Announcements

- HW1 grades released. Excellent work!
- HW2 will be handed out later this week
  (Due Nov 15)

  last 2-3 questions on computability
  which we will cover this week

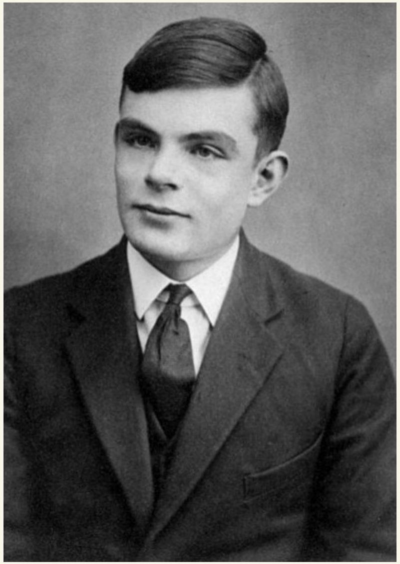
Lecture notes: Skipping FO Resolution (supplementary notes)
skipping Herbrand thm (pp 39 - 42)

COMPUTABILITY

(Lecture notes: pp 54-65)

Turing Machines

"On computable Numbers, with an application to the Entscheidungsproblem"
1936



1912 - 1954

- Concept of 1st generally convincing general model of computation.
- Proved there is no algorithm for deciding truth in mathematics
- code breaking of Nazi ciphers WW II
- also worked in mathematical biology
- prosecuted in '52 for homosexuality

Turing Machines

$$M = \{ Q, \Sigma, \Gamma, \delta, q_1, B, \{q_2\} \}$$

$Q = \{q_1, \ldots, q_k\}$ states, $k \geq 2$

Σ = finite input alphabet, including 0, 1

$\not{b}$

Γ = finite tape alphabet, $\Sigma \subseteq \Gamma$, includes B (blank symbol)

q_1 : start state

q_2 : halt state

$$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L \times R\}$$

Ex. Parity:

given $x \in \{0,1\}^*$ want M st.

$M(x)$ outputs 1 if # 1's in x is odd
0 " " " " " even

Start in state q_0

q_1

$\$$

$(q_0, 1) \to (q_3, \not{b}, R)$

$(q_0, 0) \to (q_0, \not{b}, R)$

$(q_3, 0) \to (q_3, \not{b}, R)$

$(q_3, 1) \to (q_0, \not{b}, R)$

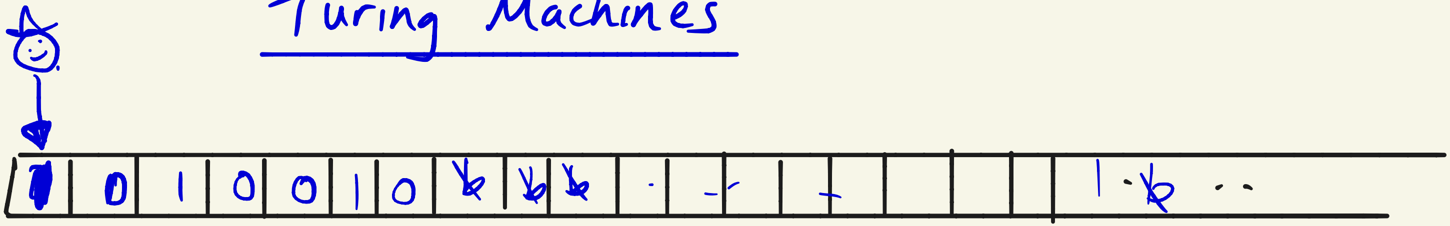
$(q_0, \not{b}) \to (q_{0,1},$

q_0: current parity is 0

q_2: current parity " 1

Turing Machines

q_0



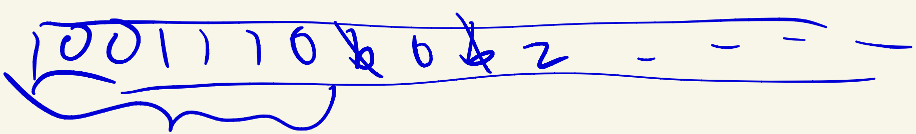
some finite alphabet

ex $\Sigma = \{0, 1, \cancel{b}\}$

finite set of states $= \{q_0, q_1, q_2\}$

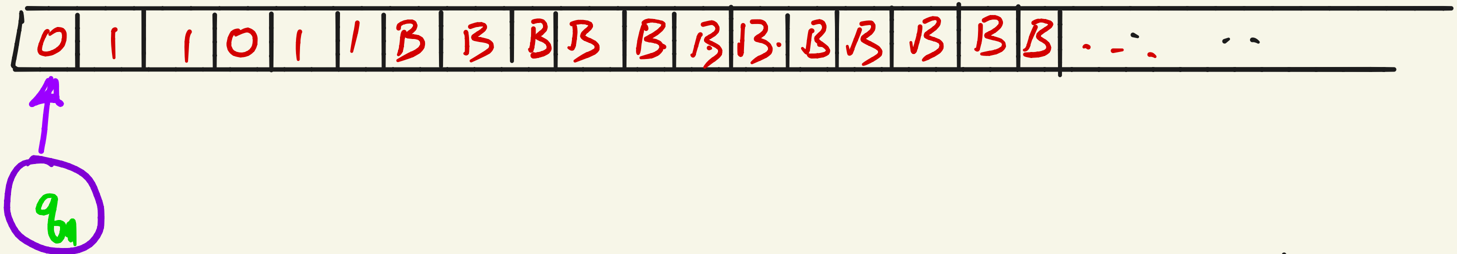
- Initially M is in start state q_1 input in 1^{st} cells, then B's
- at any point in time, tape head points to some tape cell
- every cell contains an element of Γ

$$\delta(\overset{\text{value at}}{\text{head-location}}, \text{current state}) \rightarrow (\text{new state}, \text{new symbol}, L \text{ or } R)$$



Turing Machines

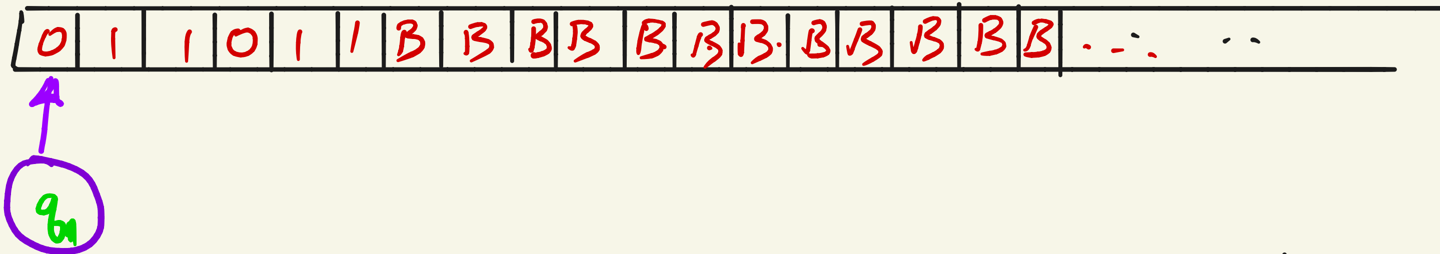
Input $x = 011011$



- Initially M is in start state q_1, input in 1^{st} cells, then B's
- at any point in time, tape head points to some tape cell
  initally head points to left most cell

Turing Machines

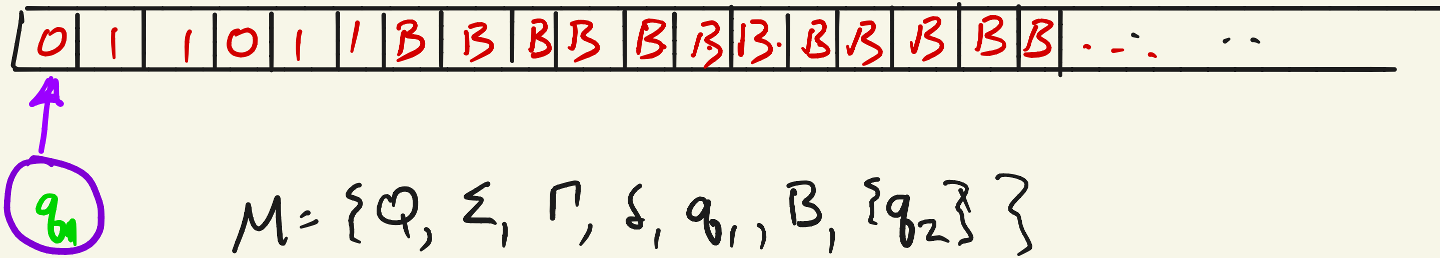
Input $x = 011011$



- Initially M is in start state q_1, input in 1st cells, then B's
- at any point in time, tape head points to some tape cell
  initally head points to left most cell
- at every time step, M makes one transition according to δ

Turing Machines

Input $x = 011011$



$$M = \{Q, \Sigma, \Gamma, \delta, q_1, B, \{q_2\}\}$$

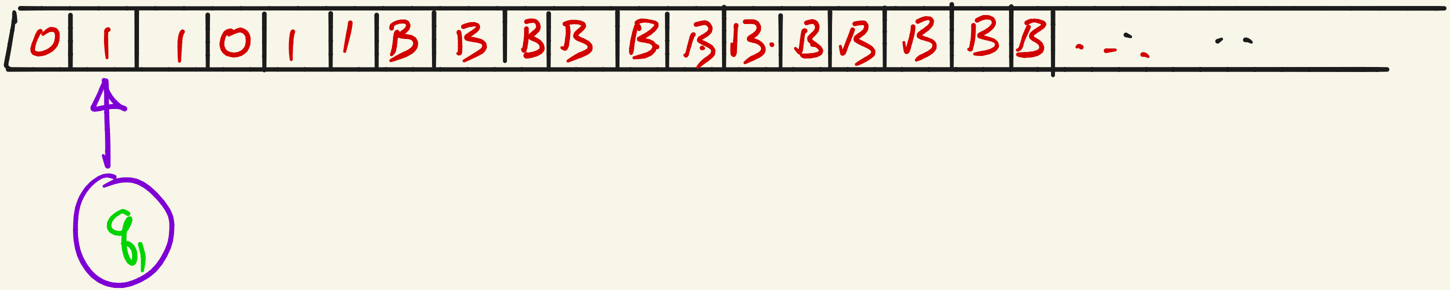
$$Q = \{q_1, q_2, q_3\}, \quad \Sigma = \{0, 1\}, \quad \Gamma = \{0, 1, B\}$$

δ:

$$(0, q_1) \rightarrow (0, q_1, R)$$
$$(1, q_1) \rightarrow (1, q_3, R)$$
$$(B, q_1) \rightarrow (B, q_1, R)$$
$$(0, q_3) \rightarrow (0, q_3, R)$$
$$(1, q_3) \rightarrow (1, q_2, R)$$
$$(B, q_3) \rightarrow (B, q_3, R)$$

Turing Machines

Input $x = 011011$

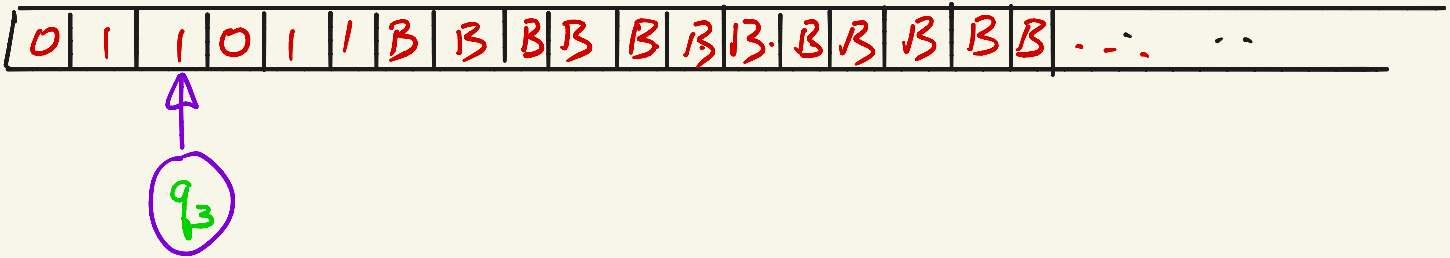


δ :

$$(0, q_1) \to (0, q_1, R)$$
$$(1, q_1) \to (1, q_3, R)$$
$$(B, q_1) \to (B, q_1, R)$$
$$(0, q_3) \to (0, q_3, R)$$
$$(1, q_3) \to (1, q_2, R)$$
$$(B, q_3) \to (B, q_3, R)$$

Turing Machines

Input $x = 011011$

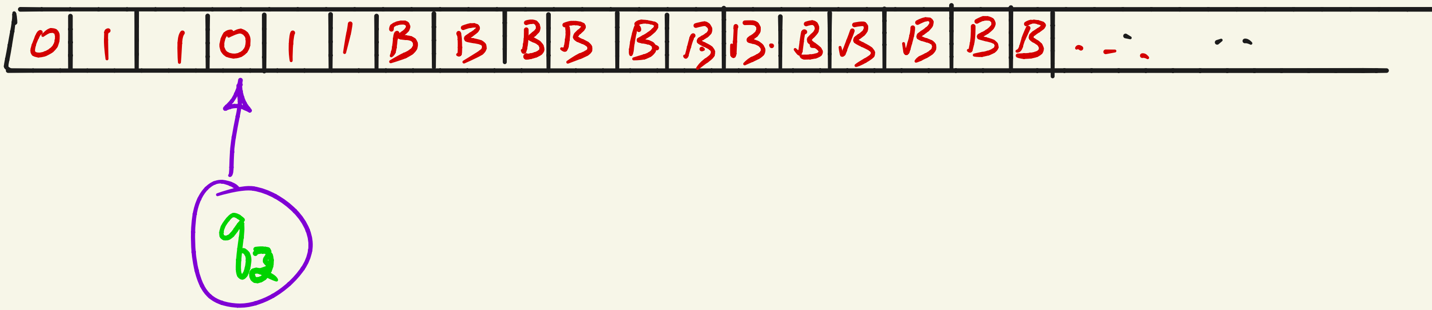


δ :

$$(0, q_1) \rightarrow (0, q_1, R)$$
$$(1, q_1) \rightarrow (1, q_3, R)$$
$$(B, q_1) \rightarrow (B, q_1, R)$$
$$(0, q_3) \rightarrow (0, q_3, R)$$
$$(1, q_3) \rightarrow (1, q_2, R)$$
$$(B, q_3) \rightarrow (B, q_3, R)$$

Turing Machines

Input $x = 011011$



δ:

$$(0, q_1) \rightarrow (0, q_1, R)$$
$$(1, q_1) \rightarrow (1, q_3, R)$$
$$(B, q_1) \rightarrow (B, q_1, R)$$
$$(0, q_3) \rightarrow (0, q_3, R)$$
$$(1, q_3) \rightarrow (1, q_2, R)$$
$$(B, q_3) \rightarrow (B, q_3, R)$$

Turing Machines

Turing Machines compute n-ary partial (or total) functions from $\mathbb{N}^n \to \mathbb{N}$ by encoding input/output as strings over Σ

Encoding of $(a_1, \dots, a_n) \in \mathbb{N}^n$ example

$$(3, 10, 8) : \underbrace{11}_{a_1 \text{ in binary}}\ 2\ \underbrace{1010}_{a_2 \text{ in binary}}\ 2\ \underbrace{100}_{a_3 \text{ in binary}}$$

separated by "2"

Let $\langle a_1, \dots, a_n \rangle$ be the encoding of $(a_1, \dots, a_n)$

Turing Machines

Turing Machines compute n-ary partial (or total) functions from $\mathbb{N}^n \to \mathbb{N}$ by encoding input/output as strings over Σ

TM M on input x halts when it enters halt state (q_2)

If M halts on x, the output y is the ~~shortest~~ longest string on tape ~~with NO B symbol~~ containing just 0's + 1's

Turing Machines

Let $f : \mathbb{N}^n \to \mathbb{N}$ be a total function

M <u>computes</u> f if for every n-tuple $(a_1 .. a_n) \in \mathbb{N}^n$
M on input $\langle a_1, .., a_n \rangle$ outputs $f(a_1 .. a_n)$
(in binary)

If there is a TM M that computes f,
then f is a <u>total</u> <u>computable</u> <u>function</u>

Turing Machines

Let $f : (\mathbb{N} \cup \{\infty\})^n \to \mathbb{N} \cup \{\infty\}$ be a partial function

(so $f(c_1 \ldots c_n) = \infty$ if any $c_i = \infty$)

M computes f if for all $(a_1 .. a_n)$ in domain of f

M on input $\langle a_1 .. a_n \rangle$ outputs $f(a_1 .. a_n)$

* M may not halt on inputs not in domain of f

If f (a partial function) is computed by some M
then f is a computable partial function

d-ary partial fxn f maps each input $x \in S$, where $S \subseteq \underbrace{\mathbb{N} \times \mathbb{N} \times \cdots \times \mathbb{N}}_{d}$

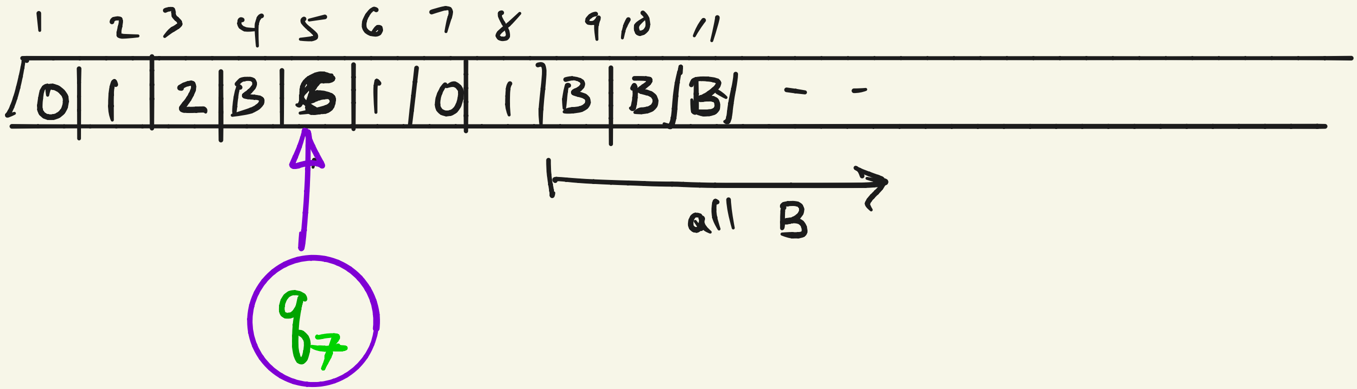
to $\mathbb{N}$

If M computes partial fxn f with domain S then

- $x \in S \Rightarrow M(\langle x \rangle)$ = binary encoding of $f(x)$
- $x \notin S \Rightarrow M(\langle x \rangle)$ either halts + outputs something or doesn't halt on x

Turing Machine Configurations

- A configuration describes entire state of a TM at some point in time



Configuration : $0, 1, 2, B, (q_7, 5), 1, 0, 1$

Turing Machine Configurations

- A tableaux is a sequence of configurations describing running M on some input x

Turing Machine Configurations

- A <u>tableaux</u> is a sequence of configurations describing running M on some input x $(0, q_1) \to (2, q_1, R)$

$t=0$	$(q_1, 0)$	0	1	1	0	2	B	. .
$t=1$	2	$(q_1, 0)$	1	1	0	2	B	. . .
$t=2$	2	2	$(q_1, 1)$	1	0	2	B	. . .
$t=3$	2	2	2	$(q_1, 1)$	0	2	B	. .
$t=4$	2	2	2	2	$(q_1, 0)$	2	B	. .
$t=5$	2	2	2	2	2	$(q_1, 2)$	B	. .
$t=6$	2	2	2	2	2	2	(q_2, B)	. .

Turing Machine Configurations

- A tableaux is a sequence of configurations describing running M on some input x

At time $t = m$, tableaux is $m \times m$

$t=0$	$(q_1, 0)$	0	1	1	0	2	B	. .
$t=1$	2	$(q_1, 0)$	1	1	0	2	B	. . .
$t=2$	2	2	$(q_1, 1)$	1	0	2	B	. . .
$t=3$	2	2	2	$(q, 1)$	0	2	B	. .
$t=4$	2	2	2	2	$(q, 0)$	2	B	. .
$t=5$	2	2	2	2	2	$(q_1, 2)$	B	. .
$t=6$	2	2	2	2	2	2	(q_2, B)	. .

Encoding Turing Machines

$M = (\Sigma, Q, \Gamma, \delta, q_1, B, \{q_2\})$

Let $\Sigma = \{0, 1, 2\}$

$Q = \{q_1, q_2, \dots q_n\}$

$\Gamma = \{X_1, X_2, \dots X_k\}$ where $X_1 = 0$ $X_2 = 1$ $X_3 = 2$ $X_4 = B$

$D_1 = \text{left}$ $D_2 = \text{right}$

We represent transition $\delta(q_i, X_j) \to (q_k, X_\ell, D_m)$ by

$$0^i 1 0^j 1 0^k 1 0^\ell 1 0^m$$

Code for M: $111 \, code_1 \, 11 \, code_2 \, 11 \dots 11 \, code_r \, 111$

where $code_1, \dots, code_r$ are the codes for transition function

Encoding Turing Machines

Example. $Q = \{q_1, q_2, q_3\}$, $\Sigma = \{0,1\}$, $\Gamma = \{0,1,B\}$

$$\delta(q_1, 1) = (q_3, 0, R) \qquad 0^1 1 0^2 1 0^3 1 0^1 1 0^2 \leftarrow c_1$$
$$\delta(q_3, 0) = (q_1, 1, R) \qquad 0^3 1 0 1 0^1 1 0^2 1 0^2 \leftarrow c_2$$
$$\delta(q_3, 1) = (q_2, 0, R) \qquad 0^3 1 0^2 1 0^2 1 0^1 1 0^2 \leftarrow c_3$$
$$\delta(q_3, B) = (q_3, 1, L) \qquad 0^3 1 0^3 1 0^3 1 0^2 1 0^1 \leftarrow c_4$$

$$M = 111\, c_1\, 11\, c_2\, 11\, c_3\, 11\, c_4\, 111$$

$$(M, 110110) \text{ encoded as } \underbrace{111\, \overbrace{c_1\, 11\, c_2\, 11\, c_3\, 11\, c_4}^{M}\, 111\, \overbrace{110110}^{x}}_{\#(M,x)}$$

* uniquely decodable

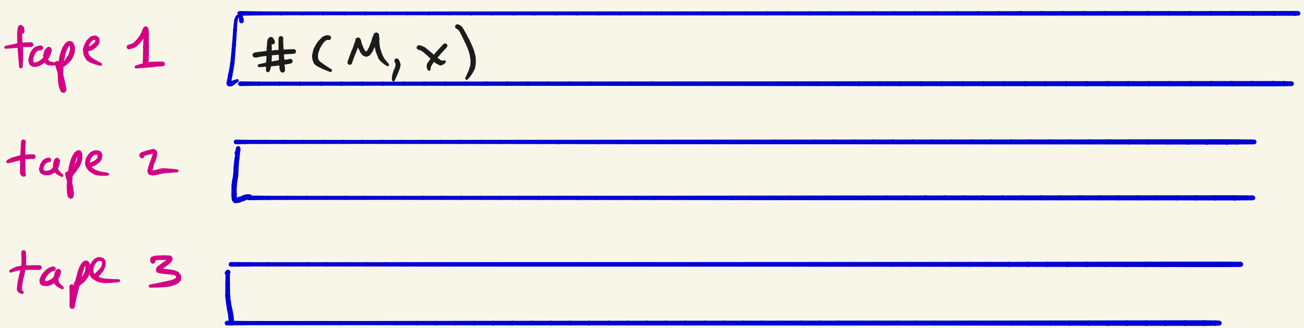
Universal Turing Machines

U: Takes as input $\#(M,x)$ and outputs y if
M on x halts and outputs y
if M does not halt on x, U does not halt on $\#(M,x)$

Universal Turing Machines

U: Takes as input $\#(M,x)$ and outputs y if
M on x halts and outputs y
if M does not halt on x, U does not halt on $\#(M,x)$

We describe a 3-tape TM (at a high level) for U.
(3-tapes can be simulated by one tape)



Universal Turing Machines

(1) initial state

tape 1 $\#(M, x)$

tape 2

tape 3

check that contents of tape 1 is legal encoding of M, x

Universal Turing Machines

2

tape 1: 11 \$ code_1 11 code_2 11 … 11 code_r 111 ← encoding of M

tape 2: \$ $\underbrace{\dot{0}110}_{x}$ ← contents of M's tape at start

tape 3: \$ 0 ← initial state of M

Initialize tapes 1 + 2 as above and tape 3 to contain \$ 0

↑ q_1 in binary

Universal Turing Machines

(2)

tape 1 | 11 \$ $code_1$ 11 $code_2$ 11 ... 11 $code_r$ 111

tape 2 | \$ x

tape 3 | \$ 0

Loop

If tape 3 contains \$00 (halt state) halt and output contents of tape 2 (to 1st "B")

OW simulate next state:

Store contents of tape 2 head and current state of M in U's state. Scan tape 1 to find corresponding code, Modify tapes 2, 3 accordingly

Universal Turing Machines

2

tape 1: $11\ \$\ code_1\ 11\ code_2\ 11\ \ldots\ 11\ code_r\ 111$

tape 2: $\$\ 0\ 0\ 1\ 2\ \dot{1}\ 0\ 1\ B\ B\ \ldots$

tape 3: $\$\ 0\ 0\ B\ B\ \ldots$

Say $\delta(q_2, 1) \rightarrow (q_3, 0, R)$

Universal Turing Machines

(2)

tape 1 $11\,\$\,code_1\,11\,code_2\,11\ldots11\,code_r\,111$

tape 2 $\$\;0\;0\;1\;2\;0\;0\;1\;B\;B\;\ldots$

tape 3 $\$\;0\;0\;0\;B\;\ldots$

Say $\delta(q_2, 1) \rightarrow (q_3, 0, R)$

Notation

$\{x\}$ = Turing machine M such that $\# M = x$

$\{x\}_1$ = the unary function computed by x

$\{x\}_n$ = the n-ary function computed by x

(can generalize earlier so M takes n inputs instead of 1)

A <u>set</u> is a subset of $\mathbb{N}^n$ (usually $n=1$)

a set / relation / 0-1 valued total function:

$$A \subseteq \mathbb{N} \text{ then } A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{if } x \notin A \end{cases}$$