More Variants of Turing Machines

Recall: Saw that the following modifications did NOT affect the computational power of our Turing machine model:

- Adding a "stay" action for the tape head
- Adding a left end marker

Showed these equivalences by way of "simulations"

"Vanilla" model can simulate the operation of the modified model, and vice versa $\Rightarrow$ Equivalent!

Today: More modifications to the model

What functionality can one add to our existing definition of TMs?

→ Add an extra bidirectional tape

Adding an extra stack to a PDA could let me recognize

$L = \{a^n b^n c^n \mid n \geq 0\}$, which is not context-free.

Can adding a tape (or multiple such) to a TM increase its computational power? No!

Easy to simulate a one-tape TM with a multiple-tape TM

What about the other way?

Suppose I have a one-tape TM M_1, and I want to simulate a k-tape TM M.

M must have k heads, each pointing to some letter on each tape

How long a string does each tape contain at any point in a run?
Finite!

How large is the tape alphabet of M? What is the worst case?
Countable

Track contents of all tapes, current positions of all tape heads

easy; concatenate with separators

how?

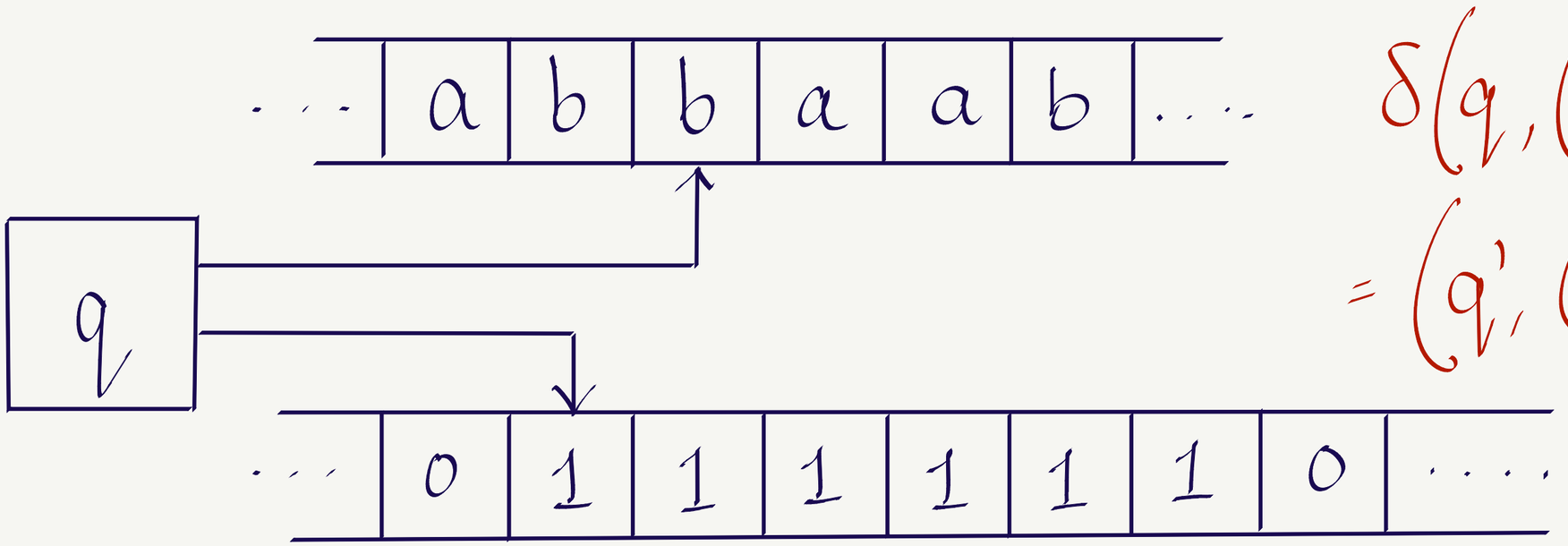
Could add a new "head symbol" for each tape,
but this needs us to modify transitions to "ignore" this symbol.

Extend the tape alphabet

Add a new "this is the head" symbol for every tape symbol

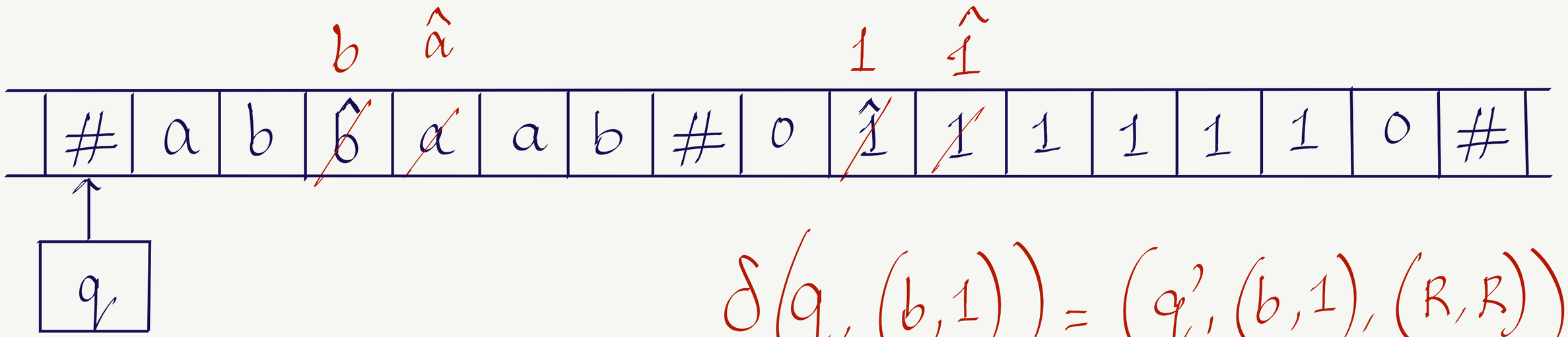
$$\Gamma = \Gamma_1 \cup \Gamma_2 \cup \dots \cup \Gamma_k \cup \{\#\} \cup \bigcup_i \{\hat{a} \mid a \in \Gamma_i\}$$

The transitions on $\hat{a}$ remain the same as on a.



$$\delta(q,(c_1,c_2)) = (q',(c_1',c_1'),(L,R))$$

is simulated as follows



$$\delta(q, (b,1)) = (q', (b,1), (R,R))$$

Input starts at the first #
Scan all the way right till you see k+1 #s
Scan back left till you see a ␣ to the left of a #
Scan right, keep track of the "head symbols", and
finally make the appropriate change in configuration. Stop if r or t.
Move back left to the leftmost # and repeat.

What functionality can one add to our existing definition of TMs?

→ Add non-determinism

$$\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}$$

$$\delta' \subseteq Q \times \Gamma \times \left(Q \times \Gamma \times \{L, R\}\right)$$

Does this add extra computational power?

What functionality can one add to our existing definition of TMs?

→ Add non-determinism

$$\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}$$

$$\delta' \subseteq Q \times \Gamma \times (Q \times \Gamma \times \{L, R\})$$

Does this add extra computational power? No!

Unsurprisingly, we can determinize a non-deterministic TM
(just like we did for NBA!)

Keep track of the tree of possibilities, each node a configuration
Each branch represents a possible computation

What if a branch is infinite? Do NOT do depth-first search!

Breadth-first search of the tree

Explore each branch to the same depth before proceeding

Visit each node till we hit an accepting configuration

Use a deterministic TM with multiple tapes

* One to hold the input (the contents of this tape never change)
* A work tape to simulate nondeterministic operation on
* A tape to keep track of where in the tree one is currently
    - Contains some node of tree (expressed as the path from root):
      Simulate operation using work tape, see if end up in $t \in Q$
      If not, move on to lexicographically next node