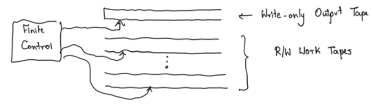
CS 205
Lecture 31
31/3/2021

TMs as Language Generators



Write-only output tape : - transitions of TM not affected by symbol on the tape
- tape head only moves right

- M starts with all tapes blank except the output tape, which contains #
- From time to time M writes some word w on the output tape followed by #

Def : M generates a string w if at some point M writes $w\#$ on the output tape.

w_1	#	w_2	#	w_3	#	...

Language generated by M : $G(M) = \{w \mid M \text{ generates } w\}$

Note : M need not generate strings in any order.

Def : L is recursively enumerable (re) if there is a TM M s.t. $L = G(M)$.

Recall : L is semi-decidable if there is a TM M s.t. $L = L(M)$.

Thm : L is r.e. iff L is semi-decidable.

Proof : ($\Rightarrow$) Let M' be a TM s.t. $L = G(M')$. Need to construct a TM M s.t. $L = L(M)$.

- M runs M'
- Whenever M' writes $w\#$, M checks whether w is the same as the input u.
- If $u = w$ then M halts and accepts u.

($\Leftarrow$) Let M be a TM s.t. $L = L(M)$. Need to construct M' s.t. $L = G(M')$.

M' : for $w = \underset{w_0}{\varepsilon}, \underset{w_1}{0}, \underset{w_2}{1}, \underset{w_3}{00}, \ldots, 01, 10, 11, \ldots$ (Canonical order)

Simulate M on w
if M accepts w then write $w\#$ on output tape.

Does not work since M may not halt on a string $w \notin L$ in which case M' will not output any more strings.

Solution : Dovetailing
Method to carry out an infinite no. of computations "in parallel.

1. Run $M(w_0)$ for 1 step.
2. Run $M(w_0)$ for 2 steps. Run $M(w_1)$ for 1 step.
3. Run $M(w_0)$ for 3 steps. Run $M(w_1)$ for 2 steps. Run $M(w_2)$ for 1 step.

⋮

i. Run $M(w_0)$ for i steps. ... Run $M(w_{i-1})$ for 1 step.

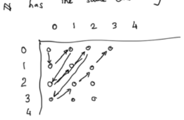
⋮

If at any point $M(w_j)$ halts and accepts output $w_j\#$.

Note : 1. If M halts before i steps on w_j then "run $M(w_j)$ for i steps means "run $M(w_j)$ until it halts".
2. A string w_j may be output many times.

Alternative View of Dovetailing

$\mathbb{N} \times \mathbb{N}$ has the same cardinality as $\mathbb{N}$



Pairing function $\langle i, j \rangle = \dfrac{(i+j+1)(i+j)}{2} + j$ bijection from $\mathbb{N} \times \mathbb{N} \to \mathbb{N}$

Generator M':
for $k = 0$ to ∞ do
Find (i, j) s.t. $\langle i, j \rangle = k$
Simulate M on w_i for j steps
If M accepts w_i in $\leq j$ steps then output $w_i\#$

Recall : L is recursive (or decidable) if L is accepted by a TM M that halts on all inputs.

Thm : L is recursive iff there is a TM M s.t. M generates L in canonical order.

Proof : ($\Rightarrow$) Let M be a TM that halts on all inputs and $L(M) = L$.

Generator of L
M' : for $i = 0$ to ∞ do
Run M on w_i
if M accepts w_i then write $w_i\#$ on the output tape.

($\Leftarrow$) Suppose M' generates strings in L in canonical order.

M : On input w
Find i s.t. $w = w_i$
Run M' until it either generates $w_i = w$;
- M accepts w
or M' generates $w' > w$ without generating w;
- M rejects w

Works only if L has an infinite no. of strings!

If L is finite then L is regular. So clearly there is a TM (the DFA for L) that always halts and accepts L.