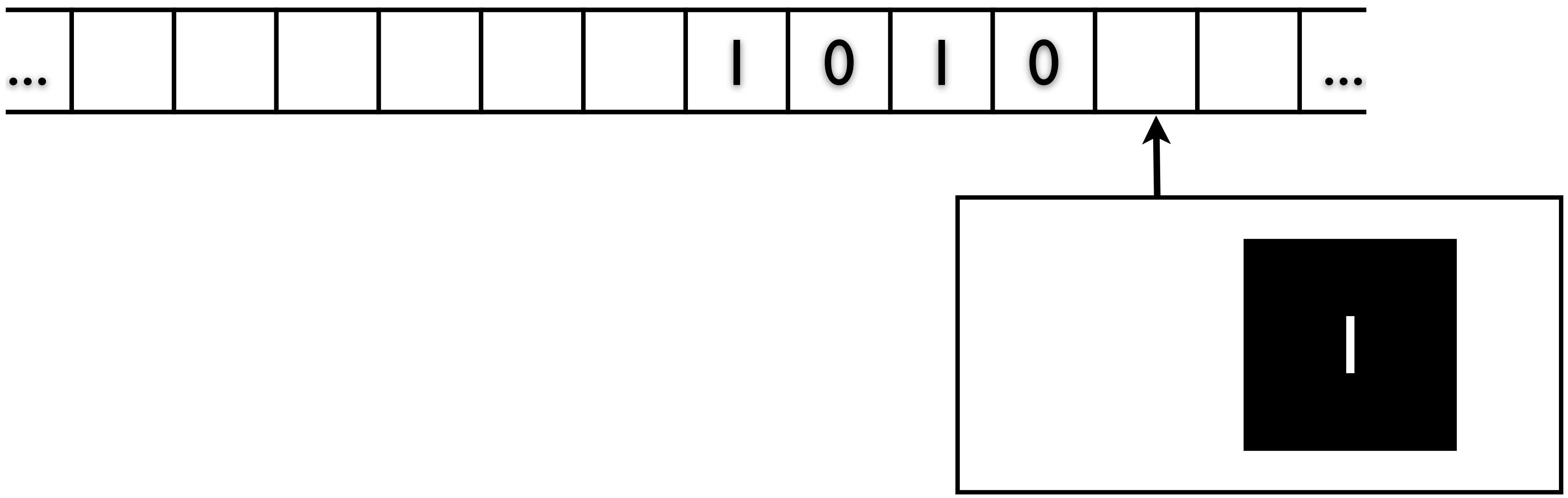
current state	current symbol	next state	next symbol	direction
3	0	3	blank	Left
3	1	3	blank	Left
3	blank	stop	0	Same
2	0	2	blank	Left
2	1	2	blank	Left
2	blank	stop	1	Same
1	1	1	1	Right
1	0	1	0	Right
1	blank	4	blank	Left
4	0	2	0	Same
4	1	3	1	Left



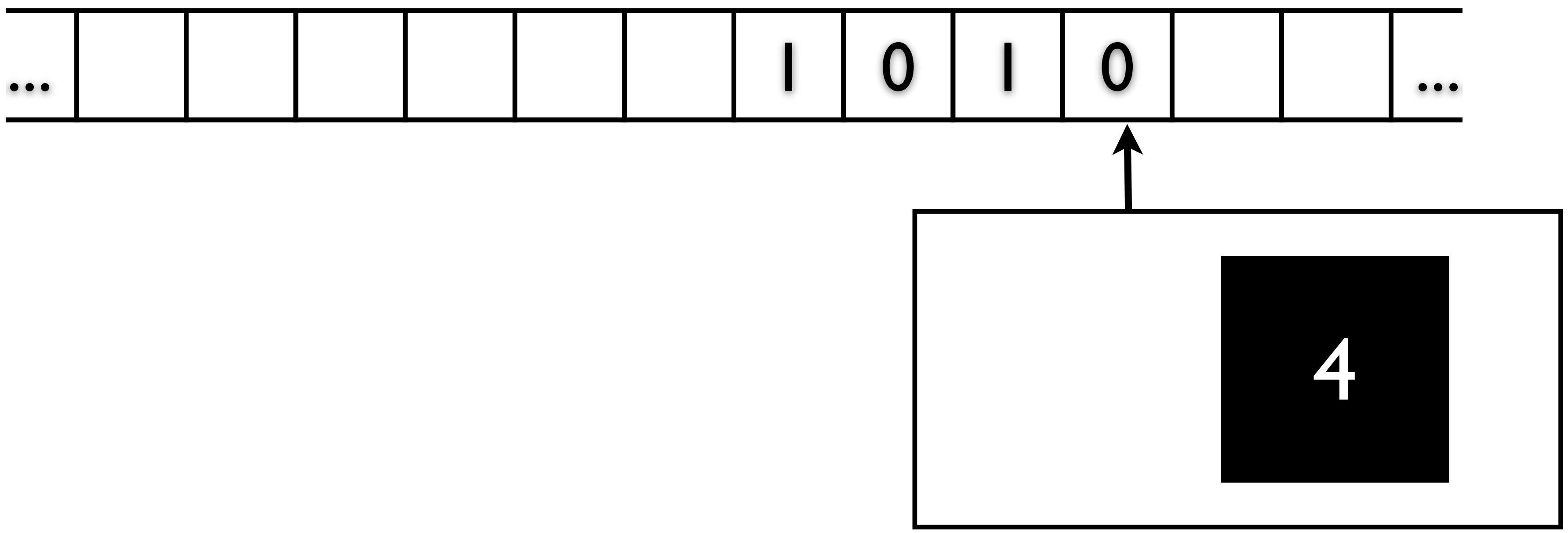
current state	current symbol	next state	next symbol	direction
3	0	3	blank	Left
3	1	3	blank	Left
3	blank	stop	0	Same
2	0	2	blank	Left
2	1	2	blank	Left
2	blank	stop	1	Same
1	1	1	1	Right
1	0	1	0	Right
1	blank	4	blank	Left
4	0	2	0	Same
4	1	3	1	Left



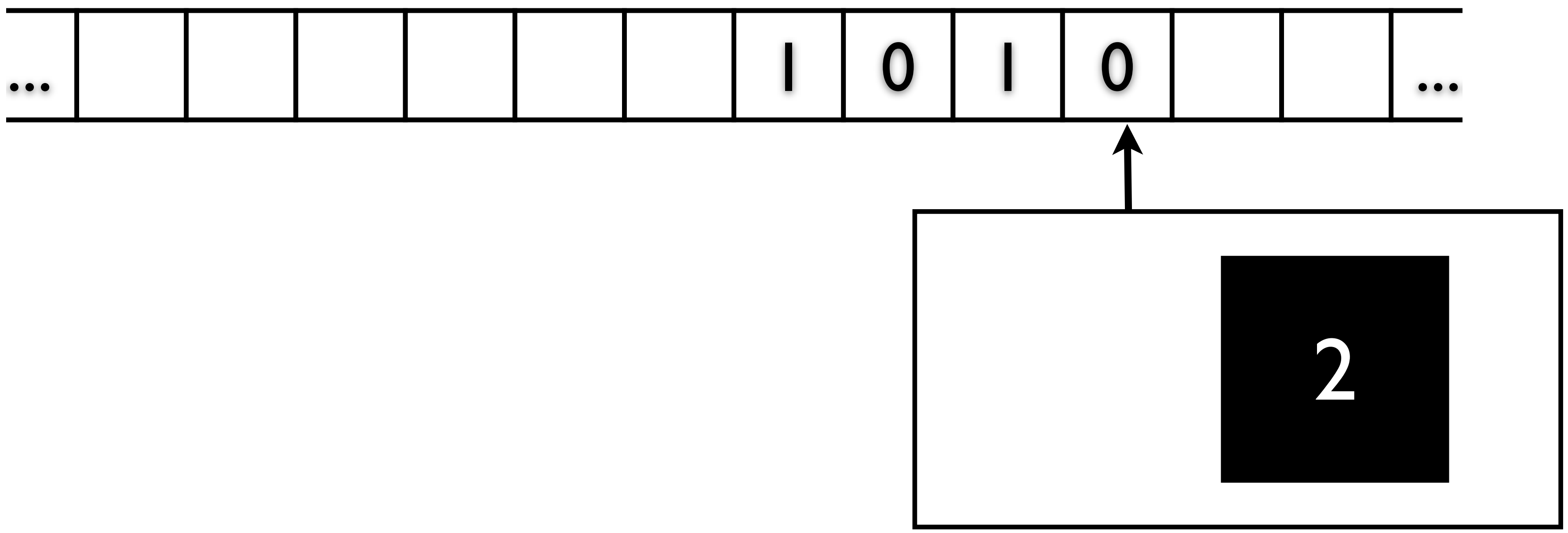
current state	current symbol	next state	next symbol	direction
3	0	3	blank	Left
3	1	3	blank	Left
3	blank	stop	0	Same
2	0	2	blank	Left
2	1	2	blank	Left
2	blank	stop	1	Same
1	1	1	1	Right
1	0	1	0	Right
1	blank	4	blank	Left
4	0	2	0	Same
4	1	3	1	Left



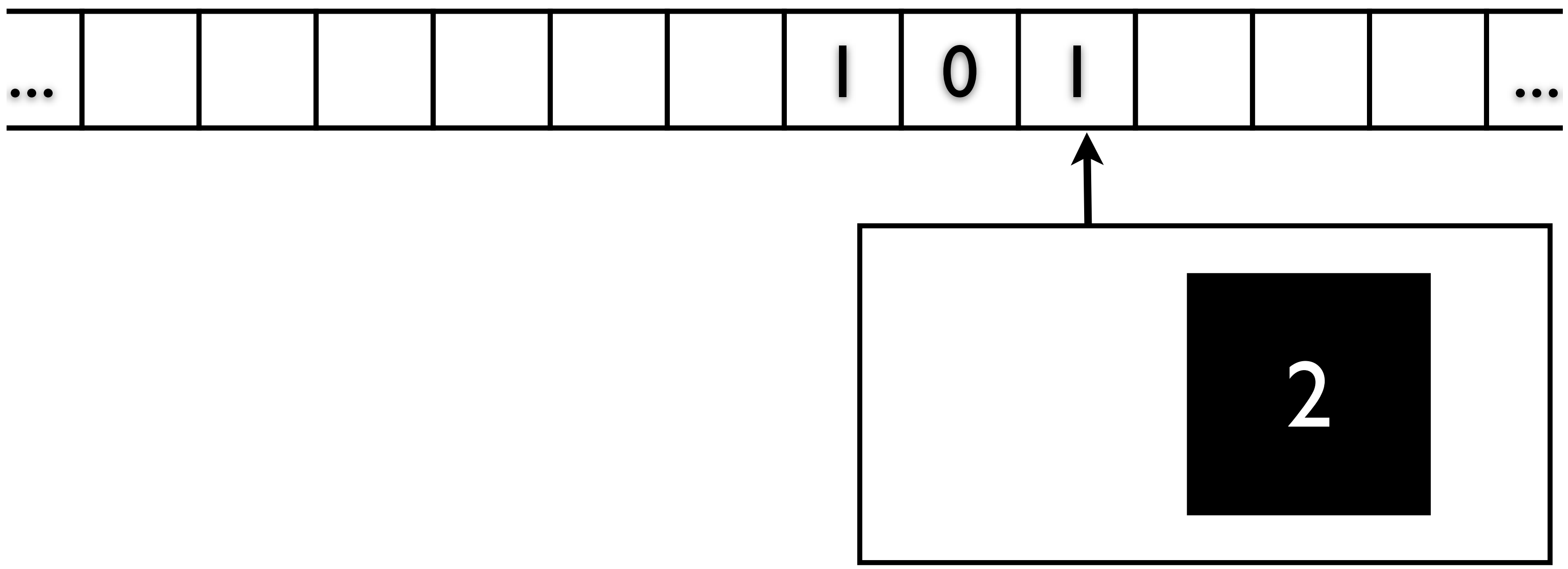
current state	current symbol	next state	next symbol	direction
3	0	3	blank	Left
3	1	3	blank	Left
3	blank	stop	0	Same
2	0	2	blank	Left
2	1	2	blank	Left
2	blank	stop	1	Same
1	1	1	1	Right
1	0	1	0	Right
1	blank	4	blank	Left
4	0	2	0	Same
4	1	3	1	Left



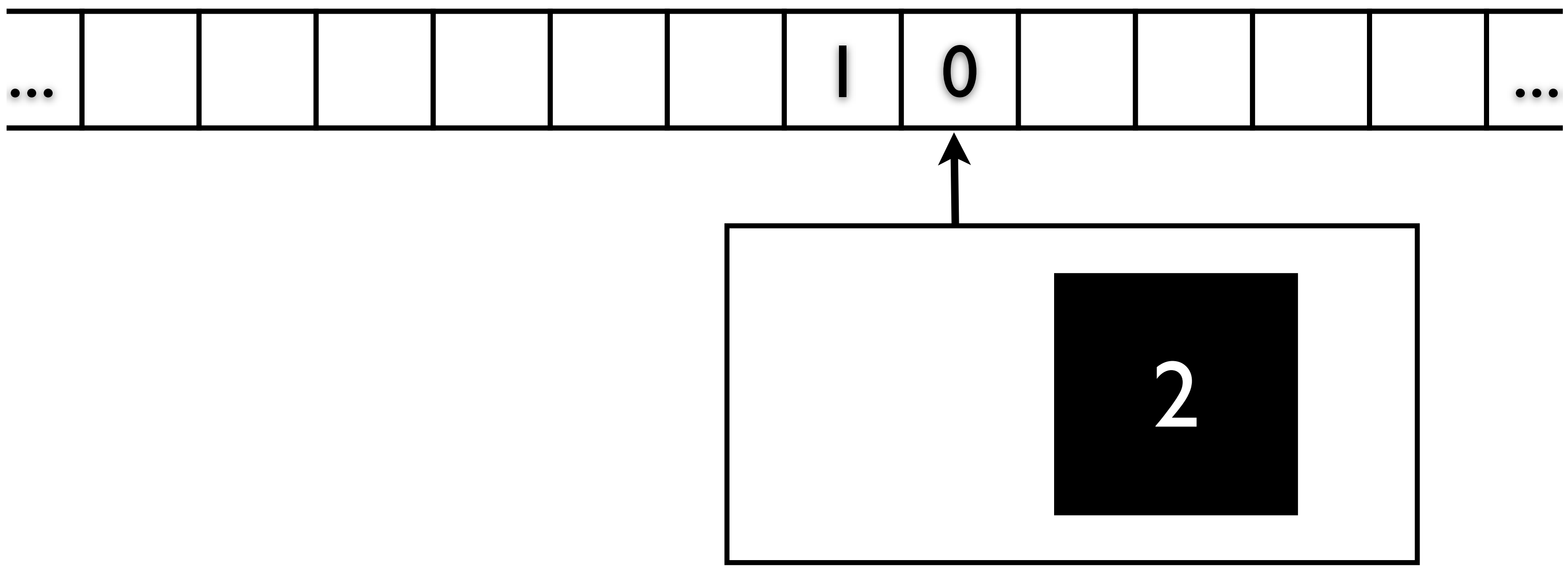
current state	current symbol	next state	next symbol	direction
3	0	3	blank	Left
3	1	3	blank	Left
3	blank	stop	0	Same
2	0	2	blank	Left
2	1	2	blank	Left
2	blank	stop	1	Same
1	1	1	1	Right
1	0	1	0	Right
1	blank	4	blank	Left
4	0	2	0	Same
4	1	3	1	Left



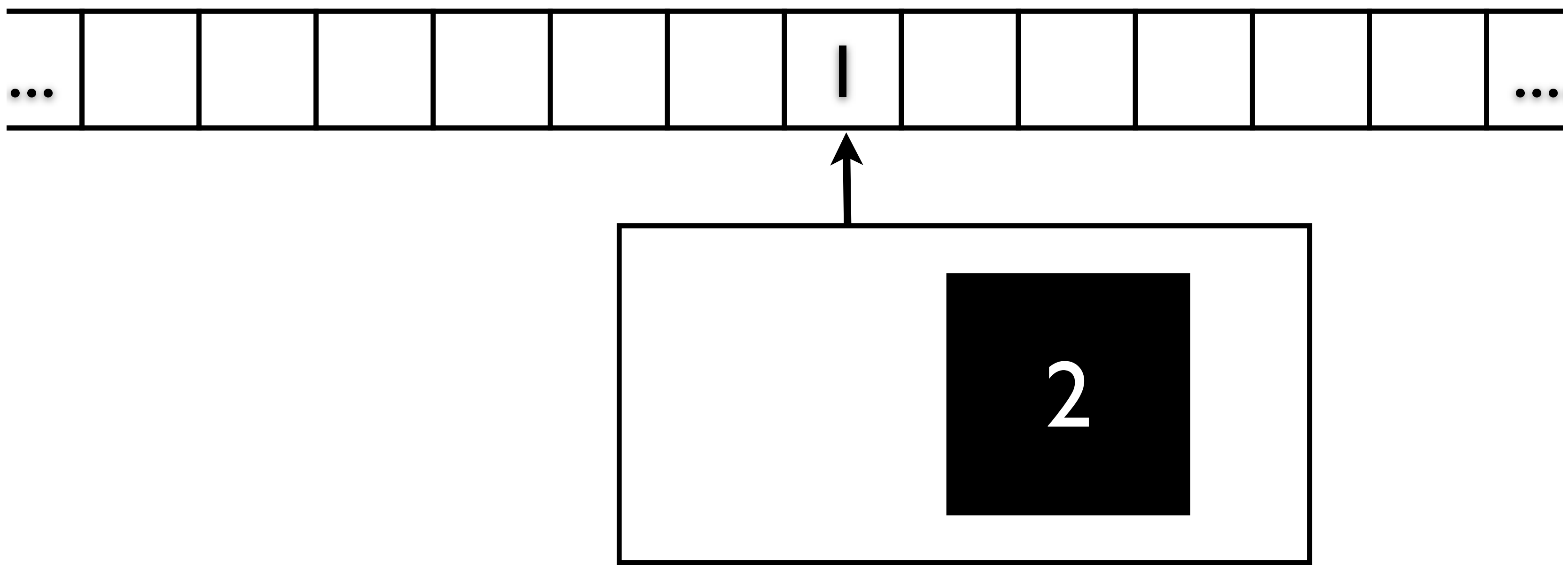
current state	current symbol	next state	next symbol	direction
3	0	3	blank	Left
3	1	3	blank	Left
3	blank	stop	0	Same
2	0	2	blank	Left
2	1	2	blank	Left
2	blank	stop	1	Same
1	1	1	1	Right
1	0	1	0	Right
1	blank	4	blank	Left
4	0	2	0	Same
4	1	3	1	Left



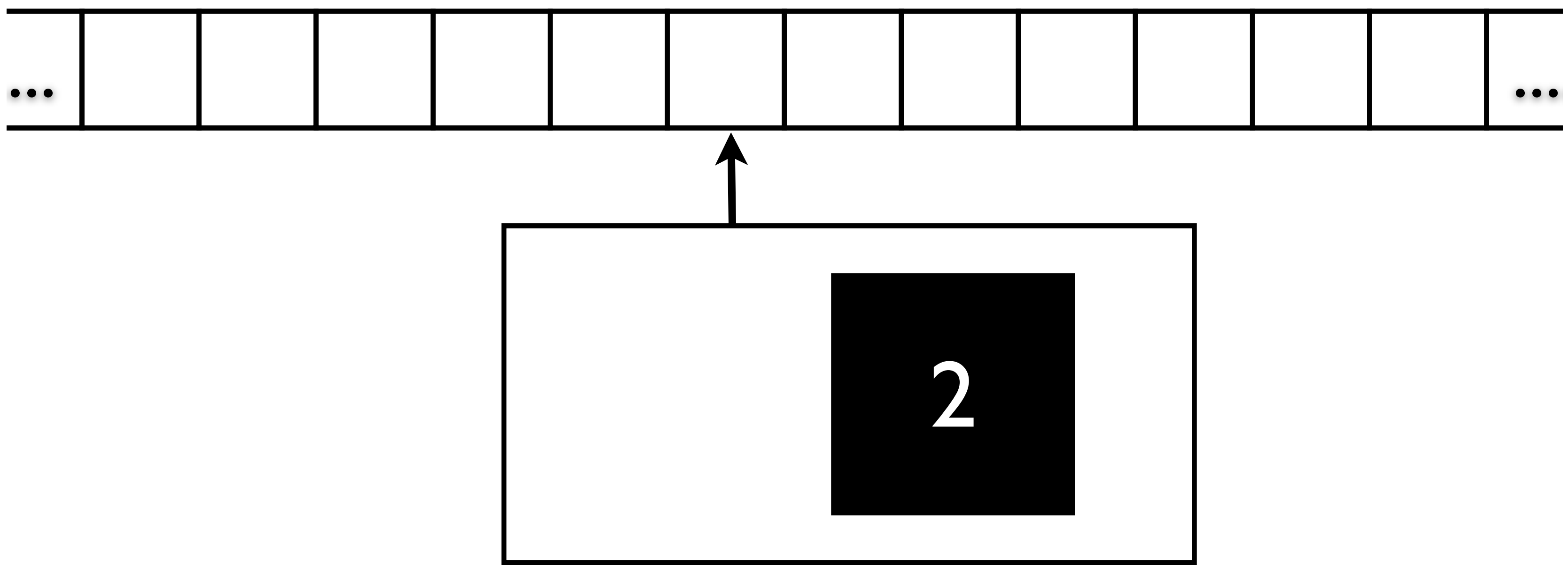
current state	current symbol	next state	next symbol	direction
3	0	3	blank	Left
3	1	3	blank	Left
3	blank	stop	0	Same
2	0	2	blank	Left
2	1	2	blank	Left
2	blank	stop	1	Same
1	1	1	1	Right
1	0	1	0	Right
1	blank	4	blank	Left
4	0	2	0	Same
4	1	3	1	Left



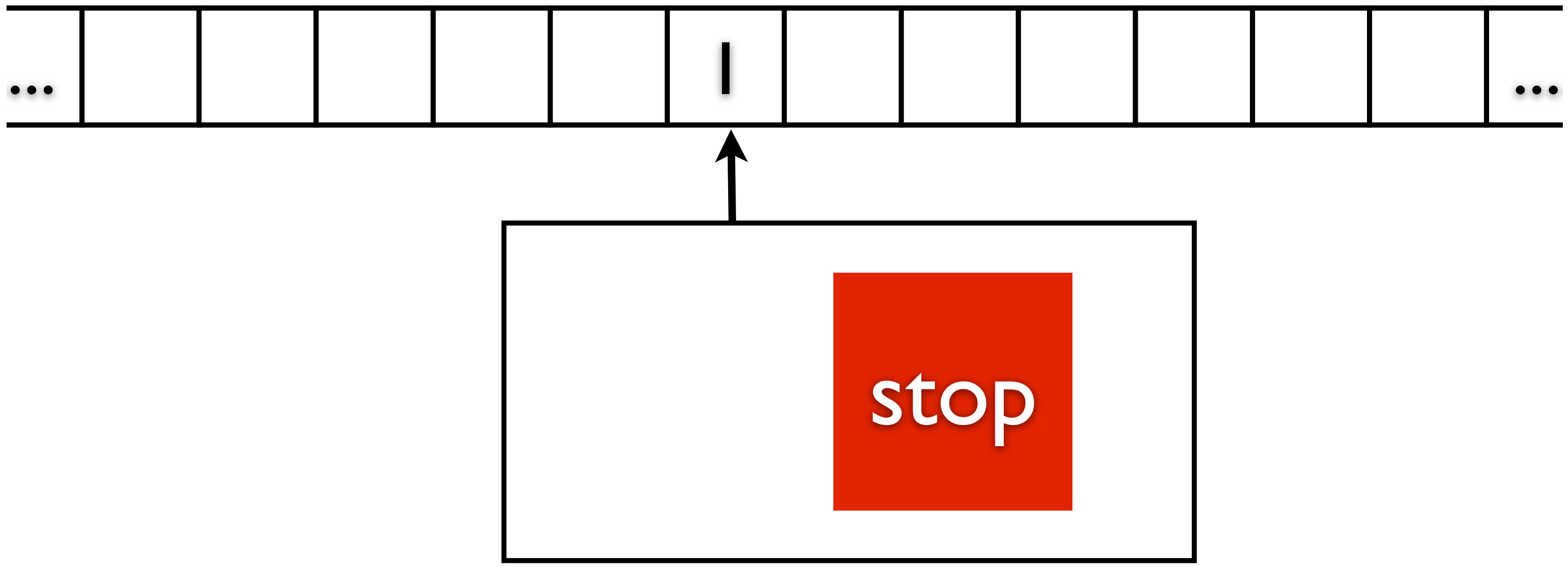
current state	current symbol	next state	next symbol	direction
3	0	3	blank	Left
3	1	3	blank	Left
3	blank	stop	0	Same
2	0	2	blank	Left
2	1	2	blank	Left
2	blank	stop	1	Same
1	1	1	1	Right
1	0	1	0	Right
1	blank	4	blank	Left
4	0	2	0	Same
4	1	3	1	Left



current state	current symbol	next state	next symbol	direction
3	0	3	blank	Left
3	1	3	blank	Left
3	blank	stop	0	Same
2	0	2	blank	Left
2	1	2	blank	Left
2	blank	stop	1	Same
1	1	1	1	Right
1	0	1	0	Right
1	blank	4	blank	Left
4	0	2	0	Same
4	1	3	1	Left



current state	current symbol	next state	next symbol	direction
3	0	3	blank	Left
3	1	3	blank	Left
3	blank	stop	0	Same
2	0	2	blank	Left
2	1	2	blank	Left
2	blank	stop	1	Same
1	1	1	1	Right
1	0	1	0	Right
1	blank	4	blank	Left
4	0	2	0	Same
4	1	3	1	Left



current state	current symbol	next state	next symbol	direction
3	0	3	blank	Left
3	1	3	blank	Left
3	blank	stop	0	Same
2	0	2	blank	Left
2	1	2	blank	Left
2	blank	stop	1	Same
1	1	1	1	Right
1	0	1	0	Right
1	blank	4	blank	Left
4	0	2	0	Same
4	1	3	1	Left

- Functions that can be computed in this manner are *Turing-computable*.

- Functions that can be computed in this manner are *Turing-computable*.
- Decision problems (Yes/No problems) that can answered in this manner are *Turing-decidable*. (Here, 1 can be used for **Y**; 2 for **N**.)

For more on TMs …

https://plato.stanford.edu/entries/turing-machine

Theorem: The Halting Problem is Turing-unsolvable.

. . .

We assume an encoding of TMs that permits identification of each with some $m \in \mathbb{Z}^+$, and say that the binary halt function h maps a machine and its input to 1 if that machine halts, and to 2 if it doesn't:

$\forall m, n\ [h(m,n) = 1 \rightarrow Goes(m, n, \text{halt})]$

$$h(m,n) = 1 \text{ if } m : n \longrightarrow \text{ halt}$$
$$h(m,n) = 2 \text{ if } m : n \longrightarrow \infty$$

So, the theorem we need can be expressed this way:

$$(\star) \quad \neg \exists m^h\ [m^h \text{ computes } h]$$

where a TM that computes a function f starts with arguments to f on its tape and goes to the value of f on applied to those arguments. Next, let's construct a TM m^c that copies a block of 1's (separated by a blank #), and (what BBJ in their *Computability & Logic* call) a "dithering" TM:

$$m^d : n \longrightarrow \text{ halt if } n > 1;\ \ m^d : n \longrightarrow \infty \text{ if } n = 1$$

Proof: Suppose for *reductio* that m^{h*} [this is our witness for the existential quantifier in (★)] computes h. Then we can make a composite machine m^3 consisting of m_c connected to and feeding m^{h*} which is in turn connected to and feeding m^d. It's easy to see (use some paper and pencil/stylus and tablet!) that

(1) if $h(n, n) = 1$, then $m^3 : n \longrightarrow \infty$

and

(2) if $h(n, n) = 2$, then $m^3 : n \longrightarrow$ halt.

To reach our desired contradiction, we simply ask: What happens when we instantiate n to m^3 in (1) and (2)? (E.g., perhaps the TM m^3 is 5, then we would have $h(5,5)$.) The answer to this question, and its leading directly to just what the doctor ordered, is left to the reader. **QED**

Church's Theorem & its proof …

Church's Theorem: The *Entscheidungsproblem* is Turing-unsolvable.

Proof-sketch: We need to show that the question $\Phi \vdash \phi$? is not Turing-decidable. (Here we are working within the framework of $\mathscr{L}_1$.) To begin, note that competent users of HyperSlate® know that any Turing machine m can be formalized in a HyperSlate® workspace. (Explore! Prove it to yourself in hands-on fashion!) They will also then know that

$$(\dagger) \quad \forall m, n \in \mathbb{N}\ \exists \Phi, \phi\ [\Phi \vdash \phi \leftrightarrow\ m : n \longrightarrow \text{ halt}]$$

where Φ and ϕ are built in HyperSlate®.

Now, let's assume for contradiction that theoremhood in first-order logic can be decided by a Turing machine m_t. But this is absurd. Why? Because imagine that someone now comes to us asking whether some arbitrary TM m halts. We can infallibly and algorithmically supply a correct answer, because we can formalize m in line with (†) and then employ m_t to given us the answer. **QED**