Mar 11 2025

Write a TM (diagram) recognizing $\{\#b_i\#b_{i+1} : b_i$ is no-leading-0's binary rep of positive integer $i.\}$

# 1 0 1 # 1 1 0 ␣ ␣ ␣ ␣	goes to q_a

# 1 0 1 # 1 1 # # ␣ ␣	goes to q_r

See 3.10 – 3.12 Examples in text.

From last time, we want to be able to execute a high-level instruction like "restore the X-ed out b's"

How?

- Introduce a new symbol $\bar{b}$ or B to symbolize an X'ed out letter b.
- B represents that a b used to be in this cell; now is is an x'ed out character.

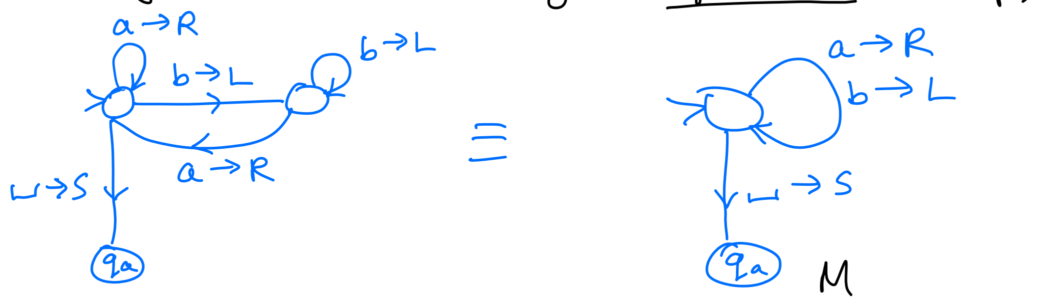
Def$^{\underline{n}}$: A TM M recognizes language L_1 iff $L_1 = \{w \mid M \text{ accepts } w\}$. Then we say $L(M) = L_1$.

$$\boxed{\text{Definition 3.5 A language is Turing-recognizable if some TM recognizes it.}}$$

recursively-enumerable.

Now, on a given input a TM might <u>loop forever</u>. ($\equiv$ loop)

E.g.



$L(M) = L(a^*)$

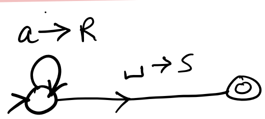
loop $\equiv$ does not halt.

M, on input w can
- accept (q_a)
- reject (q_r)
- loop

Defn: M is a <u>decider</u> if it halts on all inputs, and we say $L(M)$ is decided by M.

> Defn 3.6 A language is Turing-decidable if $\exists$ a TM that decides it.

M_2 that decides $L(a^*)$:

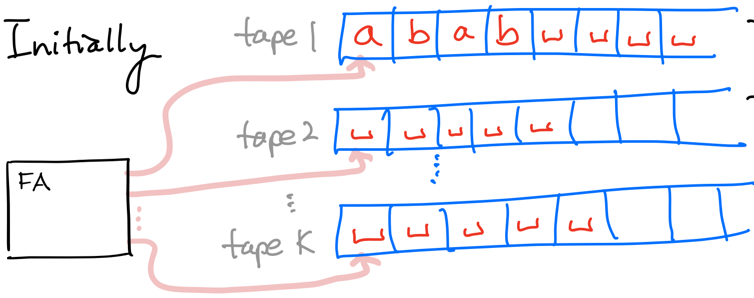


Note: hanging ($\nexists$ transition to make) is a <u>decisive</u> action
- all missing transitions go to q_r

Variants of TMs

Multitape TM
- a TM with K tapes, $K \geq 1$
- each tape has its own head

— Initially



- Input is on tape 1
- All other tapes contain blanks
- at start, all heads are on leftmost cell.

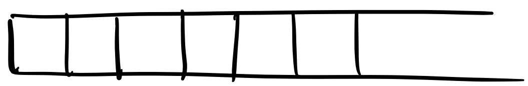
Transition function

$$\delta : Q \times \Gamma^k \to Q \times \Gamma^k \times \{L, R, S\}^k$$

Recall: 2 machines are equivalent if they recognize the same language.

A computational model is more powerful than another if it can recognize all the languages the other can plus more.

Eg. For each PDA M_P, $\exists$ a TM M_T that recognizes the same language ... because a STACK is a restricted kind of TAPE



ie the stack can be simulated by a tape.
And TMs can recognize languages PDA's cannot, like: $\{ w\#w : w \in \{a,b\}^* \}$.

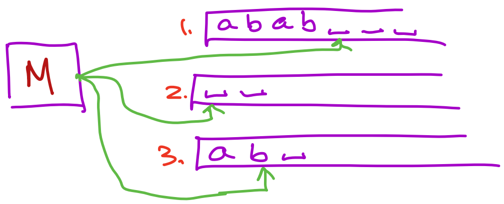
$\therefore$ TM is a more powerful computational model than PDA

Theorem 3.13 $\forall$ multi-tape TM has an equivalent single tape TM (TM)

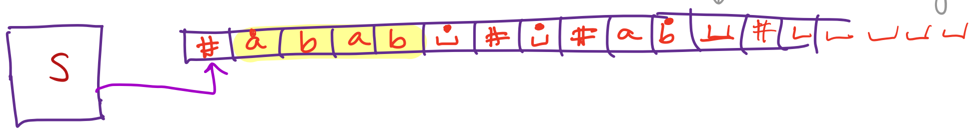
Proof: Given a multitape TM M we can construct an equivalent single-tape TM S

S simulates M

Idea: S will store the contents of all k tapes serially on its single tape.



is a symbol in S's tape alphabet that is not in M's — to allow us to mark each tape's territory.



S = " On input $w = \sigma_1 \sigma_2 \cdots \sigma_n$

1. insert a # at leftmost cell and mark off k tape areas that are initially blank, to L of w, with dot (head position) on first symbol of each tape-area.

2. Scan R and determine all symbols under all tape heads. Make a second pass, to update the symbol under each head, move the dot (as M would move the head), and change state as M would do.

3. If S needs to access tape areas outside those already demarcated by the #'s, then use a transducer to shift-R all tape

contents to R of needed cell, and
continue as before.

4. If and when M goes to q_a — ACCEPT
" " q_r — REJECT "

This TM S accepts w if M accepts w
rejects w if M rejects w
loops on w if M loops on w

$\therefore$ S is equivalent to M

$\therefore$ $\forall$ multitape TM can be simulated by a (single-tape) TM. $\blacksquare$

Corollary 3.15 A language is Turing-recognizable iff
$\exists$ a multitape-TM that recognizes it.

Proof: ($\Rightarrow$) Easy - a TM is a special (k=1) multitape TM.
($\Leftarrow$) we just proved it in Theorem 3.13. $\blacksquare$

Non-deterministic TM

A non-det TM may have several options of what to do in a particular state, with tape head over a particular symbol.

$$\delta : Q \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{L, R, S\})$$

power set, ie $\exists$ a number of options of what to do, where to go when in a certain config.

The computation of a det-TM M on input w:

$(q_0, aabab\ldots b)$

$|$

$(q_1, Aabab\ldots b)$

$|$

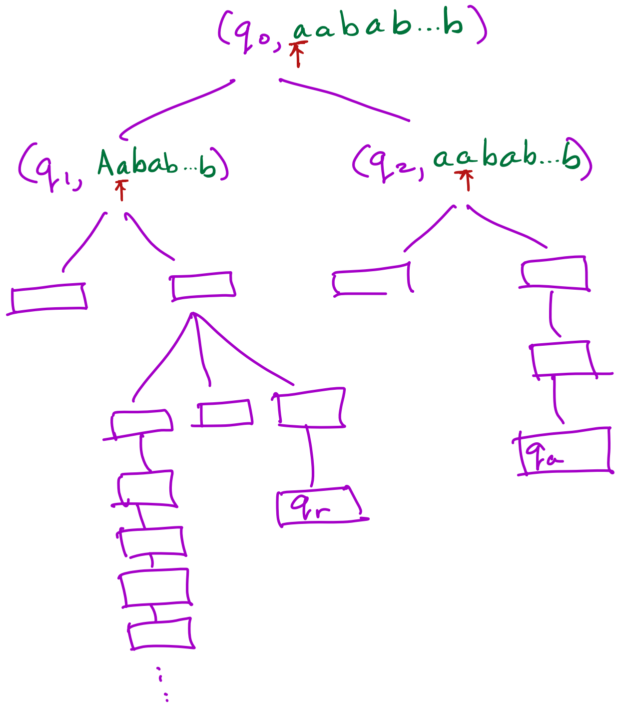
$(q_1, AAbab\ldots b)$

$|$

$(q_2, AAbab\ldots b)$

$\vdots$

The computation of a non-det TM N on input w:



Def$^{\underline{n}}$: A non-det TM M accepts w if

$\exists$ a branch of M's computation on w

that ends in h_a

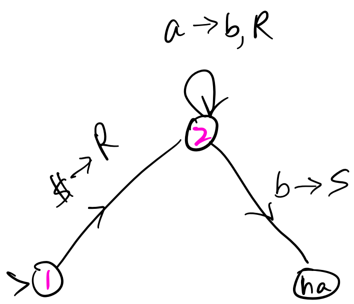
Question: Does non-determinism add "power" to the computational model "TM"?

I.e. are there languages recognized/decided by non-det TMs which no det-TM can recog/decide?

Review:

Configuration of a TM:

If you have a TM that is part way through a computation, what do you have to know to predict exactly the rest of the computation (for a det-TM) or all possible ways the computation could go (for a nondet-TM)?



- current state
- tape contents
- where the head is

(2, \$bb̲bb)