



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.  $\rightarrow$  ② & ① solved.

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

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

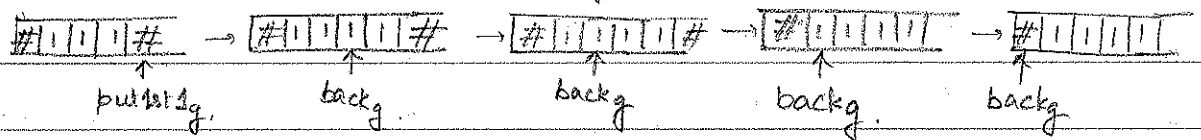
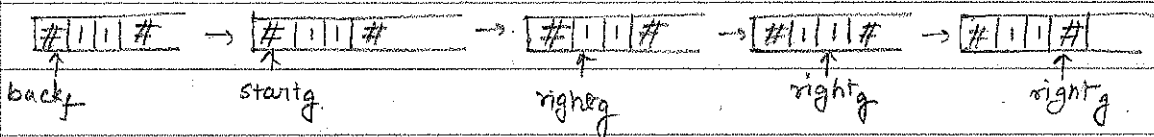
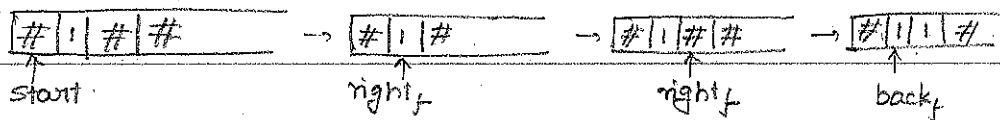
$halt_f$  will be  $(start_g, \#)$ .

$halt_g$  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  $A \cap B$  decidable.

Now, got :- for,  $f: n \rightarrow n+1$ ,  $g: n \rightarrow n+2$ .



Row 4:  $\boxed{\# \mid \mid \mid \mid \mid \mid \#}$

Label below: halt.

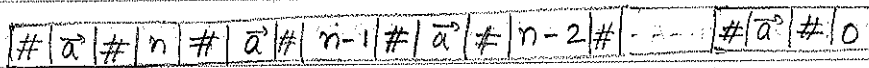
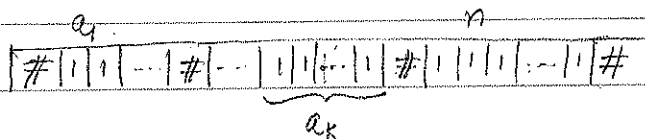
[PR is a formalization of for loop.]

TM for PR function:-

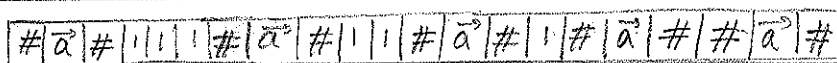
$$h = PR(f, g) \rightarrow h = f$$

$$\text{for}(i=0; i < m; i++) \{ h = g(x, f(x)) \}$$

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



untill we reach 0



$h(\vec{a}, 0) = f(\vec{a})$ . In general

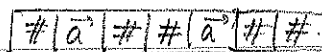
$$\# | a^1 | \# | 1 | 1 | 1 | \# | a^2 | \# | 1 | \# | a^3 | \# | \# | h(\vec{a}, 0) | h(\vec{a}, n+1) = g(\vec{a}, n, h(\vec{a}, n))$$



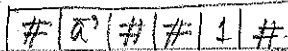
$$f(\vec{a}^3) = \mu m. P(\vec{a}^3, m)$$



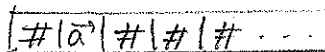
Step 1: we copy  $\vec{a}^3$ .



apply P from here.



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:

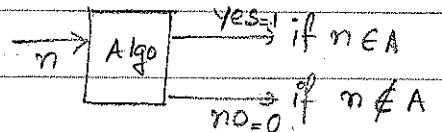
- recursive  $f_n$
- TM: bitwise.

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

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

$$A \subseteq \mathbb{N}$$

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,  $n \notin \emptyset$ .

Algorithm: always return **no** thus always returns 0.

2. Is  $\mathbb{N}$  decidable?

Algo: always return **yes**

3. A finite set  $\{n_1, \dots, n_k\} \rightarrow$  algo: compare each element with  $n$ .

4. If  $A$  is decidable &  $B$  is decidable, is  $A \cup B$  decidable.



$A \cup B$ .

check if  $n \in A$ .

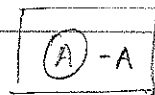
if yes return true

if no check  $n \in B$ ?

$$\therefore n \in A \cup B \equiv n \in A \vee n \in B.$$

5.  $A \cap B$

6. if  $A$  is decidable is  $\neg A$  decidable?



$$n \in \neg A \Rightarrow n \notin A \Rightarrow \neg(n \in A)$$

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.