Church-Turing thesis and universal computation

2/3/21 10:43 AM

Church-Turing Thesis. [1936]

Any problems solvable by machines can be solved by TMs.

Q. what are the next big-picture questions?

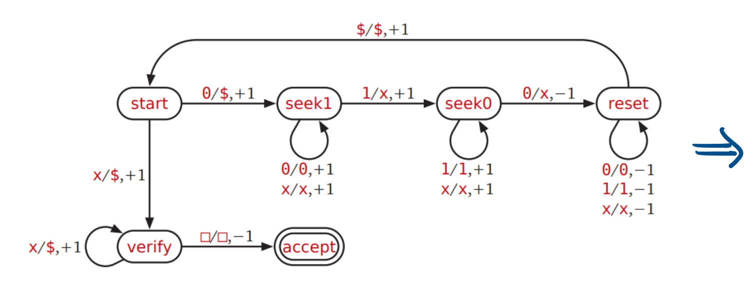
- Hardest problem? complexity.
- Can all problems be solved?
- Can TM emulates human brain?

Encoding TM. $\{0, 1, \$, x, \square\}$

Let $M = (\Gamma, Q, \text{start}, \text{accept}, \text{reject}, \delta)$ be an arbitrary TM.

Construct encoding over $\{0, 1, [,], \bullet, |, \underset{\blacktriangle}{0}, \underset{\blacktriangle}{!}\}$

- $\langle 0 \rangle = 001$
  $\langle 1 \rangle = 010$ $\quad \langle 001100 \rangle = [001 \bullet 001 \bullet 010 \bullet 010 \bullet 001 \bullet 001]$
  $\langle \$ \rangle = 011$
  $\langle x \rangle = 100$
  $\langle \square \rangle = 000$
- $\langle \text{start} \rangle = 111$
  ⋮
  $\langle \text{accept} \rangle = 110$
  $\langle \text{reject} \rangle = 000$
- $\langle \delta \rangle :=$ concatenation of $[\langle p \rangle \bullet \langle a \rangle \,|\, \langle q \rangle \bullet \langle b \rangle \bullet \langle \Delta \rangle]$
  for each transition $p \xrightarrow{a/b,\Delta} q$
- $\langle M \rangle := [\langle \text{reject} \rangle \bullet \langle \square \rangle][\langle \delta \rangle]$



$\Rightarrow$

```
[000•000][[001•001|010•011•1][001•100|101•011•1]
         [010•001|010•001•1][010•100|010•100•1]
         [010•010|011•100•1][011•010|011•010•1]
         [011•100|011•100•1][011•001|100•100•1]
         [100•001|100•001•0][100•010|100•010•0]
         [100•100|100•100•0][100•011|001•011•1]
         [101•100|101•011•1][101•000|110•000•0]]
```

Universal TM: Take $\langle M \rangle$ & $\langle w \rangle$ and run $M(w)$.

- encode configuration (curr. state + tape info):

(start, $\underset{\blacktriangle}{0}01100$) $[111]$
$[001 \bullet 001 \bullet 010 \bullet 010 \bullet 001 \bullet 001]$

(reset, $\$0x\underset{\blacktriangle}{1}x0$) $[100]$
$[001 \bullet 001 \bullet 010 \bullet 010 \bullet 001 \bullet 001]$

- UTM has an input tape to store $\langle M \rangle, \langle w \rangle$
  data tape, blank at first.
  state tape, "

- initialization: write $\underset{\text{start}}{\overset{111}{\langle \text{start} \rangle}}$ on state tape.
  write $\langle w \rangle$ on data tape, replace 0 w/ $\underset{\blacktriangle}{0}$
  $\langle 001100 \rangle = [\underset{\blacktriangle}{0}01 \bullet 001 \bullet 010 \bullet 010 \bullet 001 \bullet 001]$ (0 0 1 1 0 0)

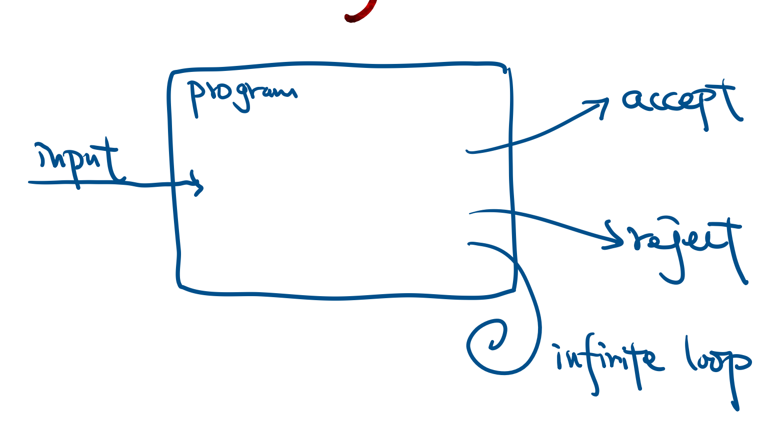
- each step of $M(w)$:
  1. find $\underset{\blacktriangle}{0}$ or $\underset{\blacktriangle}{!}$ in data tape
  2. scan through input tape to find $[\langle p \rangle \bullet \langle a \rangle \,|\, \langle q \rangle \bullet \langle b \rangle \bullet \langle \Delta \rangle]$
     s.t. $\langle p \rangle$ is on state tape
     $\langle a \rangle$ is the word containing $\underset{\blacktriangle}{0}$ or $\underset{\blacktriangle}{!}$
  3. replace $\langle p \rangle$ w/ $\langle q \rangle$ on state tape.
     $\langle \underset{\blacktriangle}{a} \rangle$ w/ $\langle b \rangle$ on work tape.
  4. move to next word on data tape, add $\blacktriangle$
  5. accept / reject when ever state tape says so. ∎

Corollary. Write $\langle M \rangle$ in TM design is safe!

code is data.

```
UTM(<M>,<w>):
    return M(w)
```

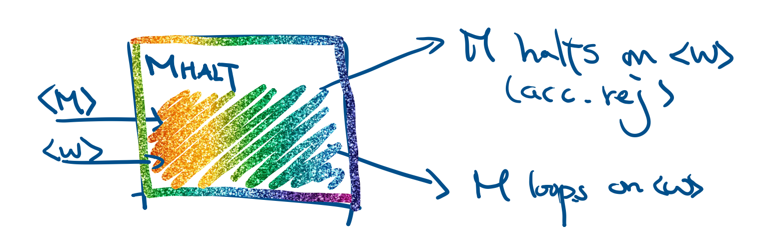
You have no idea what we have unleashed. BWAHAHAHA.

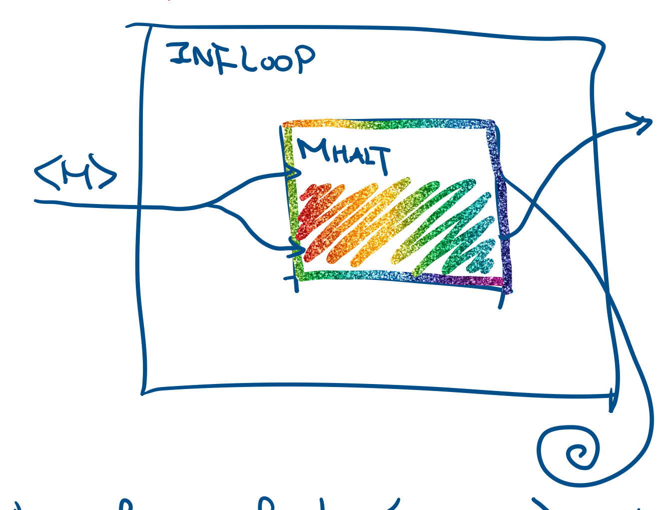


HALT($\langle M \rangle$, $\langle w \rangle$): Does M halt on w ?

Q. Is HALT computable?

Say TM $\boxed{M_{HALT}}$ solves HALT

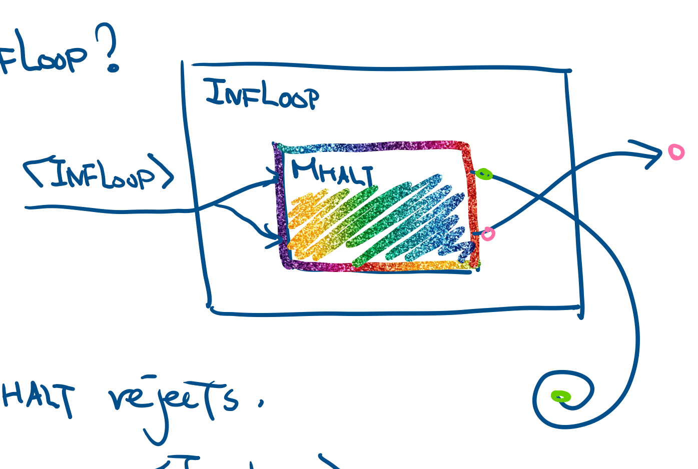




INFLOOP run $M_{HALT}(\langle M \rangle, \langle M \rangle)$

need UTM!

Q. What if we feed $\langle$INFLOOP$\rangle$ into INFLOOP?



- If INFLOOP($\langle$INFLOOP$\rangle$) halts, M_{HALT} rejects.
  $\Rightarrow$ By def. of M_{HALT}, INFLOOP loops on $\langle$INFLOOP$\rangle$.
- If INFLOOP($\langle$INFLOOP$\rangle$) loops, M_{HALT} accepts
  $\Rightarrow$ By def. of M_{HALT}, INFLOOP halts on $\langle$INFLOOP$\rangle$. $\quad \rightarrow\!\leftarrow$.

Conclusion. $\boxed{M_{HALT}}$ does not exist! HALT uncomputable.

```
:
If M(w) halts:
    :
```

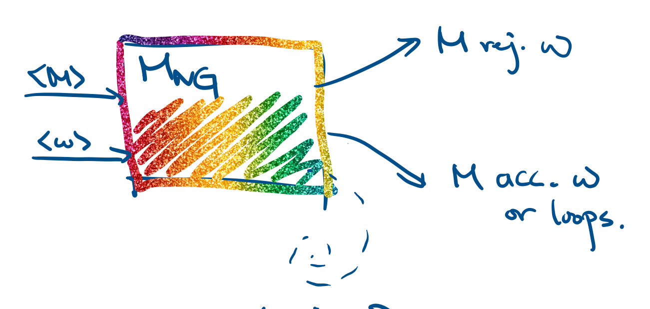
not all pseudocodes are safe!
at least not for arbitrary program M.

Another example:

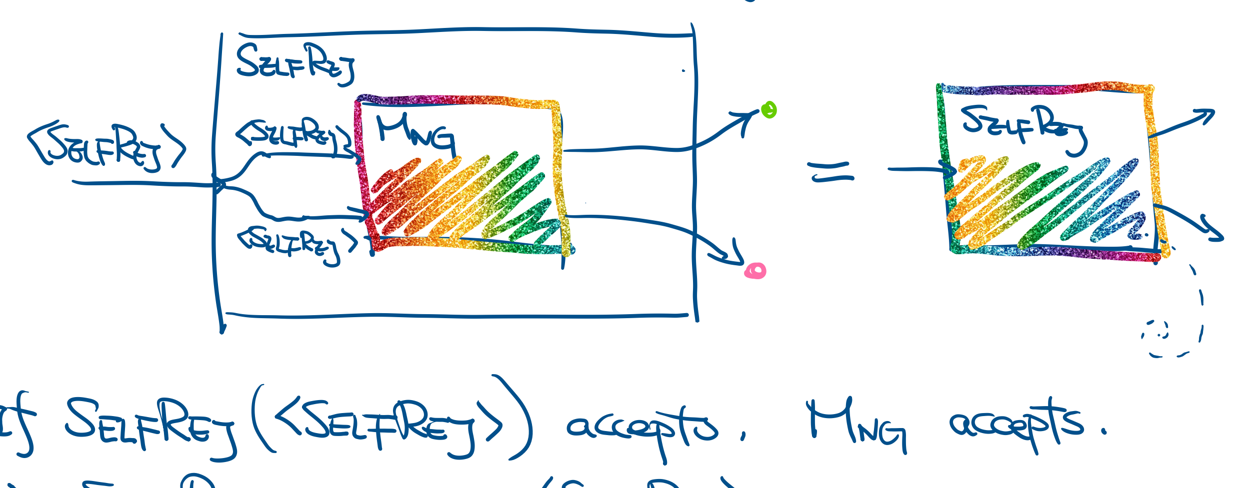
NEVERGONNAGIVEYOUUP $:= \{\langle M \rangle, \langle w \rangle : M \text{ rejects input } w\}$

Claim. NG is uncomputable.

Pf. Assume $\boxed{M_{NG}}$ exists.



SELFREJ($\langle M \rangle$): Does M rejects $\langle M \rangle$?

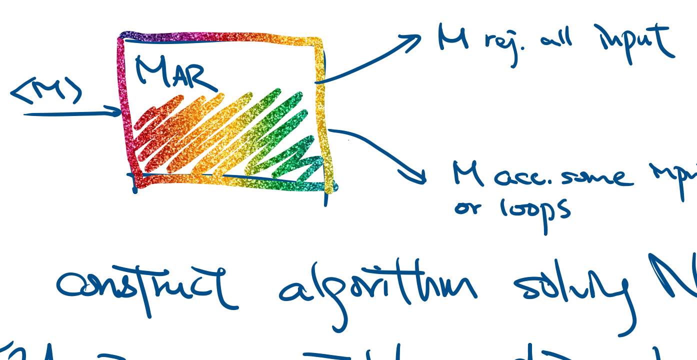


- If SELFREJ($\langle$SELFREJ$\rangle$) accepts, M_{NG} accepts.
  $\Rightarrow$ SELFREJ rejects $\langle$SELFREJ$\rangle$.
- If SELFREJ($\langle$SELFREJ$\rangle$) rejects, M_{NG} rejects.
  $\Rightarrow$ SELFREJ accepts $\langle$SELFREJ$\rangle$, or loops.
  but SELFREJ loops only if M_{NG} loops.
  by assumption that M_{NG} solves NEVERGONNAGIVEYOUUP,
  M_{NG} never loops. $\quad \rightarrow\!\leftarrow$.

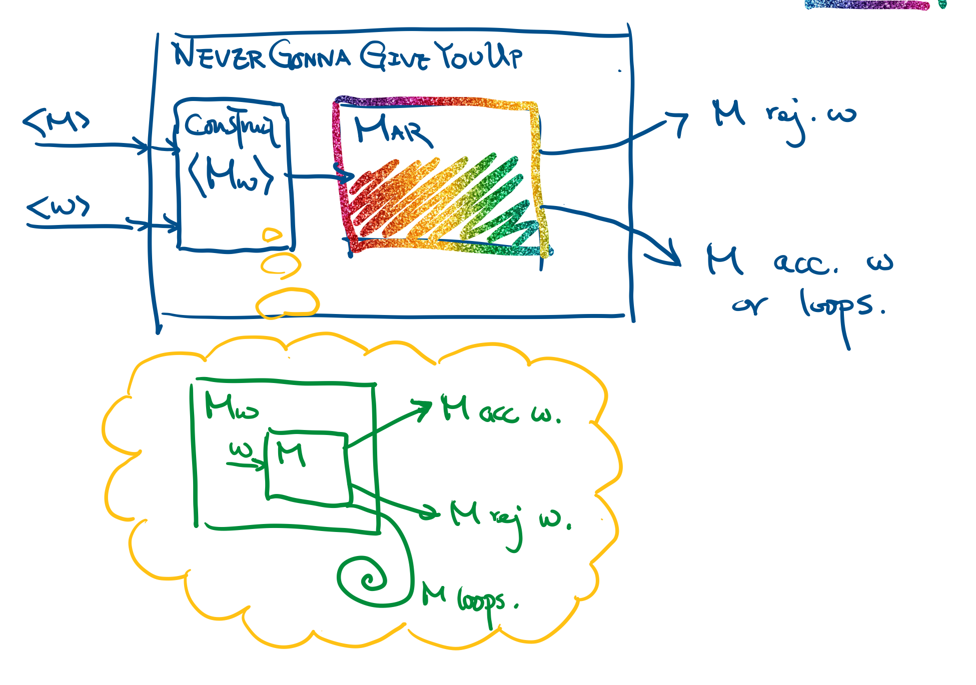
Example. ALWAYSREJ $:= \{\langle M \rangle : M \text{ always rej. any input}\}$

Claim. ALWAYSREJ is uncomputable.

Pf. Assume $\boxed{M_{AR}}$ exists.



Now we use $\boxed{M_{AR}}$ to construct algorithm solving NGGYU.
Since we proved NGGYU is uncomputable, this shows
$\boxed{M_{AR}}$ is merely a dream.



Because M_w take no input (by construction),
M_w rej. all input $\Leftrightarrow$ M rej. w
M_w acc. some input $\Leftrightarrow$ M acc. w.