Chapter 3 Turing Machines

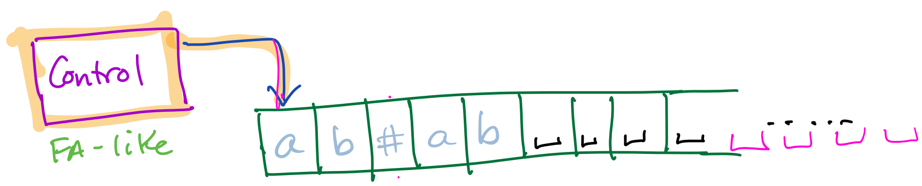
FA - good for small amount of memory

PDA — good for tasks use LIFO memory.

But we want to explore general computability, and neither FA nor PDA can even recognize $\{a^n b^n c^n \mid n \geq 0\}$ or $\{w\#w \mid w \in \{a,b\}^*\}$.

Turing Machines (TMs)

- Alan Turing in 1936
- FA with a one-way-infinite tape, R/W
  - tape head can move L and R.
  - $\exists$ an accept state and a reject state - halt



Formal Defn of a TM.

Defn 3.3 A Turing Machine (TM) is a 7-tuple.

$$(Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$$

Where: 1. Q is a finite set of states.

2. Σ is input alphabet, $\sqcup \notin \Sigma$

3. Γ is tape alphabet, $\sqcup \in \Gamma$, and $\Sigma \subseteq \Gamma$

4. $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$ is the transition function.

5. q_0 is start state

6. q_a is ACCEPT state

7. q_r is REJECT state.

A TM computes as follows:

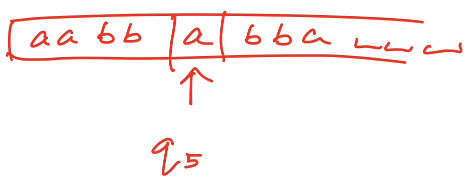
it starts with: - input string occupying all leftmost cells of tape, upto leftmost blank.
- tape is all "␣" after the input string.
- tape head is on leftmost cell.

Start up M ...

at each step M: - reads cell under current head pos.
- is in a given state
  - if state is q_a - Halt & ACCEPT
  - if state is q_r - Halt & REJECT

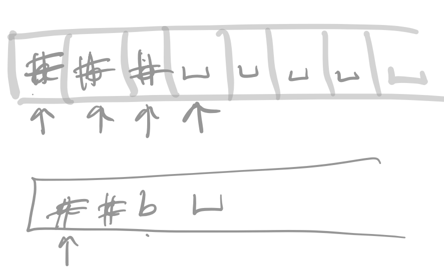
based on state, tape cell contents:
- write a symbol to current cell
- Move L, R, S.
- change state.

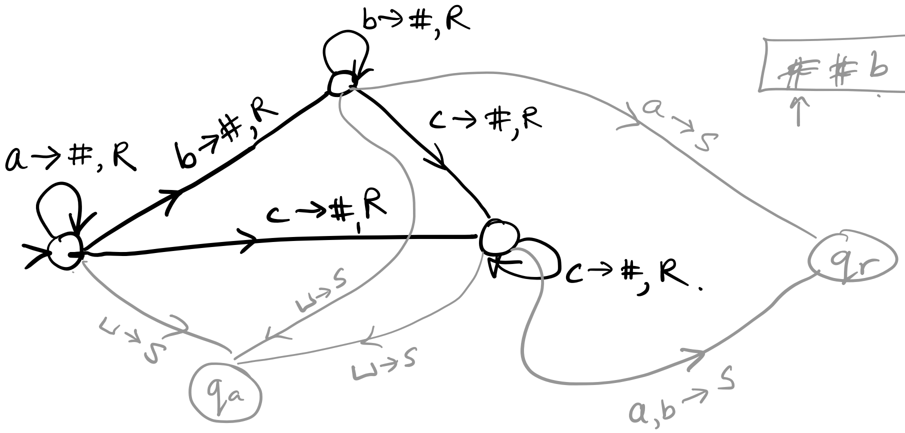
Note: if M is on leftmost cell and is supposed to move L, it just stays.



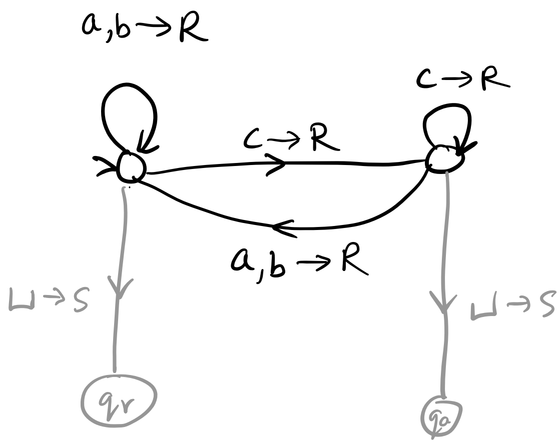
A configuration of a TM is all info necessary to say how rest of comput'n will proceed.

TM for $L(a^* b^* c^*)$:

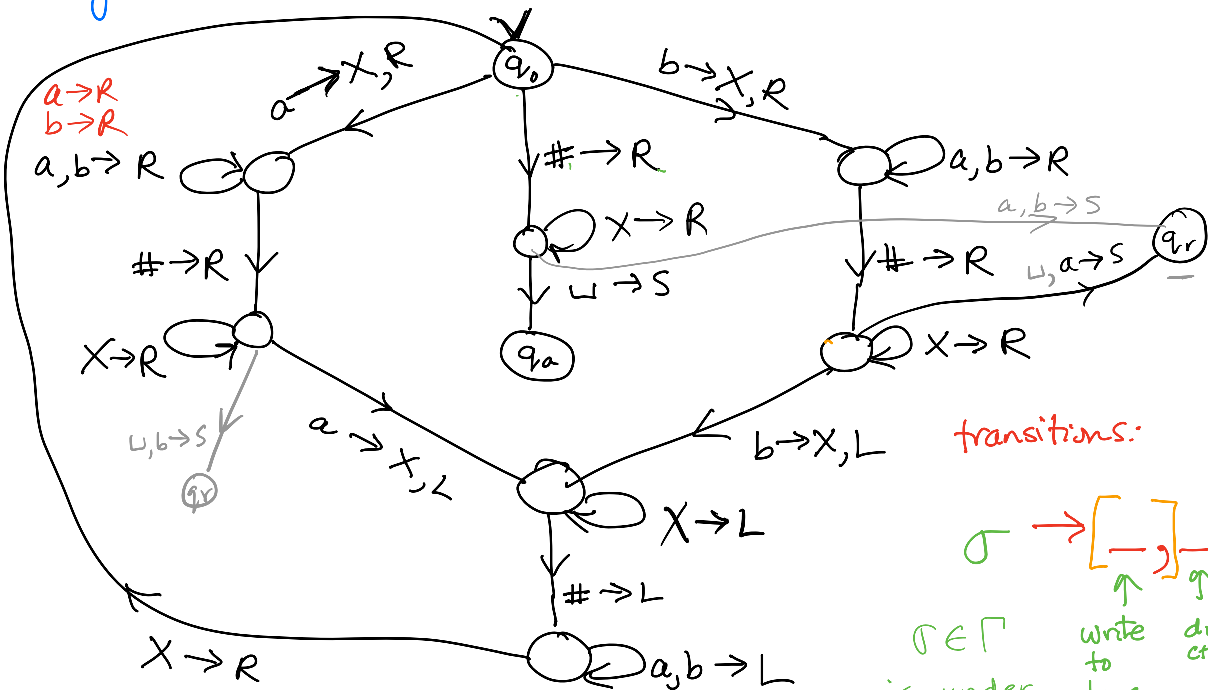




TM for "ends in c", $\Sigma = \{a, b, c\}$



Eg TM that recognizes $\{w\#w \mid w \in \{a,b\}^*\}$

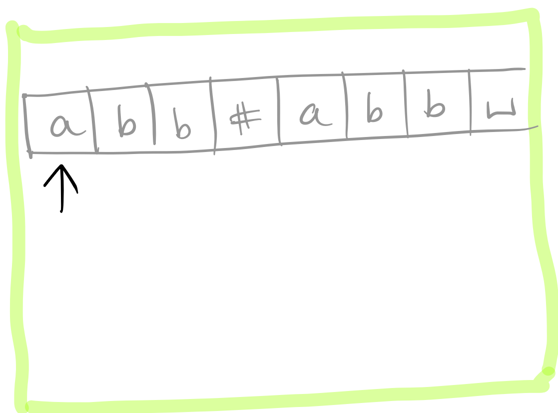


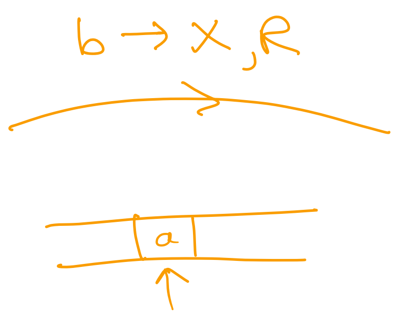
transitions: $\sigma \to [\_, \_]\_$

$\sigma \in \Gamma$ is under read head

write to tape

direction

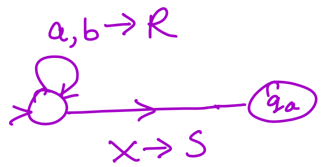




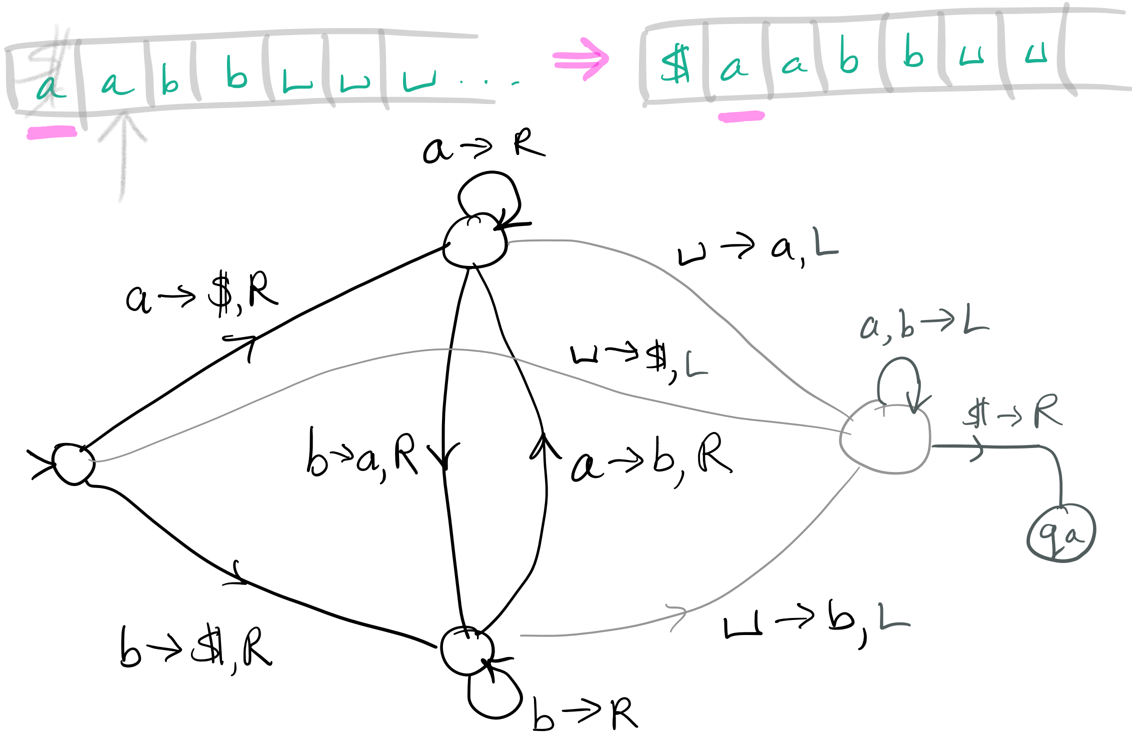
Every TM can be shown as a diagram in this way.

That said, they quickly become large and ungainly if they are doing anything complex. So we shift to a higher level description of a TM. — but we could draw the diagram if we wanted to.

eg. high-level – "scan R to first X and stay" for some $X \in \Gamma$



eg high-level description: "shift tape contents R one cell, and insert \$ at leftmost cell" $\Sigma = \{a, b\}$



$M_{\$\,\text{Shift}\,R}$ $\Sigma = \{a, b\}$

Such a TM is useful as a "function call" by a bigger TM. It transforms the tape contents — in essence, A transducer TM computes a function

$f : \Sigma^* \to \Gamma^*$

input = original tape contents
output = tape content after executing the transducer TM.

Eg of a high-level description of a TM.

$C = \{ a^i b^j c^k \mid i \times j = k \quad i, j, k \geq 1 \}$

$M =$ " On input string w :

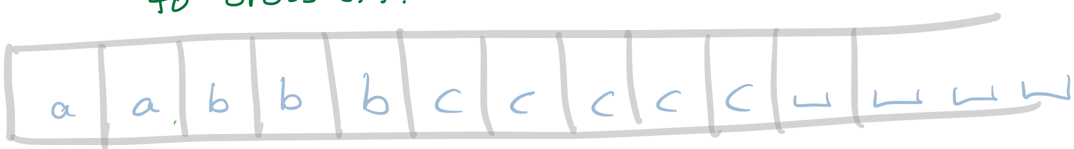
0. Shift-R the input string, leaving \$ on leftmost cell.

1. Scan R to determine if it is a member of $a^+ b^+ c^+$ — REJECT if not of this form.

2. Return head to leftmost cell.

3. Cross off an a and scan right to first b.
   Shuttle between b's and c's,
   B-ing out a b and X-ing out a c. each time.
   - if b's end early — REJECT
   if c's end early — REJECT.

4. Restore B'd-out b's and go to 3, if ∃ an a to cross off.

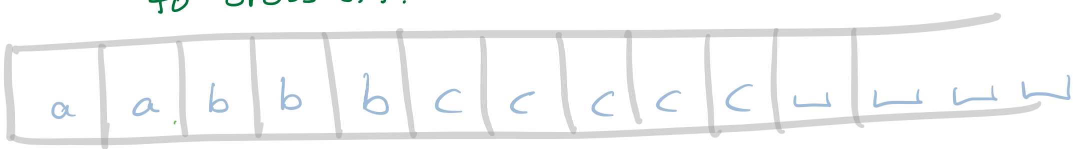


Eg of a high-level description of a TM.

$C = \{ a^i b^j c^k \mid i \times j = k \quad i,j,k \geq 1 \}$

M = " On input string w :

1. Scan R to determine if it is a member of $a^+ b^+ c^+$ — REJECT if not of this form.
2. Return head to leftmost cell.
3. Cross off an a and scan right to first b.
   Shuttle between b's and c's,
   X_b-ing out a b and a c each time.
   - if b's end early — REJECT
   - if c's end early — REJECT.
4. Restore X'd-out b's and go to 3, if $\exists$ an a to cross off.

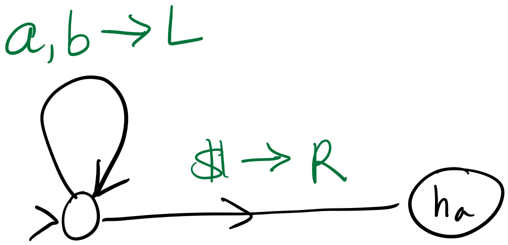


- If all a's have been crossed off, Check that all c's have been X-ed off.
  If yes - ACCEPT
  no - REJECT."

How do we return to leftmost cell?

We use $M_{\$ \text{ Shift}}$ to insert \$ onto leftmost cell (assuming \$ is not used elsewhere in the TM - ie introduce a symbol used only for this special purpose).

Then any time the high-level description says "move to leftmost cell", we do this:

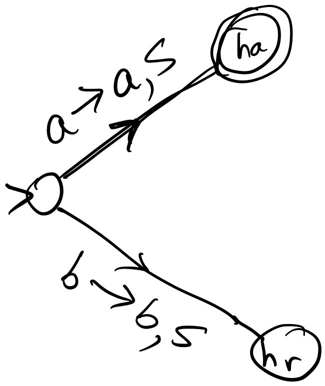


first \$ encountered

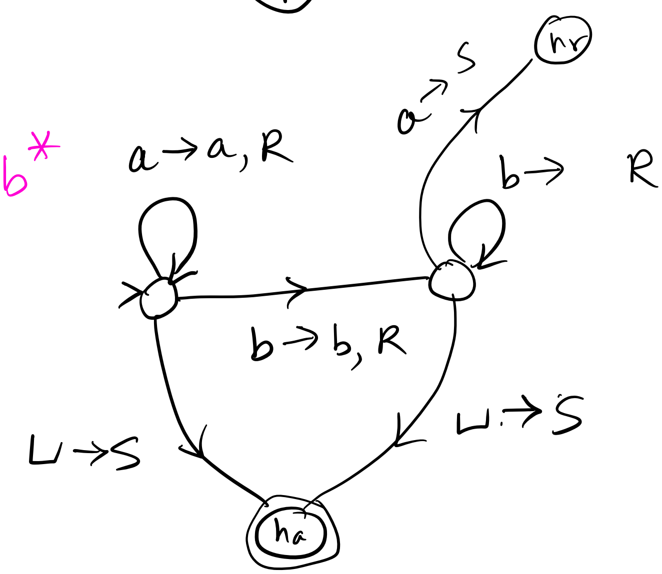
"$L_{\$}R$" = "Move L to \$, then R once"

Next time: How does a TM "restore the X-ed out b's".

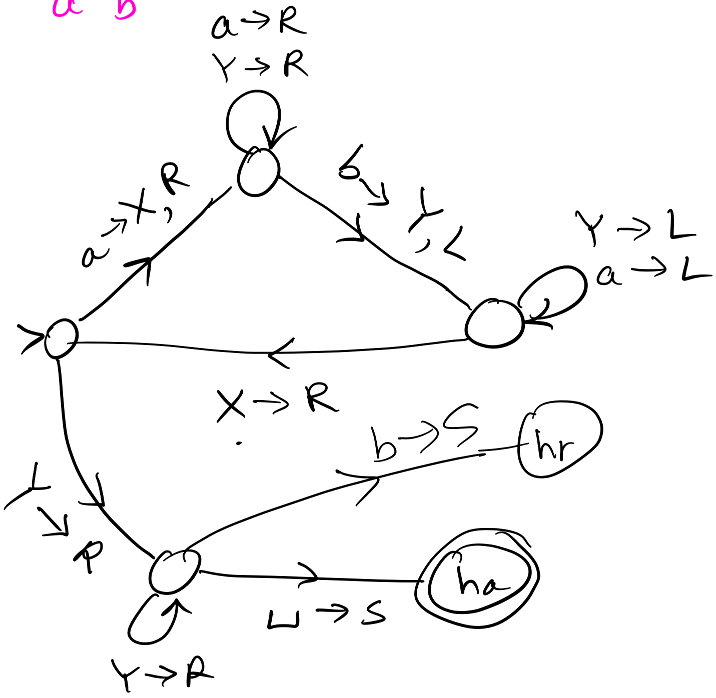
accepts if first letter is a , rejects o.w.



$a^* b^*$



$a^n b^n$

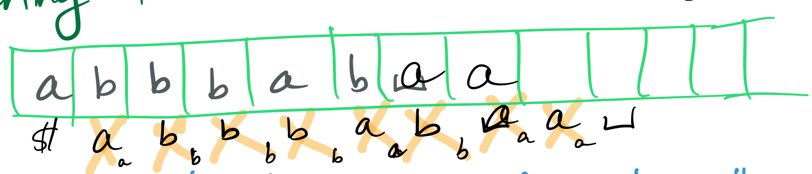


A missing transition means "hang" – another way of rejecting.

To "X-out b's" that may need to be restored:

Eg: A transducer TM that erases the input string w if $\#_a(w) \geq \#_b(w)$, and shifts it R inserting \$ on leftmost cell either way.

Idea:



1. Insert \$ to mark leftmost cell, Shift-R the cell contents, using $M_{\$ShiftR}$:

2. Scan L to \$.

3. Scan R to "a", and X it out
if no "a", go to 8. scan L to $, scan right to b; if b,
if no "b" - go left to $, move R.

4. Scan L to $

5. Scan R to "b", and X it out.
if no "b", go to 11

6. Scan L to $

7. Go to 3.

8. Scan L to $

9. Scan R to ␣, and
for each X'd out "a", restore to "a"
for each X'd out "b", restore to "b".

10. Scan L to $, halt.

11. Scan R to ␣; move L.

12. if "a" or "b", write "␣", move L.
if "$", stay and halt.