CS 205
Lecture 33
7/4/21

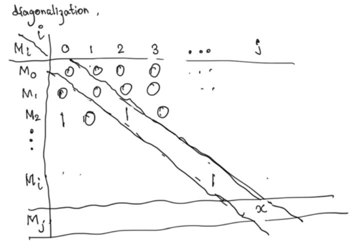
Encoding of TMs (as strings over $\{0,1\}$)

Features of the encoding

1. Each TM is mapped to a natural number.
2. Can also associate a TM with each nat. no.
   - if the binary representation of n does not correspond to a valid TM code, then we associate with n the one-state TM which halts w/o accepting any input
3. For a TM M, $\langle M \rangle$ = nat. no. that encodes M
   For $i \in \mathbb{N}$, M_i = TM corresponding to the number i

Thm : The diagonal language $\boxed{L_d = \{ i \mid i \notin L(M_i) \}}$ (where M_i is a TM with tape alphabet $\{0,1,B\}$) is not r.e.

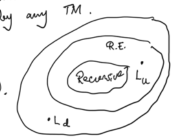
Proof : By diagonalization,



Claim L_d is not recognized by any TM. Why?
Suppose $L_d = L(M)$ for TM M, and $\langle M \rangle = j$.
Q : Does M_j accept j ? (Remember $L(M_j) = L_d$ by assumption)
If yes, then $j \in L_d$ so $j \notin L(M_j)$ (by defn. of L_d)
$\Rightarrow j \notin L_d$ (since $L_d = L(M_j)$)
Contradiction.

If no, then $j \notin L_d$ so $j \in L(M_j)$ (by defn. of L_d)
$\Rightarrow j \in L_d$ (since $L_d = L(M_j)$)
Contradiction.

Therefore, L_d cannot be accepted by any TM.
Hence L_d is not r.e.

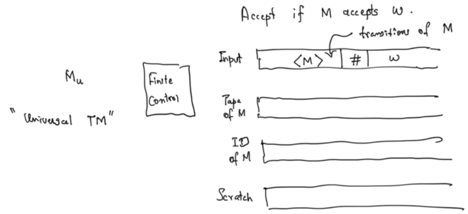


L_d is an undecidable problem (not recursive).

Q : Are there r.e. languages that are not recursive.
A: $\boxed{L_u = \{ \langle M \rangle \# w \mid M \text{ accepts } w \}}$

Claim : L_u is r.e.

Proof : TM M_u accepting L_u :
On input $\langle M \rangle \# w$
M_u generates the description of M
Simulate M on w
Accept if M accepts w.



Thm : L_u is not recursive.

Proof : We will show if L_u is recursive then L_d can be recognized by a TM, which is not true.

Suppose L_u is recursive. Then there is a TM N which on input $\langle M \rangle \# w$ always halts and answers yes if $w \in L(M)$ and no if $w \notin L(M)$.

Here is a TM M_d for L_d :
On input w
M_d simulates N on $w \# w$ and accepts if N rejects and rejects if N accepts

M_d accepts w iff N rejects $w \# w$ iff $w \notin M_w$
$\Leftrightarrow w \in L_d$

i.e. $L_d \in L(M_d)$ and M_d is a TM that halts on all inputs.

Contradiction, since L_d is not r.e.
Hence L_u is not recursive.