$(\text{checknextblank}, 1) \to (\text{finalcleaning}, \#, R)$

$(\text{finalcleaning}, \#) \to (\text{finalmove}, 1, L)$

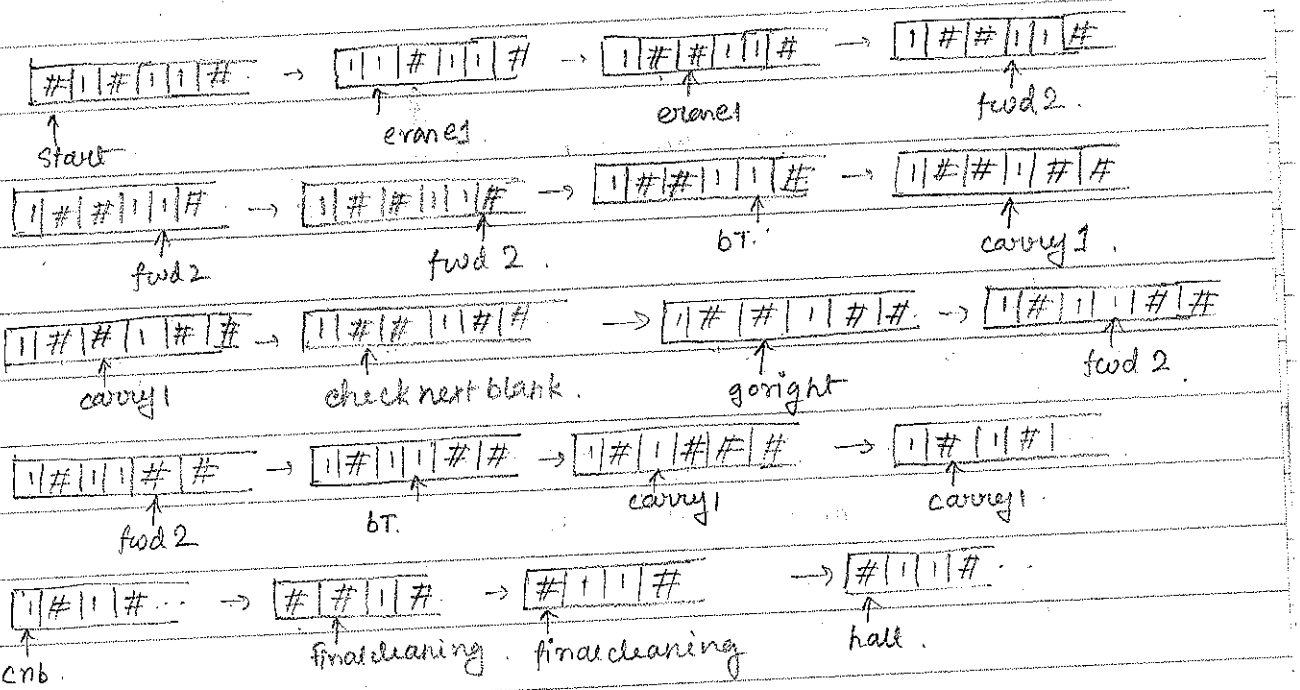
$(\text{finalmove}, \#) \to \text{halt}$

[order of these rules doesn't matter]

[h/w : Π_2^3]

$\{ (\text{checknextblank}, \#) \to (\text{goRight}, R)$

$(\text{goRight}, \#) \to (1, R, \text{fwd}2)$



cnb = checknextblank

Thursday 19.2.09

~~we started with TM~~

$n \to n+1$. $\boxed{\#|1|1|\#}$

$(\text{start}_f, \#) \to (\text{right}_f, R)$

$(\text{right}_f, 1) \to (\text{right}_f, R)$

$(\text{right}_f, \#) \to (\text{back}_f, 1, L)$

$(\text{back}_f, \#) \to \text{halt}_f$.

$(\text{back}_f, 1) \to (\text{back}_f, L)$

$n \to n+2$.

$(\text{start}_g, \#) \to (\text{right}_g, R)$

$(\text{right}_g, 1) \to (\text{right}_g, R)$

$(\text{right}_g, \#) \to (\text{put 1st 1}_g, 1, R)$

$(\text{put 1st 1}_g, \#) \to (\text{back}_g, 1, L)$.

$(\text{back}_g, 1) \to (\text{back}_g, L)$

$(\text{back}_g, \#) \to \text{halt}_g$.

~~a TM for computing~~

How to implement composition of both ?

$g : n \to n+2$

TM to compute.

$h(n) = g(f(n))$ Idea: first run TM for f, then TM for g.

why cannot we just combine the roles.

① we have diff conclusions for same state & same symbol.

② Same name may be used in both TM for slightly diff. things.

③ we can't have 'halt' after the first TM.

Solution :- rename the states. → ② & ① solved.

for ③, $halt_f$ would be replaced by $(start_g, \#)$.

So, Now, 'start' state is $(start_f, \#)$.

$halt_f$ will be $(start_g, \#)$.

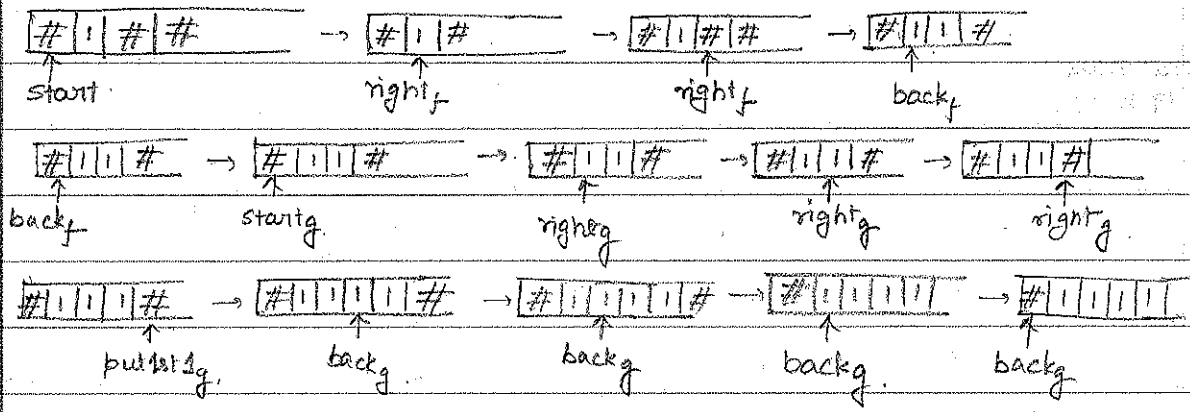
$halt_g$ will be actually 'halt' for TM.

HW :-

① Take any two T.M & design a composition.

② Prove that if set A & B decidable, then the A∩B decidable.

Now, gof :- for, $f: n \to n+1$, $g: n \to n+2$.



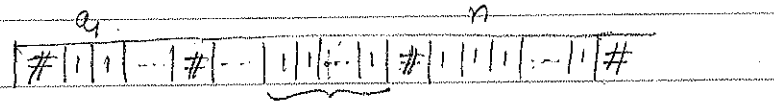
[PR is a formalization of for loop.]

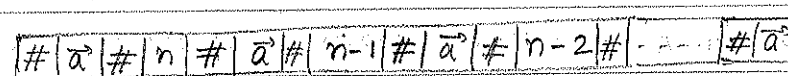
TM for PR function :-

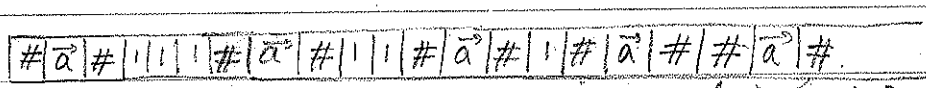
$h = PR(f, g)$, $\rightarrow$ $h = f$.

for($i=0$; $i<n$; $i++$) { $h = g(x, f(x))$

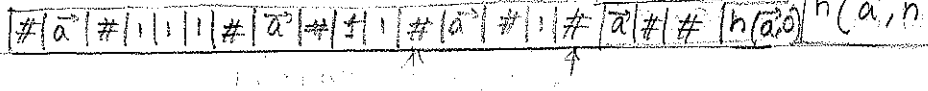
Here we need to have recursion in order to have PR computed by TM

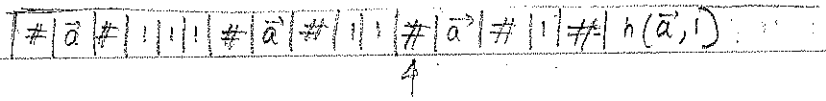




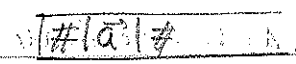


In general

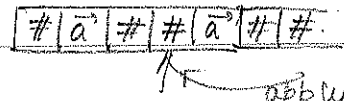


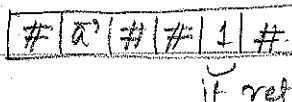


$f(\vec{a}) = \mu m . P(\vec{a}, m)$



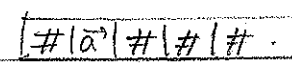
Step 1: we copy $\vec{a}$.



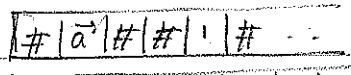


if returns true then $m=0$. (we return it by projection)

if returns false



① increase m by 1



So far, we had 2 languages:

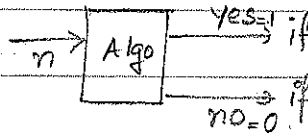
- recursive fn
- TM: bitwise.

here is another operation: sets. how sets can be related to computer.

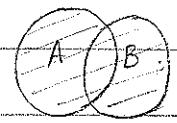
$\mathbb{N} = \{0, 1, 2, 3 \dots\}$

$A \subseteq \mathbb{N}$

Sets: def: A set A is called decidable. if there is an algorithm, that given a natural number n, decides whether $n \in A$.



1. Is empty sets decidable? yes, since whatever n is, $n \notin \phi$.
   Algorithm: always return (no) thus always returns 0.
2. Is $\mathbb{N}$ decidable?
   Algo: always return (yes)
3. A finite set $\{n_1, \dots n_k\}$ → algo: compare each element with n.
4. If A is decidable & B is decidable, is $A \cup B$ decidable



$A \cup B$.

check if $n \in A$.
if yes return true
if no check $n \in B$?

$\therefore n \in A \cup B \equiv n \in A \,||\, n \in B$.

5. $A \cap B$
6. if A is decidable is $-A$ decidable? (A) -A

$n \in -A \Rightarrow n \notin A \Rightarrow !(n \in A)$

check if $n \in A$
if yes return false

7. Is there a set which is not decidable?

$\{\langle P, d\rangle : P \text{ halts on } d\}$ is not decidable.

The set of all programs, $\{P : P \text{ always returns } 0\}$ is not decidable.

Def: A set A is called recursively enumerable (r.e.) if $\exists$ an algorithm that eventually prints all elements of A.

```
n = 0;
while (true) { System.out.println (n);
               n++; }
```

This shows that $\{n\}$ is recursively enumerable.