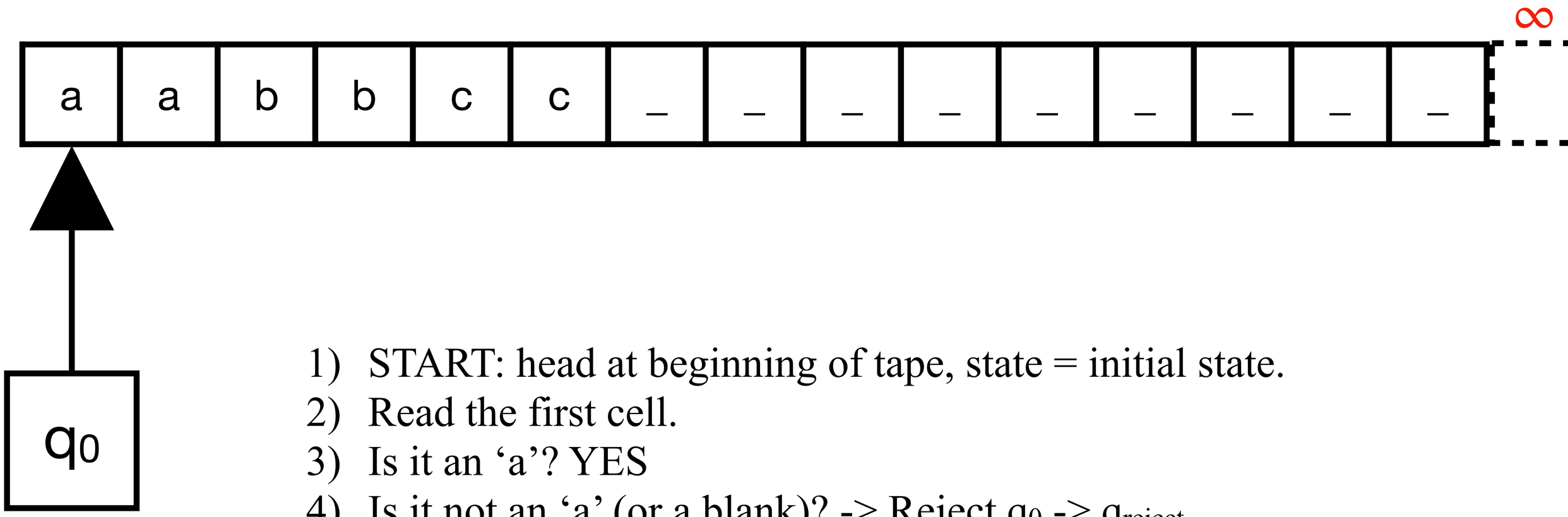
Example of TM computation

Language (not a CFL!): $L = \{a^n b^n c^n \mid n \geq 1\}$

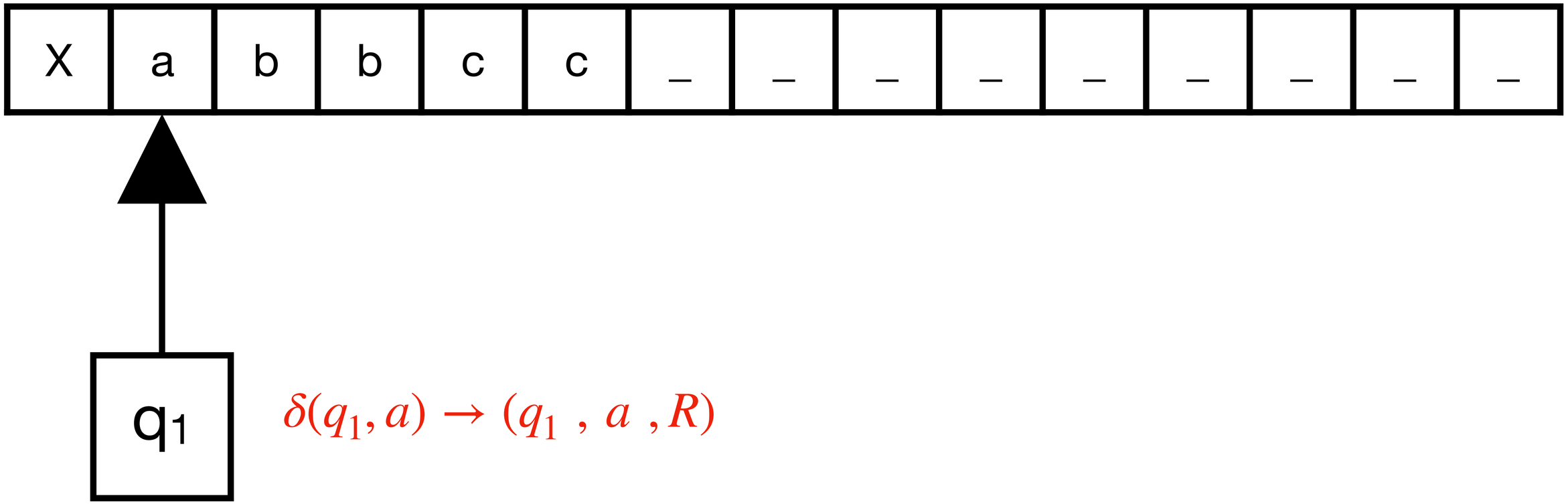
"Algorithm" idea:

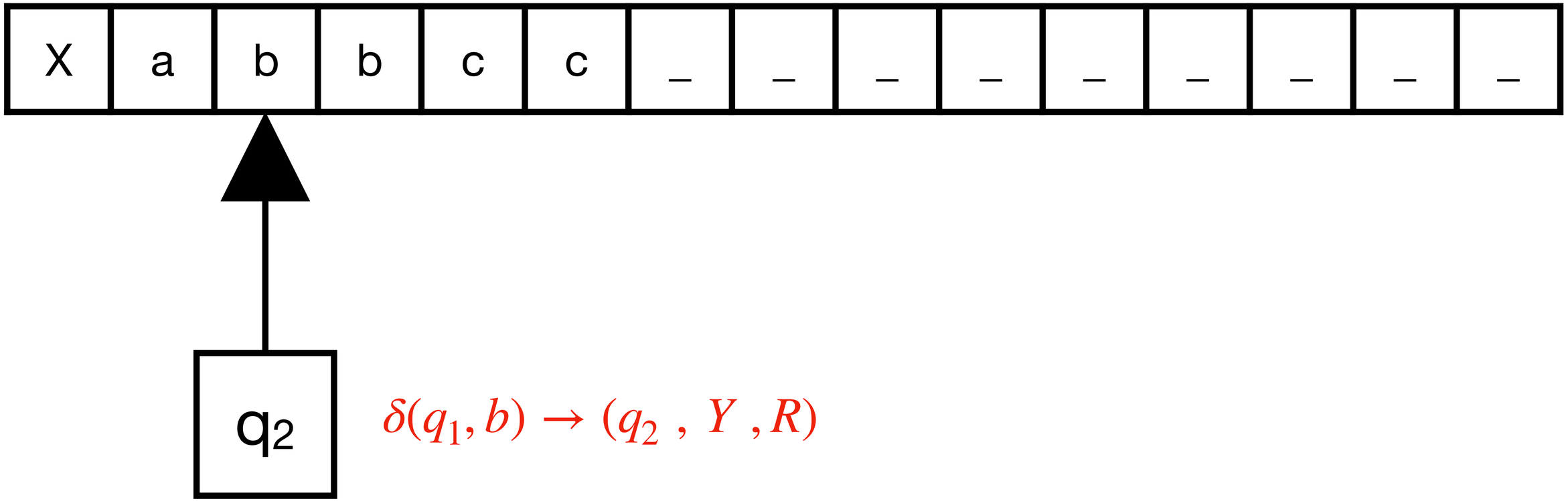
1) Find the leftmost unmarked 'a' and mark it with X,
2) Move right fund the first unmarked 'b', mark with Y,
3) Move right, find the first unmarked 'c', mark ut with Z,
4) Return to the left and repeat from 1),
5) If not unmarked 'a' remains, accept IFF everything is marked.



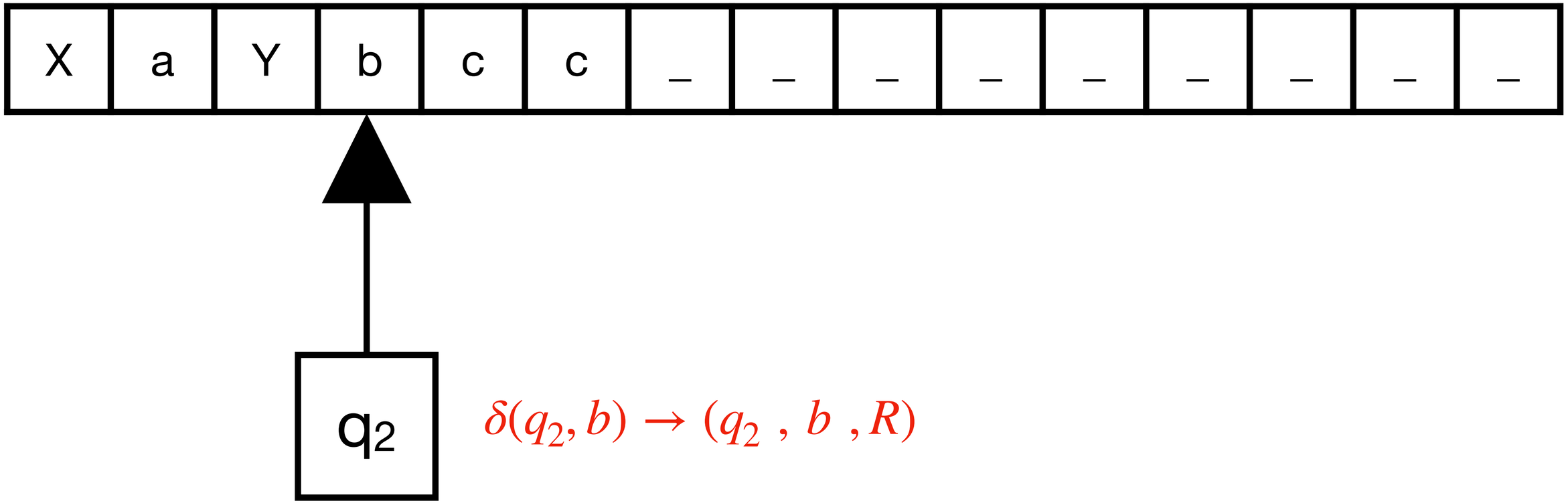
1) START: head at beginning of tape, state = initial state.
2) Read the first cell.
3) Is it an ‘a’? YES
4) Is it not an ‘a’ (or a blank)? -> Reject q_0 -> q_{reject}

If it is an ‘a’: $\delta(q_0, a) \rightarrow (q_1\ ,\ X\ ,\ R)$

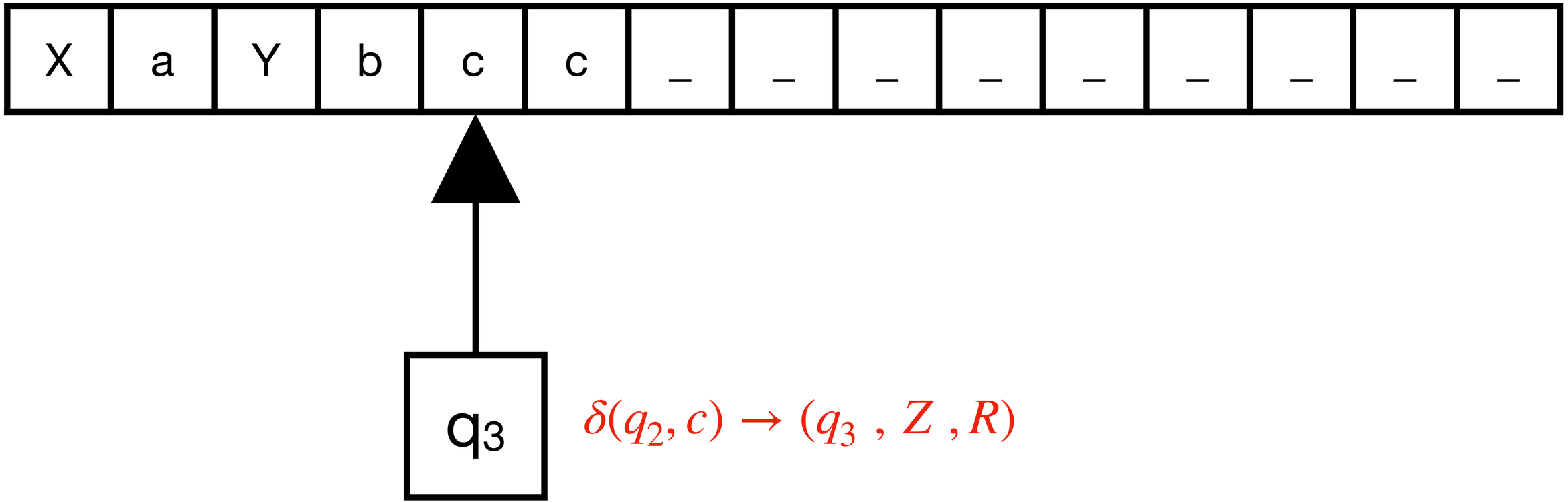

X
a
b
b
c
c
_
_
_
_
_
_
_
_
_
q1
$\delta(q_1, a) \rightarrow (q_1 \ , \ a \ , R)$

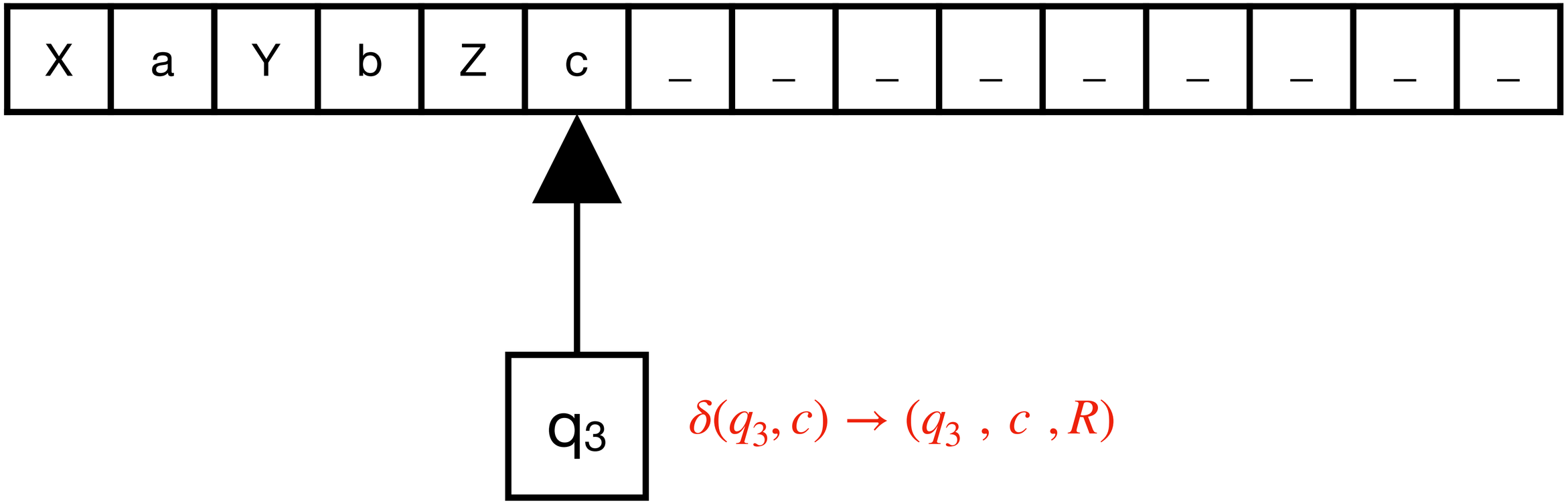


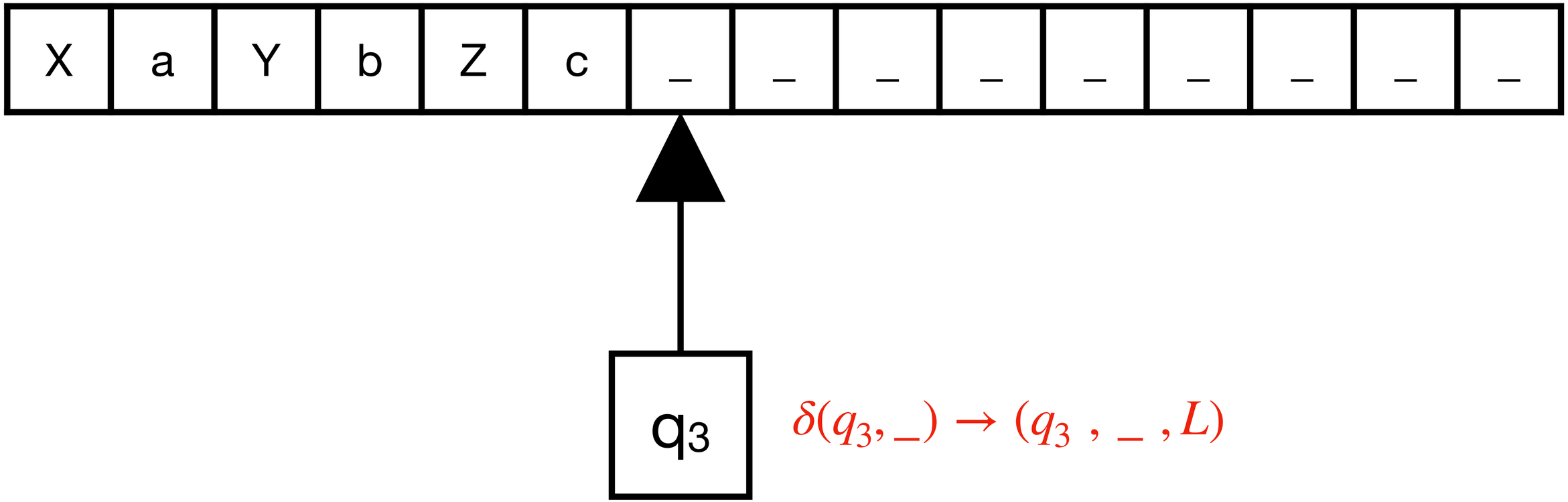
$\delta(q_1, b) \rightarrow (q_2\ ,\ Y\ ,R)$



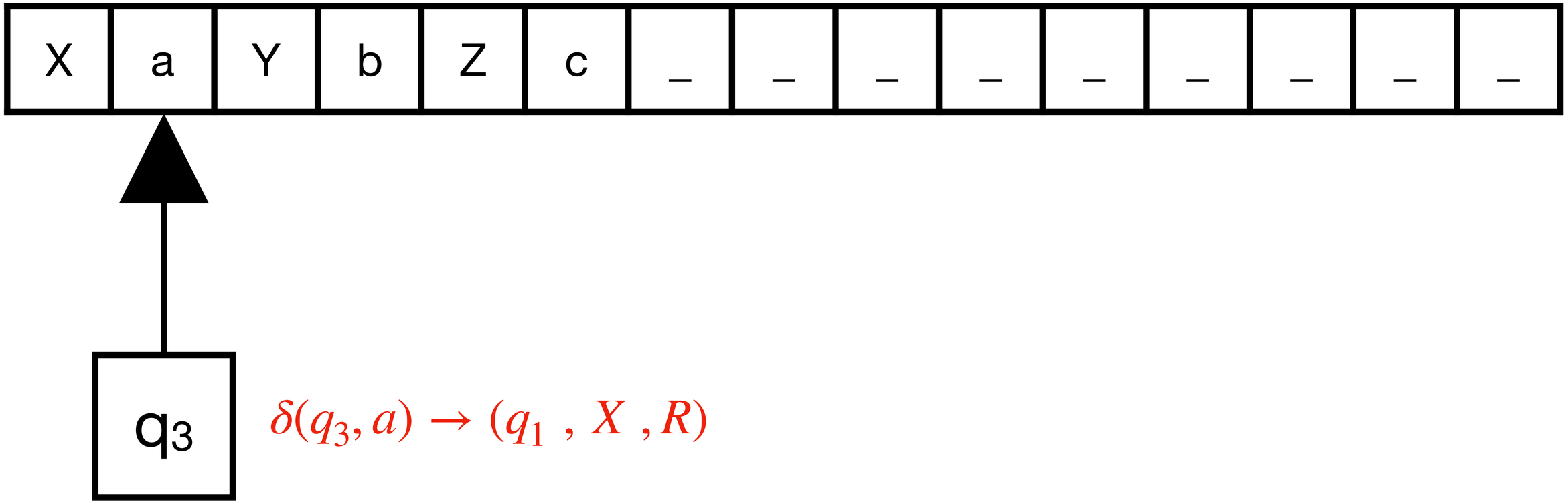
$$\delta(q_2, b) \rightarrow (q_2\ ,\ b\ ,R)$$


X
a
Y
b
c
c
_
_
_
_
_
_
_
_
_
q3
$\delta(q_2, c) \rightarrow (q_3\ ,\ Z\ ,R)$

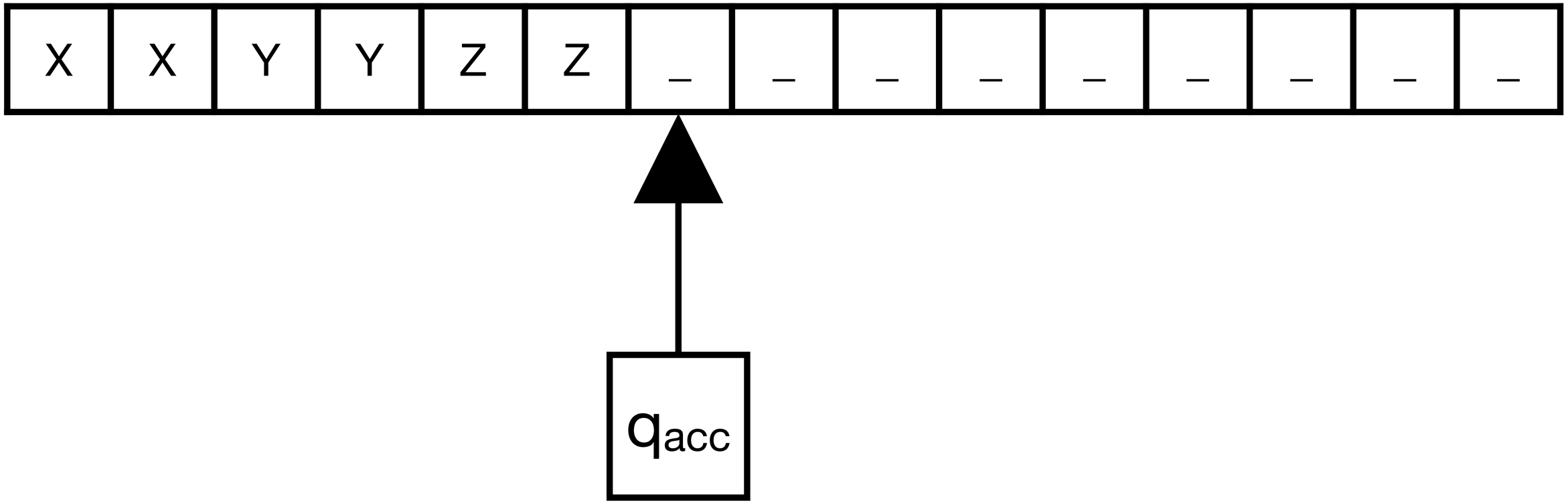

X
a
Y
b
Z
c
_
_
_
_
_
_
_
_
_
q3
$\delta(q_3, c) \rightarrow (q_3\ ,\ c\ ,R)$



- And keep going left until we find an ‘a’
- If we reach the head/start symbol, accept



- Repeat the previous procedure until accept (all marked) / reject.
- Skip if read X,Y,Z .


X
X
Y
Y
Z
Z
_
_
_
_
_
_
_
_
_
q_{acc}

Complete Table

State	a	b	c	X	Y	Z	▢
q_0	(q_1,X,R)	(qrej,b,R)	(qrej,c,R)	(q_0,X,R)	(q_0,Y,R)	(q_0,Z,R)	$(q_4,▢,R)$
q_1	(q_1,a,R)	(q_2,Y,R)	(qrej,c,R)	(q_1,X,R)	(q_1,Y,R)	(qrej,Z,R)	(qrej,▢,R)
q_2	(qrej,a,R)	(q_2,b,R)	(q_3,Z,L)	(qrej,X,R)	(q_2,Y,R)	(q_2,Z,R)	(qrej,▢,R)
q_3	(q_3,a,L)	(q_3,b,L)	(q_3,c,L)	(q_3,X,L)	(q_3,Y,L)	(q_3,Z,L)	$(q_0,▢,R)$
q_4	(qrej,a,R)	(qrej,b,R)	(qrej,c,R)	(q_4,X,R)	(q_4,Y,R)	(q_4,Z,R)	(qacc,▢,R)

Final steps

Final skipping

State	a	b	c	X	Y	Z	□
q_0	(q_1,X,R)	(qrej,b,R)	(qrej,c,R)	(q_0,X,R)	(q_0,Y,R)	(q_0,Z,R)	$(q_4,□,R)$
q_1	(q_1,a,R)	(q_2,Y,R)	(qrej,c,R)	(q_1,X,R)	(q_1,Y,R)	(qrej,Z,R)	(qrej,□,R)
q_2	(qrej,a,R)	(q_2,b,R)	(q_3,Z,L)	(qrej,X,R)	(q_2,Y,R)	(q_2,Z,R)	(qrej,□,R)
q_3	(q_3,a,L)	(q_3,b,L)	(q_3,c,L)	(q_3,X,L)	(q_3,Y,L)	(q_3,Z,L)	$(q_0,□,R)$
q_4	(qrej,a,R)	(qrej,b,R)	(qrej,c,R)	(q_4,X,R)	(q_4,Y,R)	(q_4,Z,R)	(qacc,□,R)

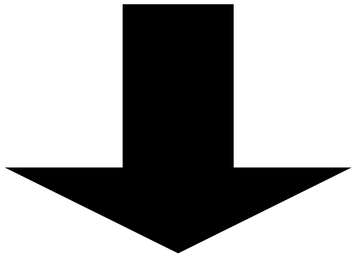
Final steps

State	a	b	c	X	Y	Z	▢
q_0	(q_1,X,R)	(qrej,b,R)	(qrej,c,R)	(q_0,X,R)	(q_0,Y,R)	(q_0,Z,R)	$(q_4,▢,R)$
q_1	(q_1,a,R)	(q_2,Y,R)	(qrej,c,R)	(q_1,X,R)	(q_1,Y,R)	(qrej,Z,R)	(qrej,▢,R)
q_2	(qrej,a,R)	(q_2,b,R)	(q_3,Z,L)	(qrej,X,R)	(q_2,Y,R)	(q_2,Z,R)	(qrej,▢,R)
q_3	(q_3,a,L)	(q_3,b,L)	(q_3,c,L)	(q_3,X,L)	(q_3,Y,L)	(q_3,Z,L)	$(q_0,▢,R)$
q_4	(qrej,a,R)	(qrej,b,R)	(qrej,c,R)	(q_4,X,R)	(q_4,Y,R)	(q_4,Z,R)	(qacc,▢,R)

NOTE:

The “power” of the TM comes from:

- Non-local memory access (L/R movement). With a stack you cannot do it.
- Read/WRITE memory (store intermediate results on tape)
- Unbounded, revisitable memory



A Turing machine is powerful because it can treat its tape as a *rewritable infinite workspace* that it can move through arbitrarily many times.