Encoding TMs and TM computations as binary numbers

We want to consider problems that take a description of a TM as input, together with an input to M

To do this we have to decide on an encoding of TMs and TM computations. Next we describe a particular encoding (but there are many possible encodings)

Lecture 16

- HW3 out on Monday Nov 6 (Due Nov 20)

- Today:

  (1) Universal TMs

  (2) Closure Properties of Recognizable / Decidable Languages

  (3) Proving some Languages are <u>not</u> recognizable/decidable
  - Diagonalization
  - Turing Machine Reductions

Encoding Turing Machines

Let $M = (\Sigma, Q, \Gamma, \delta, q_1, B, q_2, q_3)$

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

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

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

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

* Note: we always assume without loss of generality that q_1 = start state, q_2 = halt and accept, q_3 = halt and reject state and $\Sigma = \{0,1,2\}$.

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: $111\ \text{code}_1\ 11\ \text{code}_2\ 11 \dots 11\ \text{code}_r\ 111$

where $\text{code}_1, \dots, \text{code}_r$ are the codes for transition function

Encoding Turing Machines

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

$$\delta(q_1, 1) = (q_3, 0, R) \qquad 0^1 1 0^2 1 0^3 1 0^1 1 0^2 \leftarrow c_1$$

$$\delta(q_3, 0) = (q_1, 1, R) \qquad 0^3 1 0 1 0^1 1 0^2 1 0^2 \leftarrow c_2$$

$$\delta(q_3, 1) = (q_2, 0, R) \qquad 0^3 1 0^2 1 0^2 1 0^1 1 0^2 \leftarrow c_3$$

$$\delta(q_3, B) = (q_3, 1, L) \qquad 0^3 1 0^3 1 0^3 1 0^2 1 0^1 \leftarrow c_4$$

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

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

* uniquely decodable

Universal Turing Machines

Defn. $A_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$

Theorem A_{TM} is recognizable/r.e.

Pf We describe a universal TM U

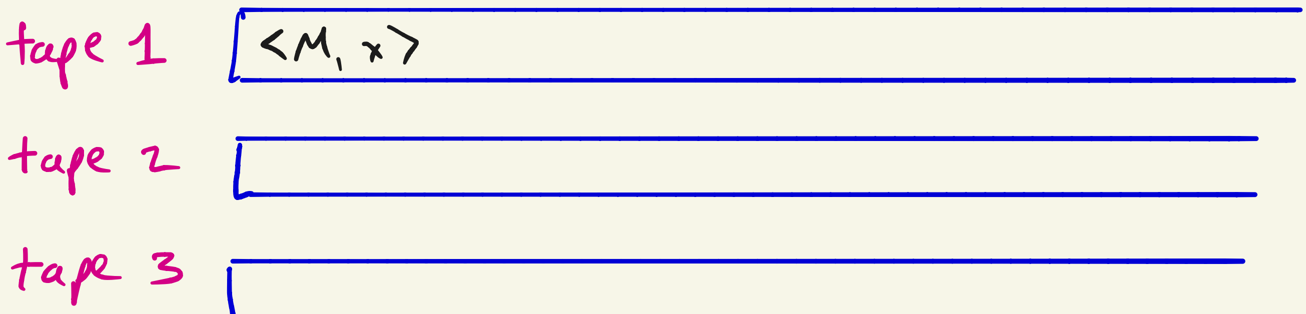
U: Takes as input $\langle M, x\rangle$

- U halts and accepts $\langle M, x\rangle$ if M halts and accepts x
- U halts and rejects $\langle M, x\rangle$ if M halts and rejects x
- U gets into infinite loop on $\langle M, x\rangle$ if M gets into infinite loop on input x

Universal Turing Machines

We describe a 3-tape TM (at a high level) for $\mathcal{U}$.

(3-tapes can be simulated by one tape)



Universal Turing Machine $\mathcal{U}$ on $\langle M, x\rangle$

① initial state

tape 1 $11\$\ code_1\ 11\ code_2\ 11\ \ldots\ 11\ code_r\ 111\ x_1\ x_2\ \ldots\ x_n$ ← encoding of $\langle M, x\rangle$

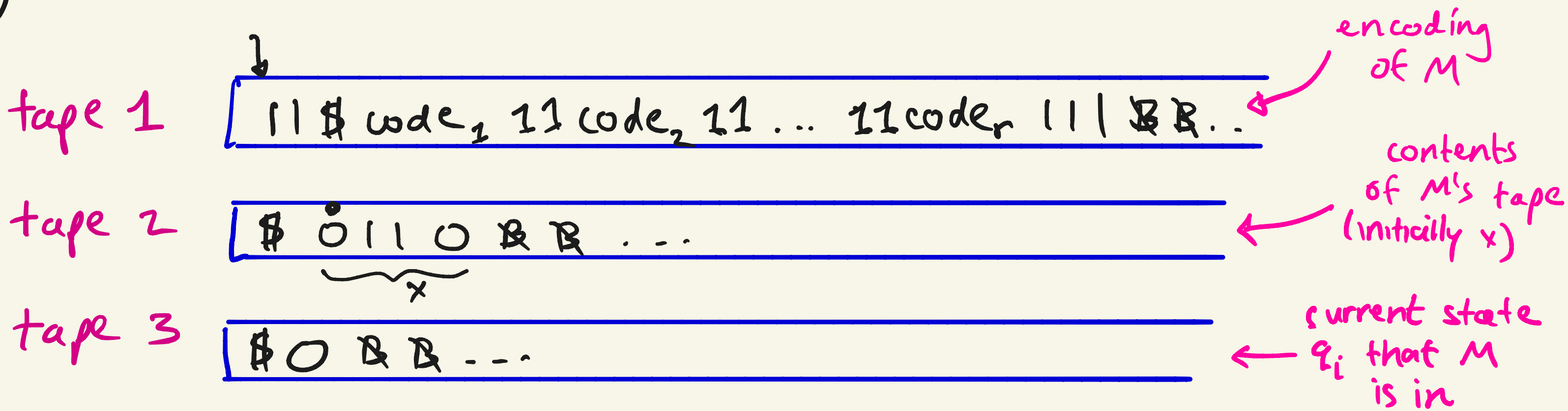
tape 2 $\not{B}\ \not{B}\ \ldots$

tape 3 $\not{B}\ \not{B}\ \ldots$

- check that contents of tape 1 is a legal encoding $\langle M, x\rangle$

Universal Turing Machines

(2)



- Initialize tape 1 to contain $\langle M \rangle$,
  tape 2 to contain $\$x$
  tape 3 to contain $\$q_1$ (start state of M)

Universal Turing Machines

(3)

tape 1 | $11 \, \$ \, code_1 \, 11 \, code_2 \, 11 \ldots 11 \, code_r \, 111$

tape 2 | $\$ \, x_1 \, x_2 \ldots x_n$

tape 3 | $\$ \, q_1$

Loop

If tape 3 contains $\$ q_2$ (accept state) halt and accept

If tape 3 contains $\$ q_3$ (halt + reject state) halt and reject

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

(3) Example (of Loop): Say $\delta(q_4, 1) \rightarrow \delta(q_5, 0, R)$

tape 1	1 1 \$ code$_1$ 1 1 code$_2$ 1 1 ... 1 1 code$_r$ 1 1 1 $\not B$ ...
tape 2	\$ 0 0 1 2 1 0 1 $\not B$ $\not B$ ...
tape 3	\$ 0 0 0 0 $\not B$ $\not B$ ..

$$\Downarrow$$

tape 1	1 1 \$ code$_1$ 1 1 code$_2$ 1 1 ... 1 1 code$_r$ 1 1 1
tape 2	\$ 0 0 1 2 0 0 1 $\not B$ $\not B$ ...
tape 3	\$ 0 0 0 0 0 $\not B$ $\not B$ ..

CLOSURE PROPERTIES

1. L recursive $\Rightarrow$ L r.e.

2. closure of recursive languages under $\cap, \cup, \neg$:

   L_1, L_2 recursive $\Rightarrow$ $L_1 \cup L_2$, $L_1 \cap L_2$, $\neg L_1$, $\neg L_2$ are recursive

3. Closure of r.e. languages under $\cap, \cup$

   L_1, L_2 r.e. $\Rightarrow$ $L_1 \cup L_2$, $L_1 \cap L_2$ recursive

What about closure of r.e. under $\neg$?

4. L is r.e. and $\bar{L}$ is r.e. $\Rightarrow$ L is recursive

We'll sketch proof of 4; rest are left as exercises

CLOSURE PROPERTIES

④ L r.e., and $\bar{L}$ r.e. $\Rightarrow$ L is recursive

Proof sketch: (Dovetailing)

Let M_1 be a TM st $\mathcal{L}(M) = L$ and let M_2 be a TM st $\mathcal{L}(M) = \bar{L}$

New TM M on x:

(1) For $i = 1, 2, 3, \ldots$

(2) Run M_1 on x for i steps

(3) if M_1 accepts x, halt + accept

(4) Run M_2 on x for i steps

(5) if M_2 accepts x, halt and reject

M_1 + M_2 may not always halt.
We want to design M that always halts and such that $\mathcal{L}(M) = L$

CLOSURE PROPERTIES

④ L r.e., and $\overline{L}$ r.e. $\Rightarrow$ L is recursive

Proof sketch: (Dovetailing)

Let M_1 be a TM st $\mathcal{L}(M) = L$ and let M_2 be a TM st $\mathcal{L}(M) = \overline{L}$

New TM M on x:

(1) For $i = 1, 2, 3, \ldots$

(2) Run M_1 on x for i steps

(3) if M_1 accepts x, halt + accept

(4) Run M_2 on x for i steps

(5) if M_2 accepts x, halt and reject

M_1 + M_2 may Not always halt.
We want to design M that always halts and such that $\mathcal{L}(M) = L$

Claim: M always halts and $\mathcal{L}(M) = L$:

$\forall x$ exactly one of $M_1(x)$ and $M_2(x)$ halts and accepts.

$\therefore$ $\forall x$ there is some time step i s.t. either (i) $M_1(x)$ halts and accepts or (ii) $M_2(x)$ halts and accepts

If (i) then $x \in L$ and $M(x)$ halts + accepts (line 3)

If (ii) then $x \notin L$ and $M(x)$ halts + rejects (line 5)

Computability

Q: what problems are Turing decidable?

Q: Can we decide if a given program halts on all inputs?

Motivating Example: Does this Process Terminate?

Hailstone (n) input $n > 0$

Repeat:

If $n = 1$ halt

Else if n is even set $n = \frac{n}{2}$

Else set $n = 3n + 1$

Motivating Example: **Does this Process Terminate?**

Hailstone (n) input $n > 0$

Open Problem!

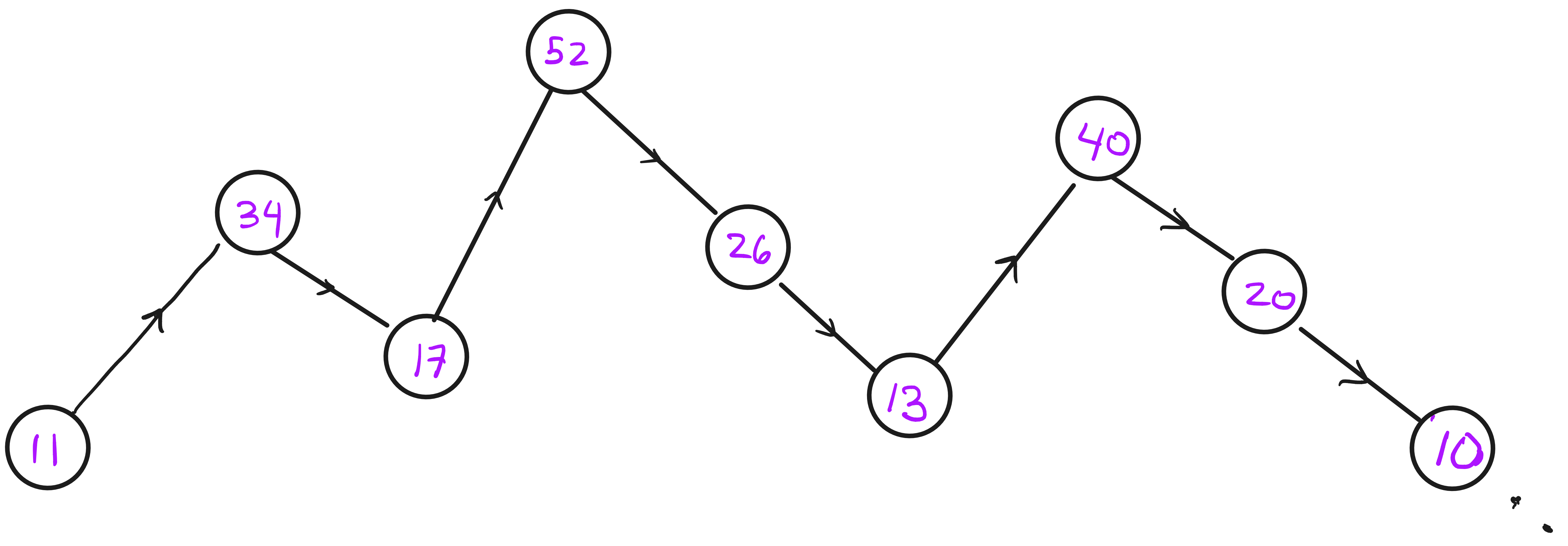
<u>Repeat:</u>

If $n = 1$ halt

Else if n is even set $n = \frac{n}{2}$

Else set $n = 3n+1$

$n = 11$



Many Languages are Not r.e. (recognizable)!

Idea: show there are way more languages than r.e. languages.

To compare sizes of infinite sets we use the notion of countable vs uncountable.

Defn A set S is countable if there is a 1-1 mapping from $S \to \mathbb{N}$

Many Languages are Not r.e. (recognizable)!

Idea: show there are way more languages than r.e. languages.

To compare sizes of infinite sets we use the notion of countable vs uncountable.

Defn A set S is countable if there is a 1-1 mapping from $S \to \mathbb{N}$

Examples

1. Any finite set S is countable
2. The set of all pairs $(i,j) \in \mathbb{N} \times \mathbb{N}$ is countable

Many Languages are Not r.e. (recognizable)!

Defn A set S is countable if there is a 1-1 mapping from $S \to \mathbb{N}$

How to show that some (infinite) set is Not countable?

Proof Diagonalization argument

(similar to Cantor's argument showing that the set of all real numbers is uncountable by showing there is No 1-1 map from $\underbrace{\mathbb{R}}_{\text{reals}} \Rightarrow \mathbb{N}$)

Example The set of all real numbers in $[0,1]$ is not countable.

Suppose (for contradiction) $\exists$ a 1-1 mapping f from $\mathbb{R} \Rightarrow \mathbb{N}$:

i	$f(i)$
0	.0 1 7 4 3 9 8 ...
1	.0 0 2 5 6 9 0 2 2 ...
2	.1 1 8 4 0 0 ...
3	.1 1 2 3 ...
⋮	

Construct real number $x = .x_1 x_2 x_3 \ldots$ such that $x_i \neq f(i)_i$

e.g. Let $x_i = \begin{cases} 0 & \text{if } f(i)_i \neq 0 \\ 1 & \text{if } f(i)_i = 0 \end{cases}$

$x = .1100\ldots$

Since x not any row of table (by construction), f is not a 1-1 mapping. #

Many Languages are not r.e. (recognizable)!

Proof Idea Let $\Sigma = \{0,1\}$

- Every TM over $\Sigma = \{0,1\}$ is encoded by a unique string $\langle M \rangle \in \Sigma^*$
- Thus every Turing recognizable language over $\{0,1\}$ can be described/encoded by a string $\langle M \rangle \in \Sigma^*$ (the M that accepts L)
- Thus the set of all Turing-recognizable languages is countable
- But on the other hand the set of all languages over $\{0,1\}$ is uncountable
- Thus most languages $L \subseteq \Sigma^*$ are not recognizable

__Theorem__ There exists a Language $L \subseteq \{0,1\}^*$ that is not r.e. (recognizable)

__Pf__ (diagonalization)

Fix an enumeration of all TMs over $\{0,1\}$ using our encoding of TMs

$M_1, M_2, M_3, \ldots$

order lexiographically by their encodings (so $\langle M_1 \rangle < \langle M_2 \rangle < \ldots$)

Define $D = \{ \langle M \rangle \mid \langle M \rangle$ encodes TM M, and M on input $\langle M \rangle$ does not halt and accept $\}$

called the Diagonal Language

Define $D = \{ \langle M \rangle \mid \langle M \rangle$ encodes TM M, and M on input $\langle M \rangle$ does not halt and accept $\}$

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	. . . . . .
M_1	0	1	0	0	. .
M_2	1	0	1	1	. .
M_3	0	1	1	1	
M_4	0	1	0	1	
⋮					

entry $(M_i, \langle M_j \rangle)$: $\begin{cases} 1 & \text{if } M_i(\langle M_j \rangle) \text{ halts and accepts} \\ 0 & \text{if } M_i(\langle M_j \rangle) \text{ either halts and rejects or never halts} \end{cases}$

Define $D = \{ \langle M \rangle \mid \langle M \rangle$ encodes TM M, and M on input $\langle M \rangle$ does not halt and accept $\}$

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	. . . . . .
M_1	0	1	0	0	. .
M_2	1	0	1	1	. .
M_3	0	1	1	1	
M_4	0	1	0	1	
⋮					

D is the "opposite" of what is on diagonal

That is : $D(M_i, \langle M_i \rangle) = \begin{cases} 0 & \text{if } M_i(\langle M_i \rangle) \text{ halts and accepts} \\ 1 & \text{otherwise} \end{cases}$

Define $D = \{ \langle M \rangle \mid \langle M \rangle$ encodes TM M, and M on input $\langle M \rangle$ does not halt and accept $\}$

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	1	1	0	0	$\cdots$
M_2	1	1	1	1	$\cdots$
M_3	0	1	0	1	
M_4	0	1	0	0	
$\vdots$					

D is the "opposite" of what is on diagonal

That is: $D(\langle M_i \rangle) = \begin{cases} 0 & \text{if } M_i(\langle M_i \rangle) \text{ halts and accepts} \\ 1 & \text{otherwise} \end{cases}$

<u>Theorem</u> There exists a Language $L \subseteq \{0,1\}^*$ that is not r.e. (recognizable)

<u>Pf</u> (diagonalization)

Fix an enumeration of all TMs over $\{0,1\}$ using our encoding of TMs

$M_1, M_2, M_3, \ldots$

order lexicographically by their encodings (so $\langle M_1 \rangle < \langle M_2 \rangle < \ldots$)

Define $D = \{ \langle M \rangle \mid \langle M \rangle$ encodes TM M, and M on input $\langle M \rangle$ does not halt and accept $\}$

called the Diagonal Language

<u>Claim</u> D is not r.e. (recognizable)

<u>Pf</u> By construction, for all TMs M_i over Σ,

$\mathcal{L}(M_i) \neq D$ since $\langle M_i \rangle \in D$ if and only if $\langle M_i \rangle \notin \mathcal{L}(M_i)$

Define $\overline{D} = \{ \langle M \rangle \mid \langle M \rangle$ encodes TM M, and M on input $\langle M \rangle$ accepts $\}$

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	0	1	0	0	$\cdots$
M_2	1	0	1	1	$\cdots$
M_3	0	1	1	1	
M_4	0	1	0	1	
$\vdots$					

$\overline{D}$ is what is on the diagonal

That is $\overline{D}(\langle M_i \rangle) = \begin{cases} 1 & \text{if } M_i(\langle M_i \rangle) \text{ accepts} \\ 0 & \text{otherwise} \end{cases}$

Claim $\overline{D}$ is recognizable / r.e.

Pf : TM for $\overline{D}$ on input $\langle M \rangle$

- Check to see if input is legal encoding of a TM
  if not, reject
- otherwise run M on $\langle M \rangle$:
  If simulation halts and accepts $\rightarrow$ halt & accept

Thus we have shown:

$\overline{D}$ is r.e.

D is not r.e.

Question: Is $\overline{D}$ recursive / decidable?