Turing's Thesis

Turing's thesis (1930):

Any computation carried out
by mechanical means
can be performed by a Turing Machine

Algorithm:

An algorithm for a problem is a
Turing Machine which solves the problem

The algorithm describes the steps of
the mechanical means

This is easily translated to computation steps
of a Turing machine

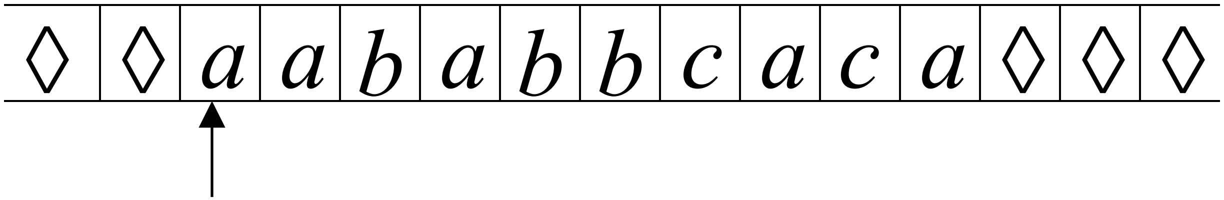
When we say: There exists an algorithm

We mean: There exists a Turing Machine that executes the algorithm

Variations of the Turing Machine

The Standard Model

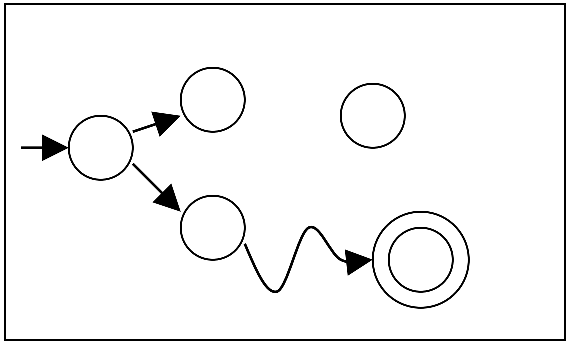
Infinite Tape



Read-Write Head

(Left or Right)

Control Unit



Deterministic

Variations of the Standard Model

Turing machines with:

- Stay-Option
- Semi-Infinite Tape
- Off-Line
- Multitape
- Multidimensional
- Nondeterministic

Different Turing Machine **Classes**

Same Power of two machine classes:

both classes accept the same set of languages

We will prove:

each new class has the same power with Standard Turing Machine

(accept Turing-Recognizable Languages)

Same Power of two classes means:

for any machine M_1 of first class

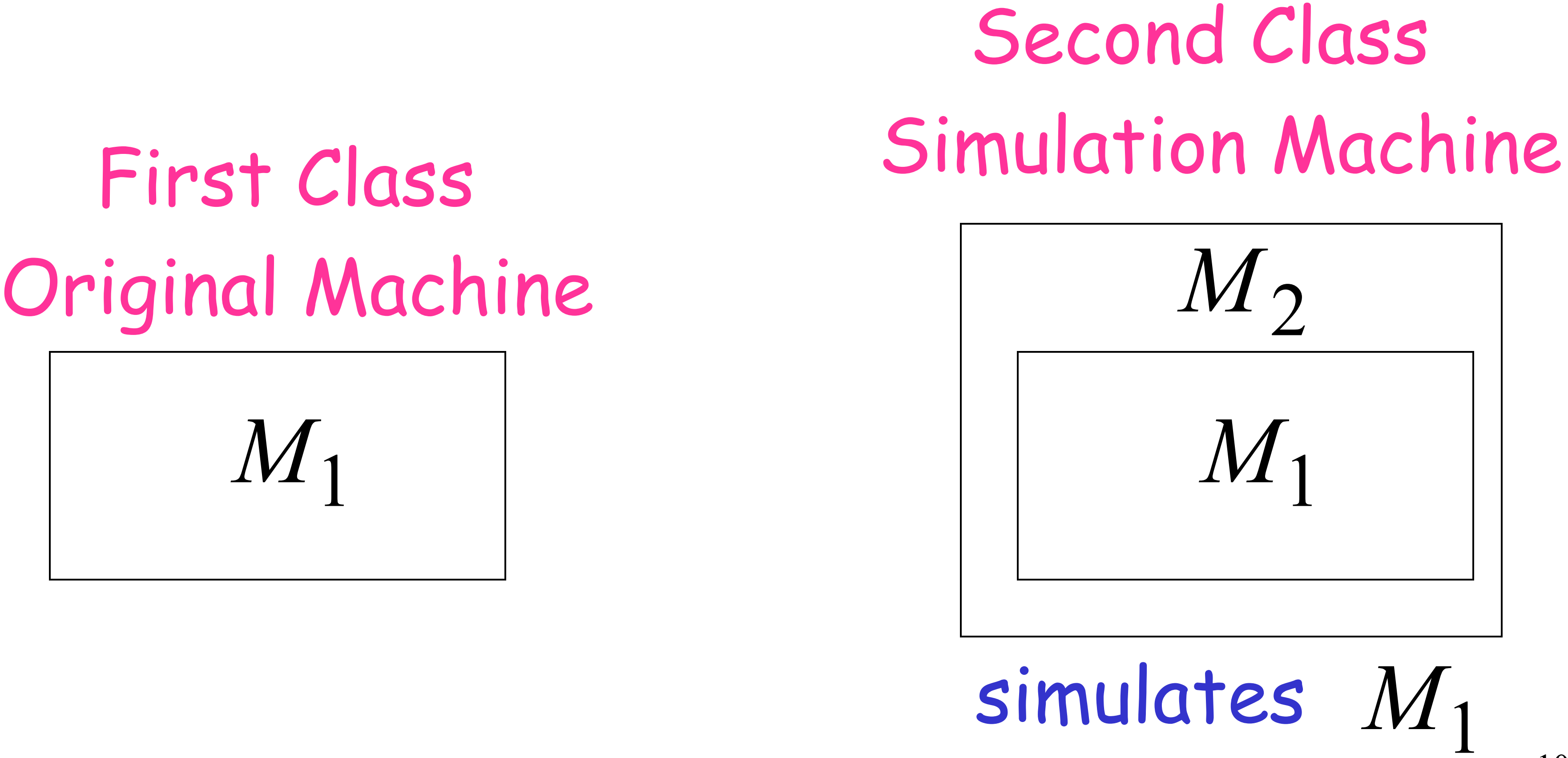
there is a machine M_2 of second class

such that: $L(M_1) = L(M_2)$

and vice-versa

Simulation: A technique to prove same power.

Simulate the machine of one class with a machine of the other class



First Class
Original Machine

M_1

Second Class
Simulation Machine

M_2

M_1

simulates M_1

Configurations in the Original Machine M_1 have corresponding configurations in the Simulation Machine M_2

$$\begin{tikzcd}
 & & & M_1 & & & & \\
\text{Original Machine:} & d_0 \arrow[d, leftrightarrow] & \succ & d_1 \arrow[d, leftrightarrow] & \succ & \cdots & \succ & d_n \arrow[d, leftrightarrow] \\
\text{Simulation Machine:} & d'_0 & \overset{*}{\succ} & d'_1 & \overset{*}{\succ} & \cdots & \overset{*}{\succ} & d'_n \\
 & & & M_2 & & & &
\end{tikzcd}$$

Accepting Configuration

Original Machine: d_f

↕

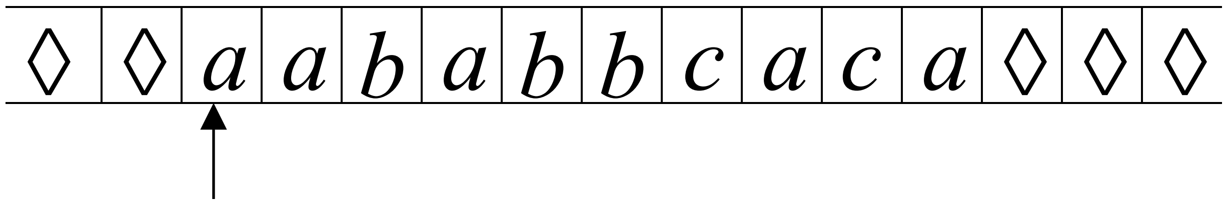
Simulation Machine: d'_f

the Simulation Machine
and the Original Machine
accept the same strings

$$L(M_1) = L(M_2)$$

Turing Machines with Stay-Option

The head can stay in the same position



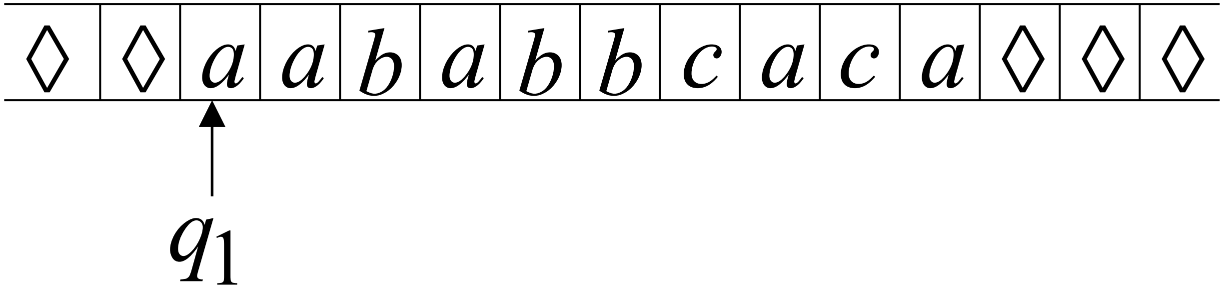
◊	◊	a	a	b	a	b	b	c	a	c	a	◊	◊	◊

Left, Right, Stay

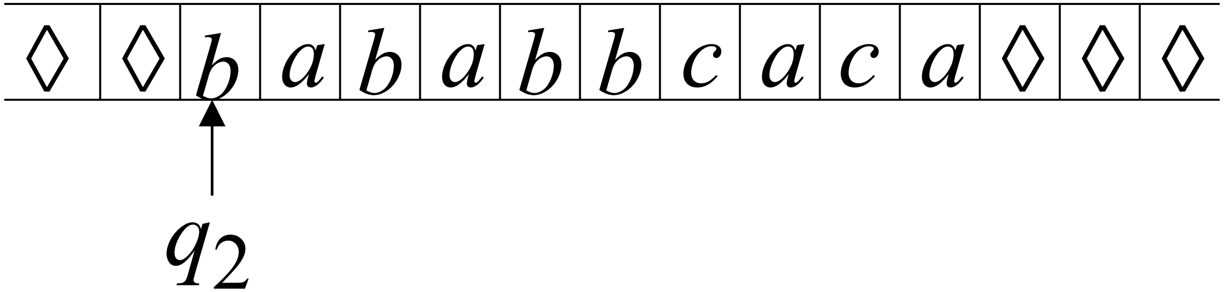
L,R,S: possible head moves

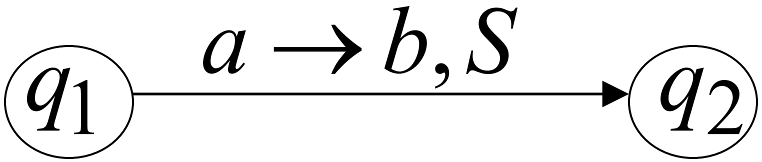
Example:

Time 1



Time 2





Theorem: Stay-Option machines
have the same power with
Standard Turing machines

Proof: 1. Stay-Option Machines
simulate Standard Turing machines

2. Standard Turing machines
simulate Stay-Option machines

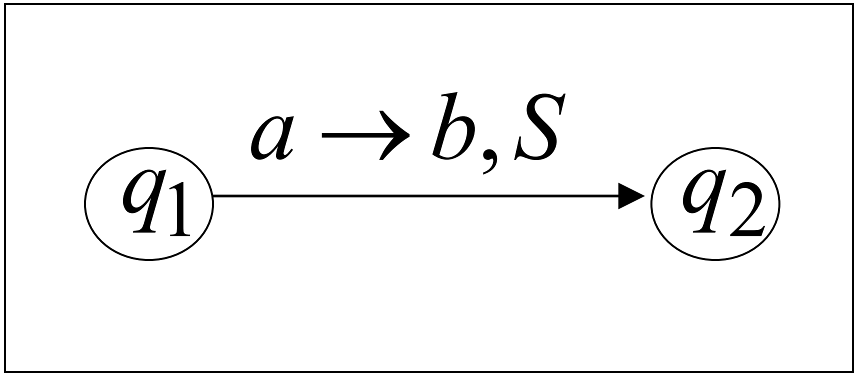
1. Stay-Option Machines simulate Standard Turing machines

Trivial: any standard Turing machine
is also a Stay-Option machine

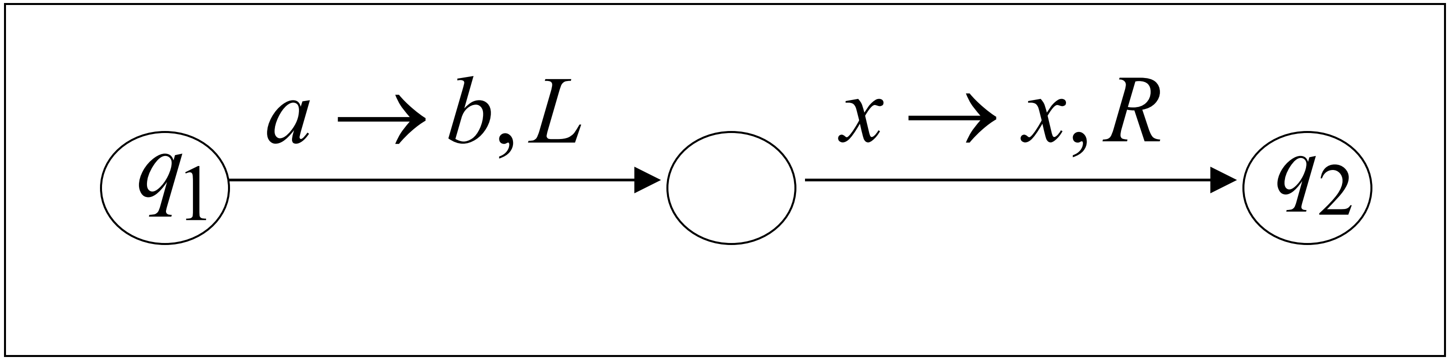
2. Standard Turing machines simulate Stay-Option machines

We need to simulate the stay head option with two head moves, one left and one right

Stay-Option Machine



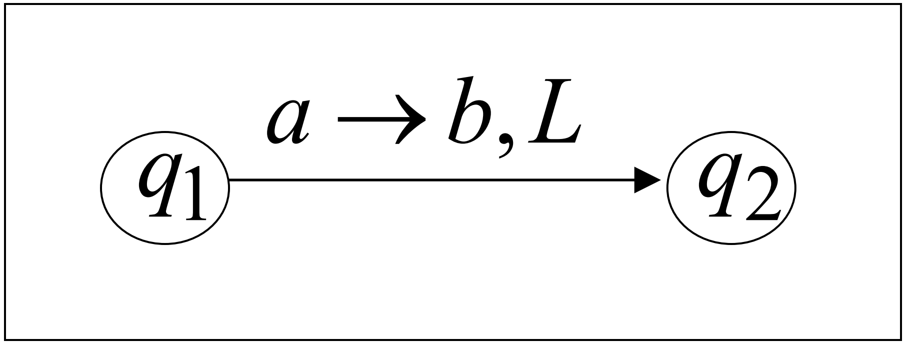
Simulation in Standard Machine



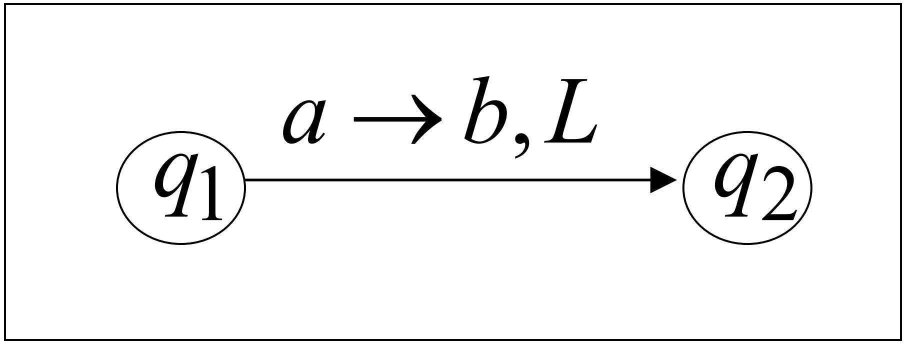
For every possible tape symbol x

For other transitions nothing changes

Stay-Option Machine



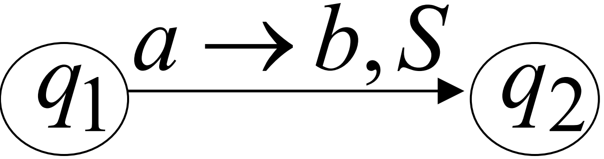
Simulation in Standard Machine

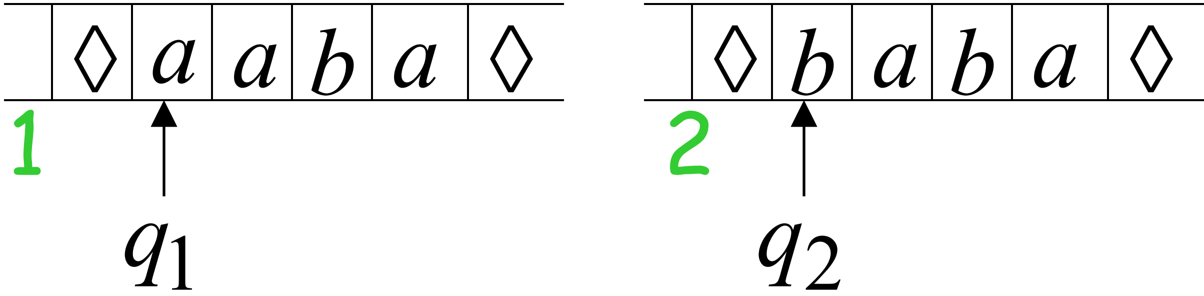


Similar for Right moves

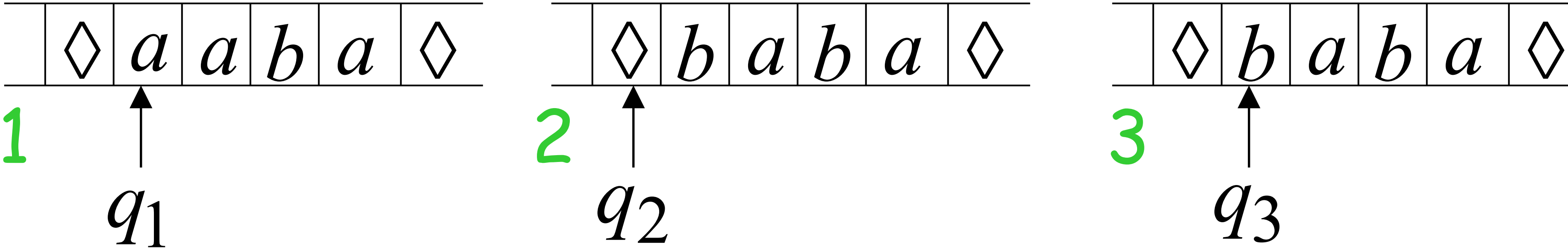
example of simulation

Stay-Option Machine:





Simulation in Standard Machine:

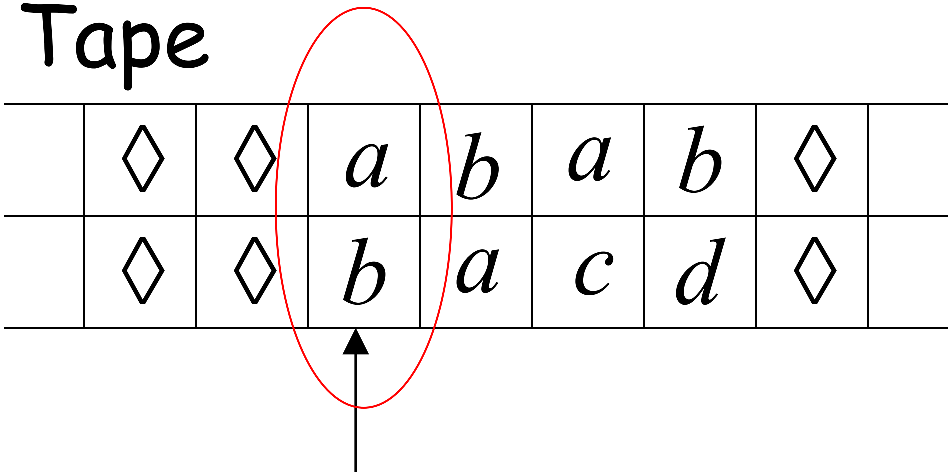


END OF PROOF

Multiple Track Tape

A useful trick to perform more complicated simulations

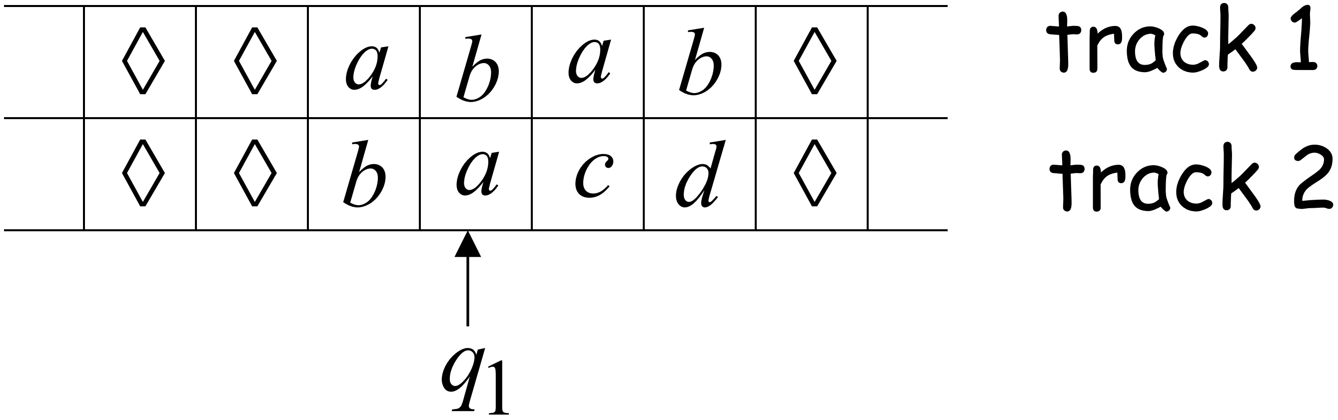
One Tape

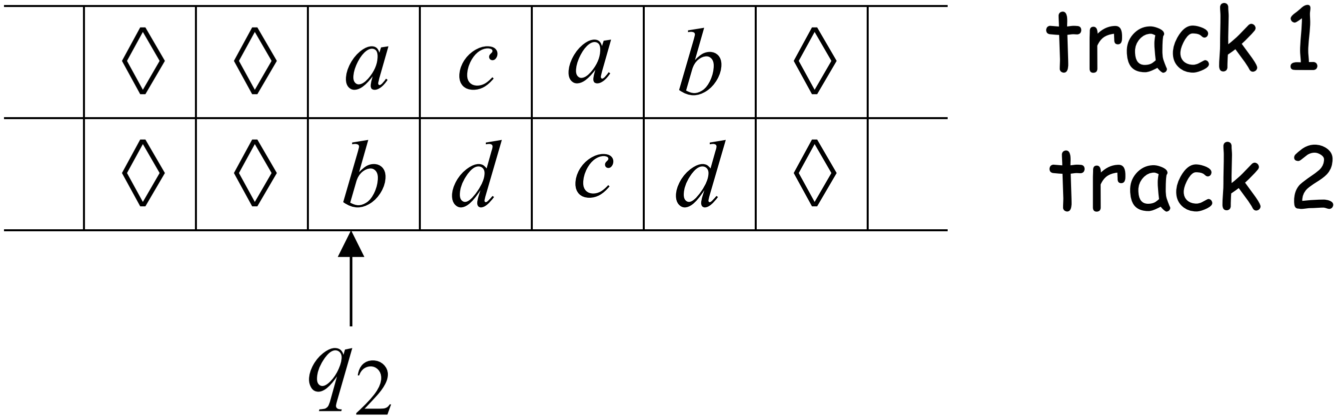


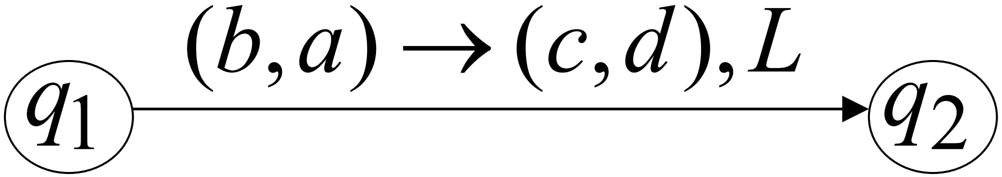
	◊	◊	a	b	a	b	◊		track 1
	◊	◊	b	a	c	d	◊		track 2

One head

One symbol (a,b)

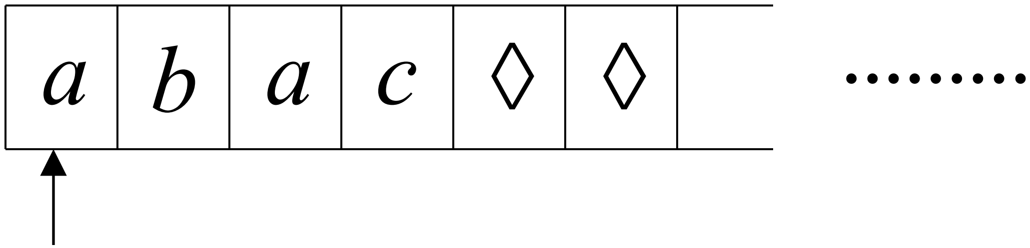

◊ ◊ a b a b ◊
◊ ◊ b a c d ◊
track 1
track 2
q_1


◊ ◊ a c a b ◊
◊ ◊ b d c d ◊
track 1
track 2
q_2


q_1
$(b,a) \rightarrow (c,d), L$
q_2

Semi-Infinite Tape

The head extends infinitely only to the right



- Initial position is the leftmost cell

- When the head moves left from the border, it returns to the same position

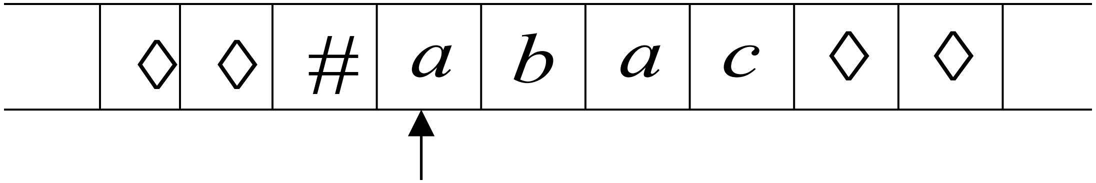
Theorem: Semi-Infinite machines have the same power with Standard Turing machines

Proof:

1. Standard Turing machines simulate Semi-Infinite machines

2. Semi-Infinite Machines simulate Standard Turing machines

1. Standard Turing machines simulate Semi-Infinite machines:



Standard Turing Machine

a. insert special symbol #

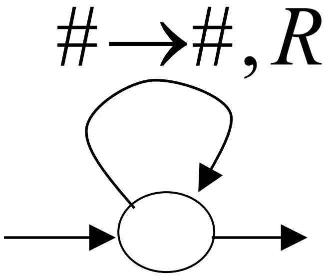
at left of input string

b. Add a self-loop

to every state

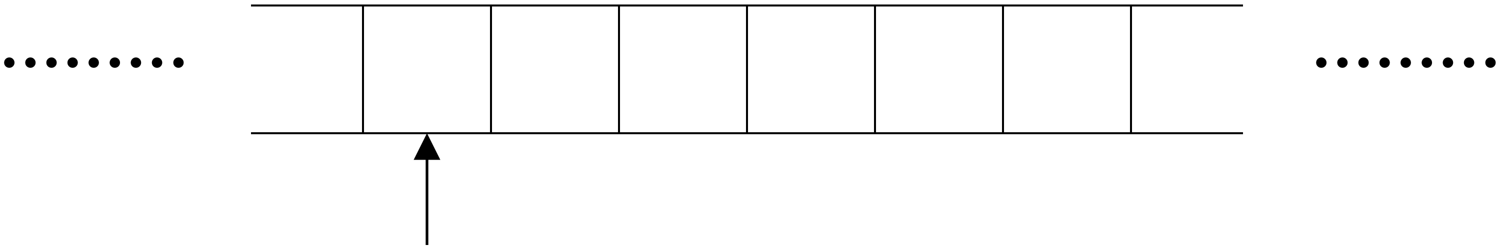
(except states with no

outgoing transitions)

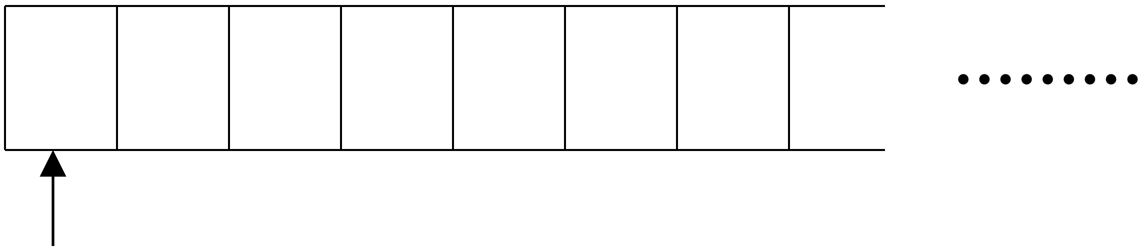


2. Semi-Infinite tape machines simulate Standard Turing machines:

Standard machine

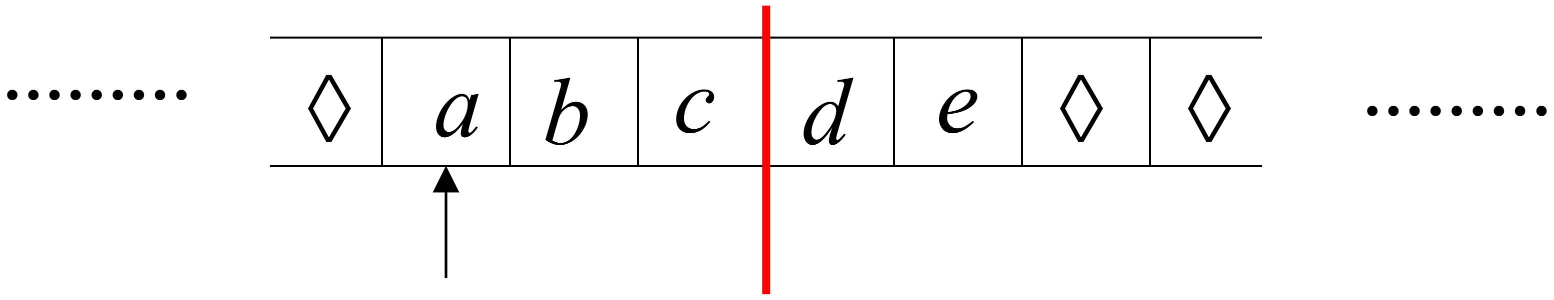


Semi-Infinite tape machine



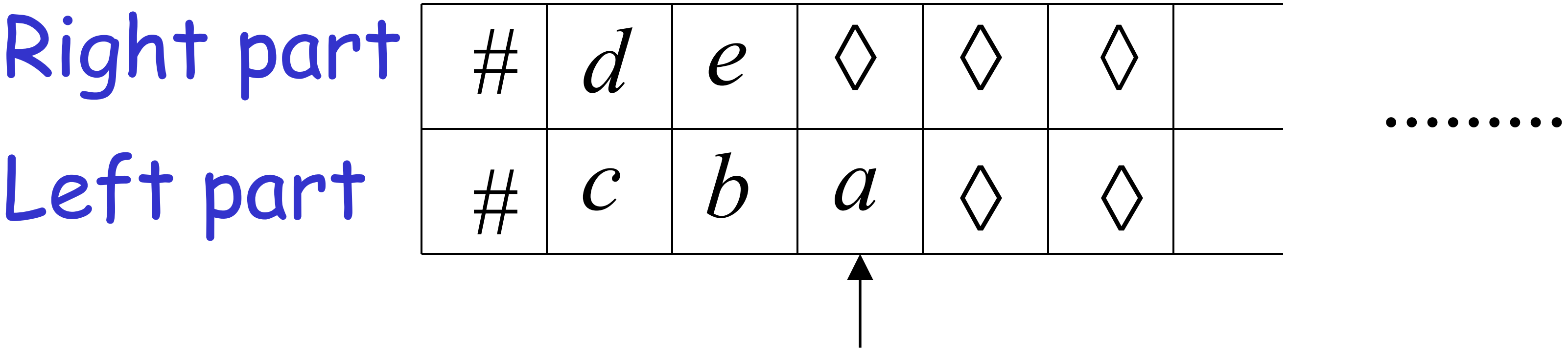
Squeeze infinity of both directions in one direction

Standard machine

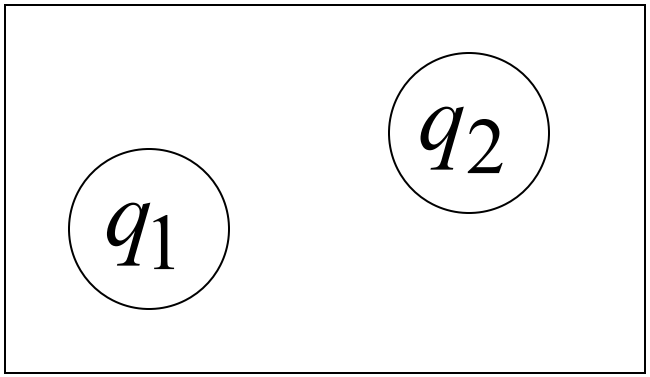


reference point

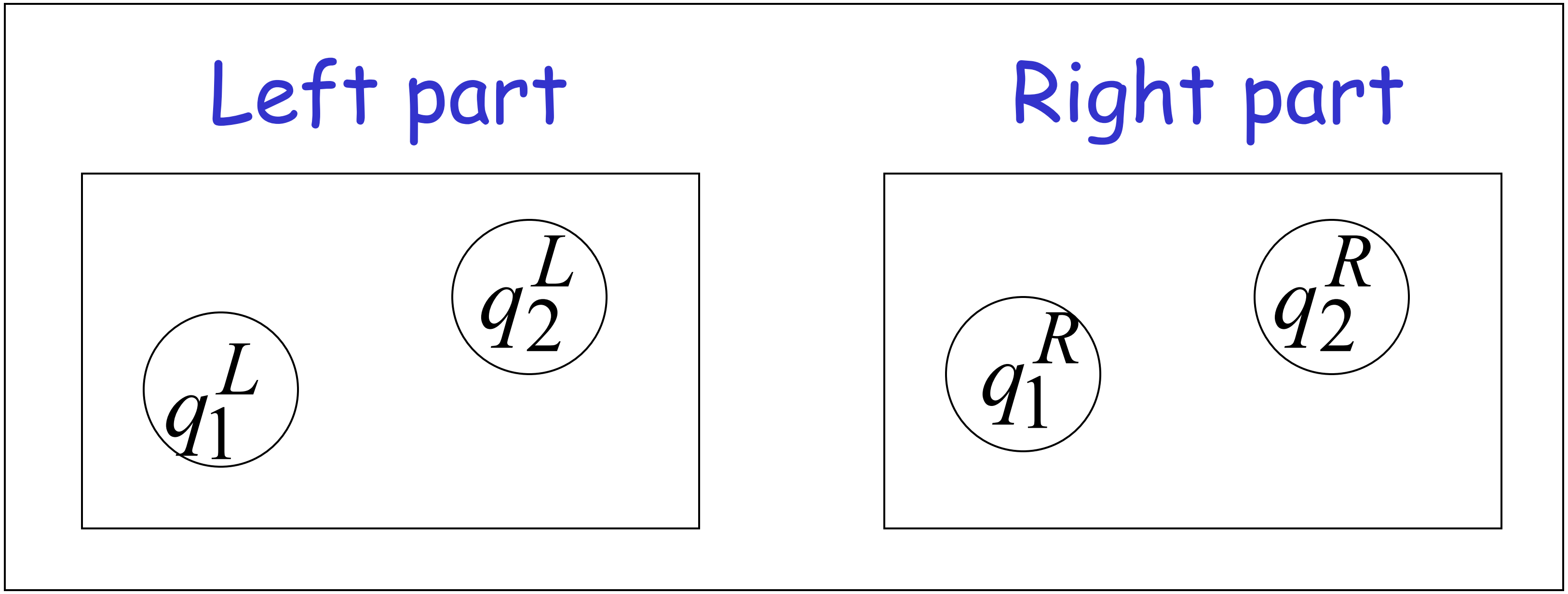
Semi-Infinite tape machine with two tracks



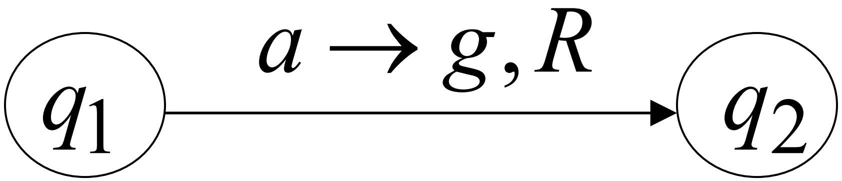
Standard machine



Semi-Infinite tape machine

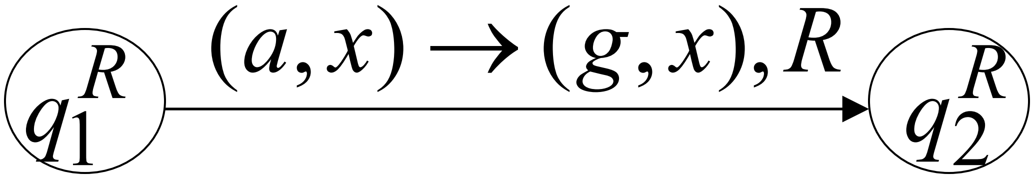


Standard machine

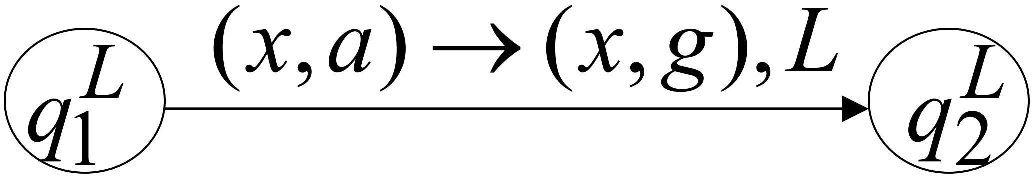


Semi-Infinite tape machine

Right part



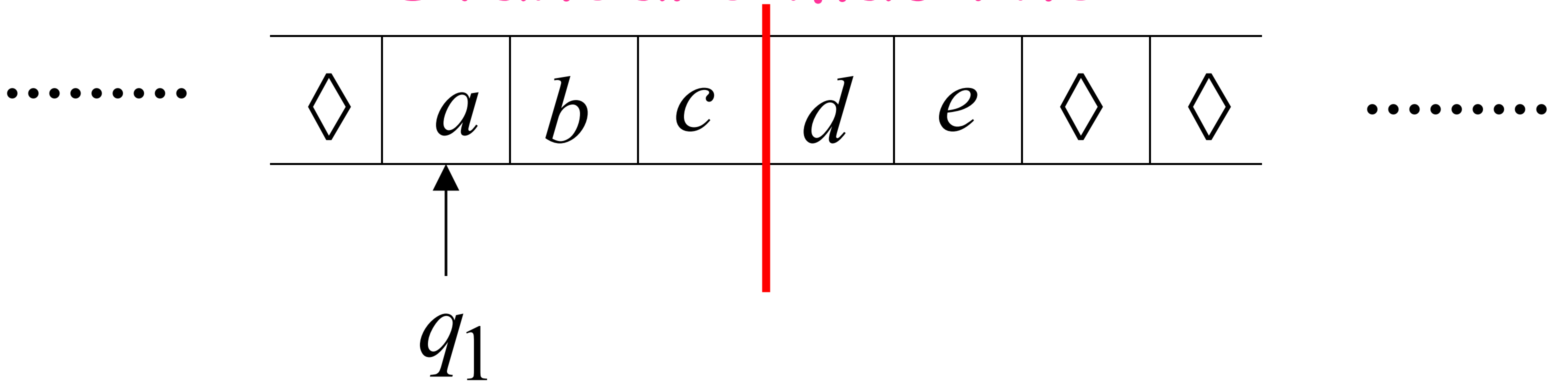
Left part



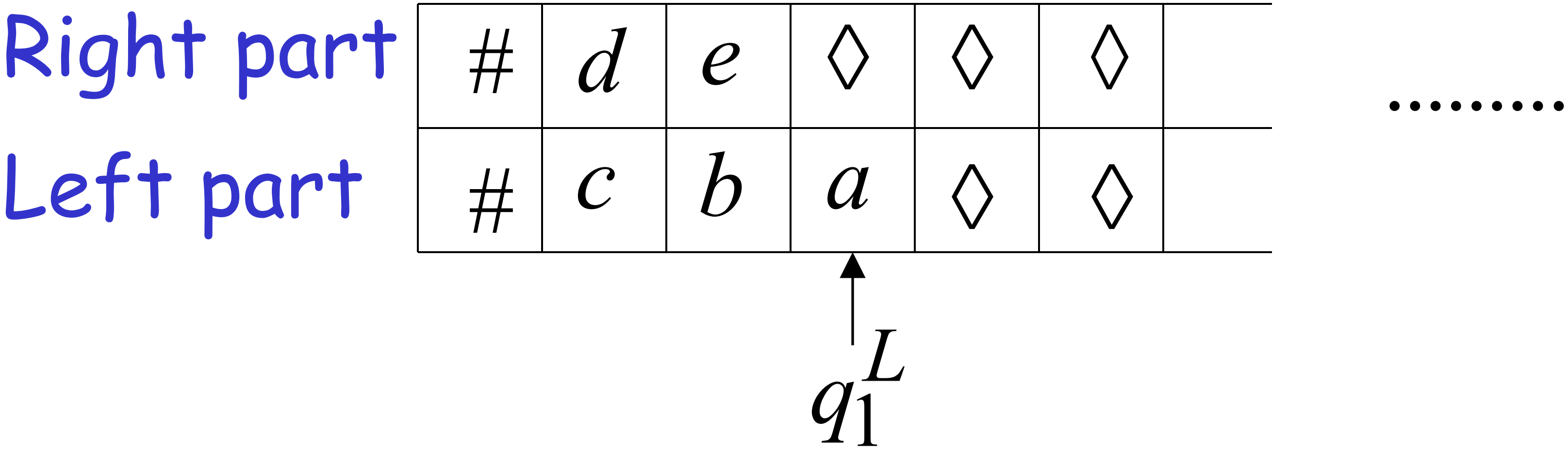
For all tape symbols x

Time 1

Standard machine

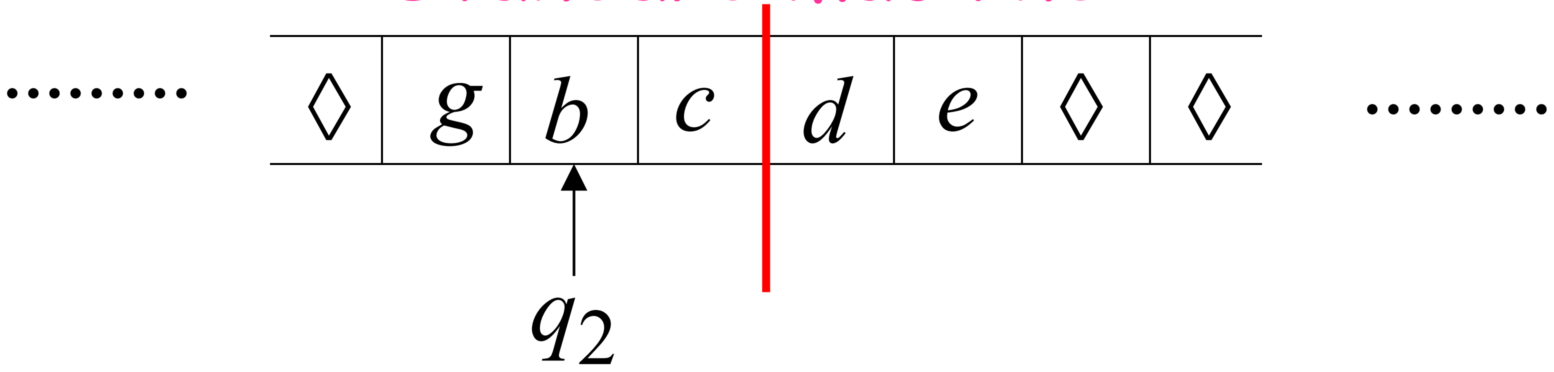


Semi-Infinite tape machine



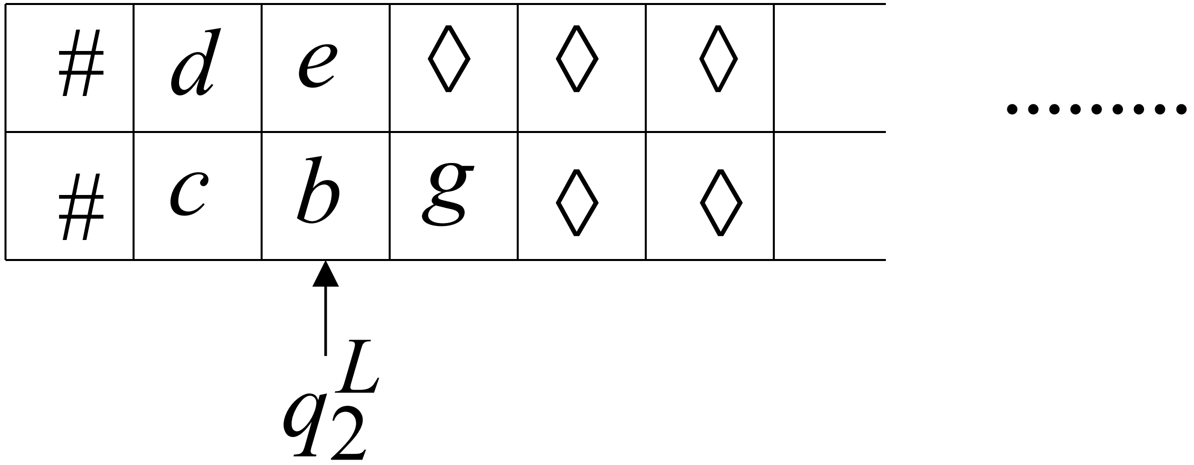
Time 2

Standard machine



………	◊	g	b	c	d	e	◊	◊	………

Semi-Infinite tape machine

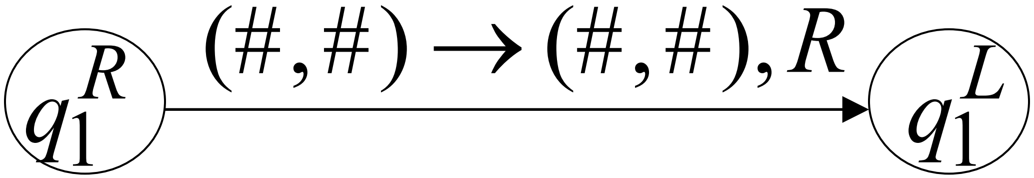


Right part	#	d	e	◊	◊	◊		………
Left part	#	c	b	g	◊	◊		

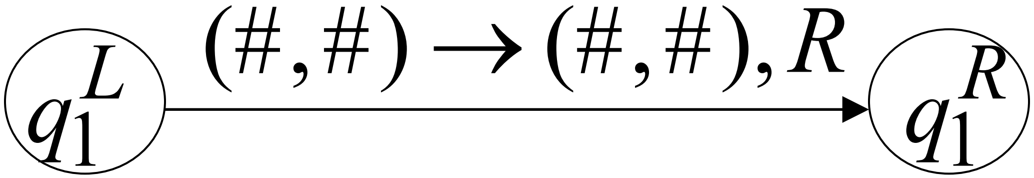
At the border:

Semi-Infinite tape machine

Right part

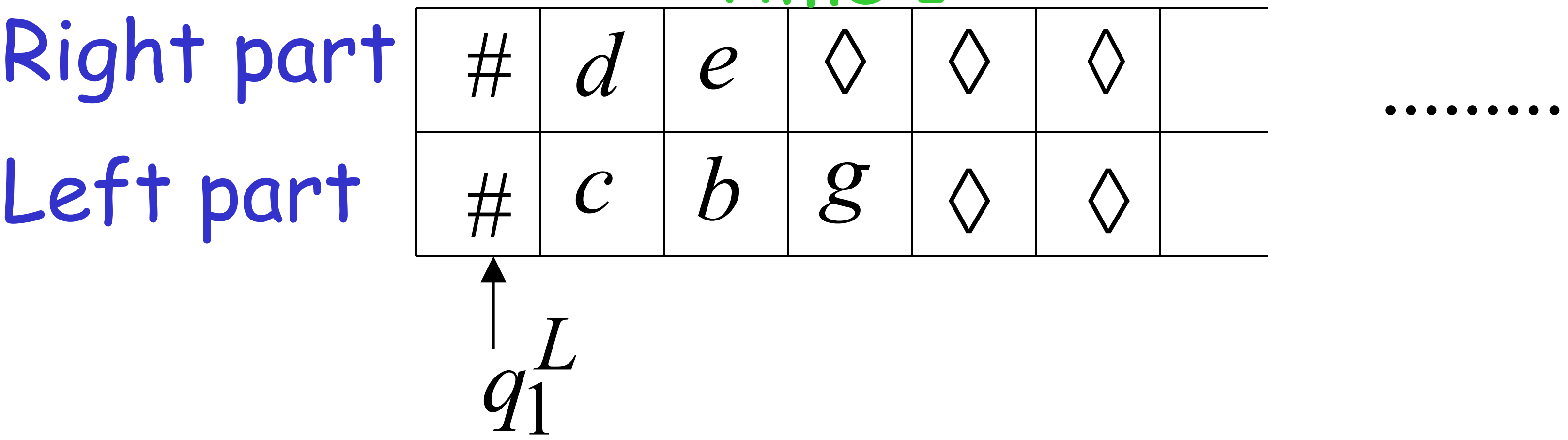


Left part



Semi-Infinite tape machine

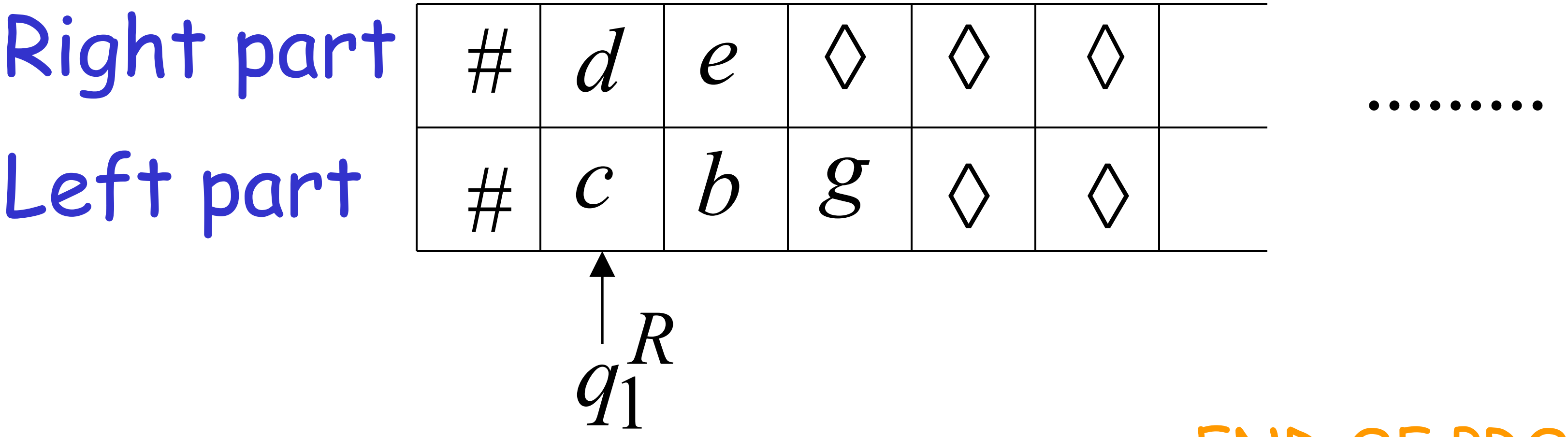
Time 1



Right part	#	d	e	◊	◊	◊		....
Left part	#	c	b	g	◊	◊		

q_1^L

Time 2

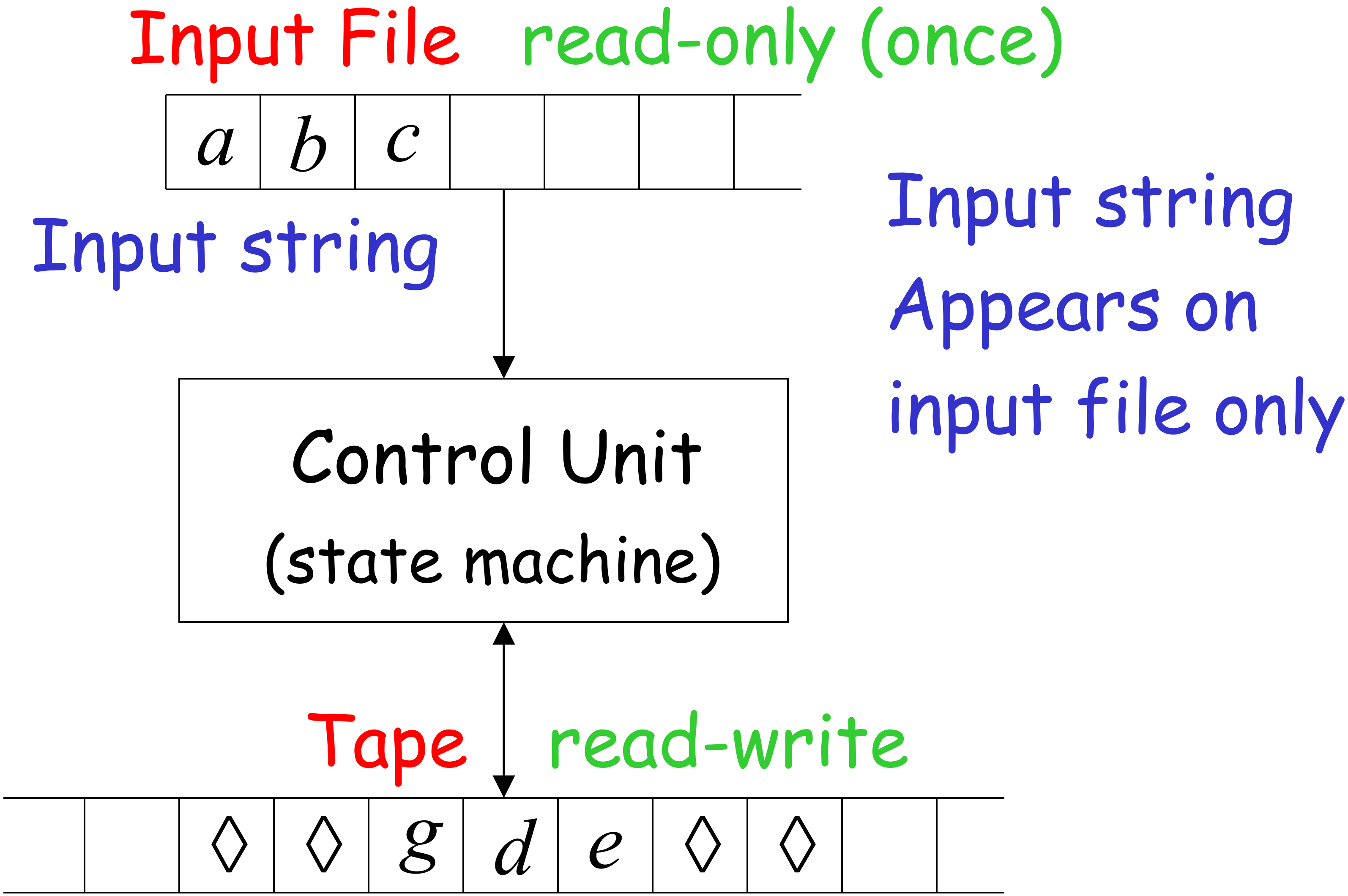


Right part	#	d	e	◊	◊	◊		....
Left part	#	c	b	g	◊	◊		

q_1^R

END OF PROOF

The Off-Line Machine



Input File read-only (once)

a	b	c				

Input string

Control Unit
(state machine)

Tape read-write

		◊	◊	g	d	e	◊	◊		

Input string
Appears on
input file only

Theorem: Off-Line machines
have the same power with
Standard Turing machines

Proof: 1. Off-Line machines
simulate Standard Turing machines

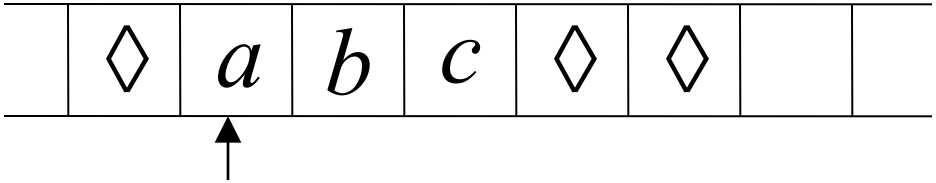
2. Standard Turing machines
simulate Off-Line machines

1. Off-line machines simulate Standard Turing Machines

Off-line machine:

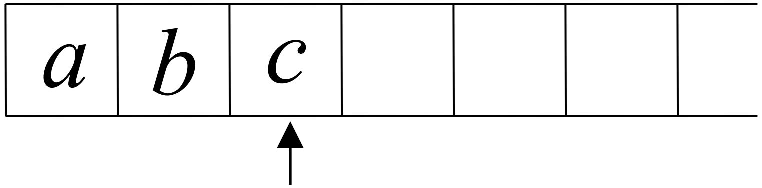
1. Copy input file to tape
2. Continue computation as in Standard Turing machine

Standard machine

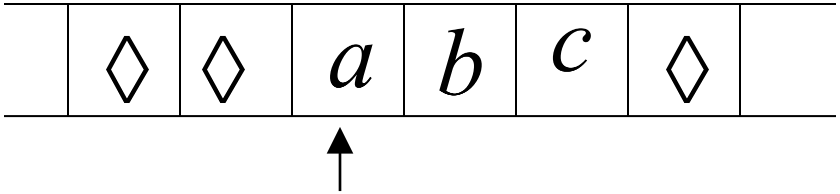


Off-line machine

Input File

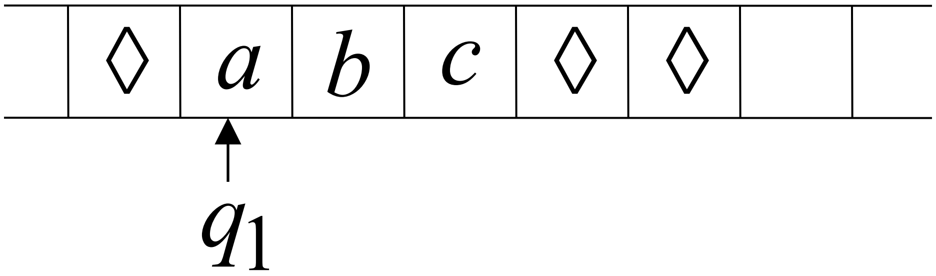


Tape



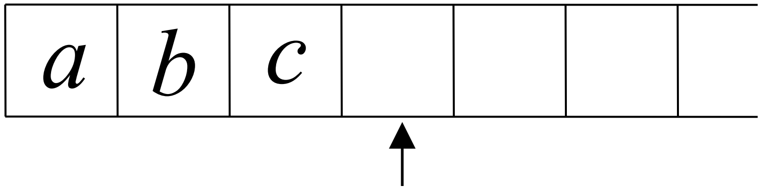
1. Copy input file to tape

Standard machine

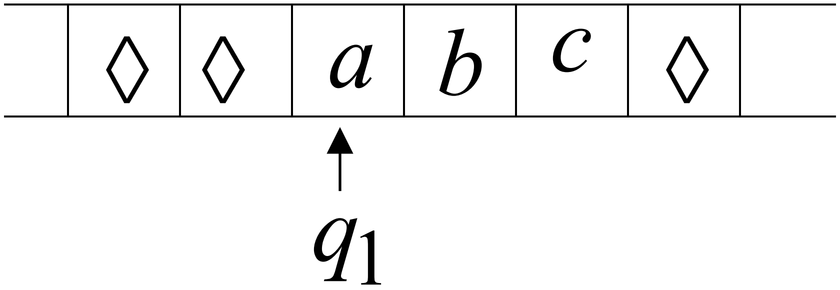


Off-line machine

Input File



Tape



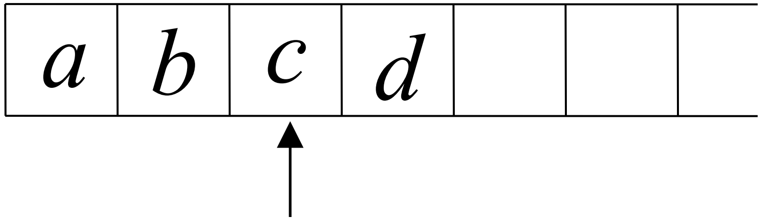
2. Do computations as in Turing machine

2. Standard Turing machines simulate Off-Line machines:

Use a Standard machine with
a four-track tape to keep track of
the Off-line input file and tape contents

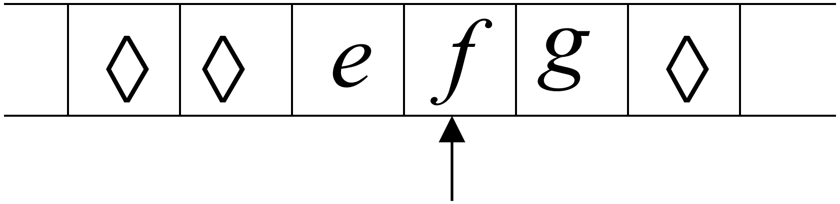
Off-line Machine

Input File



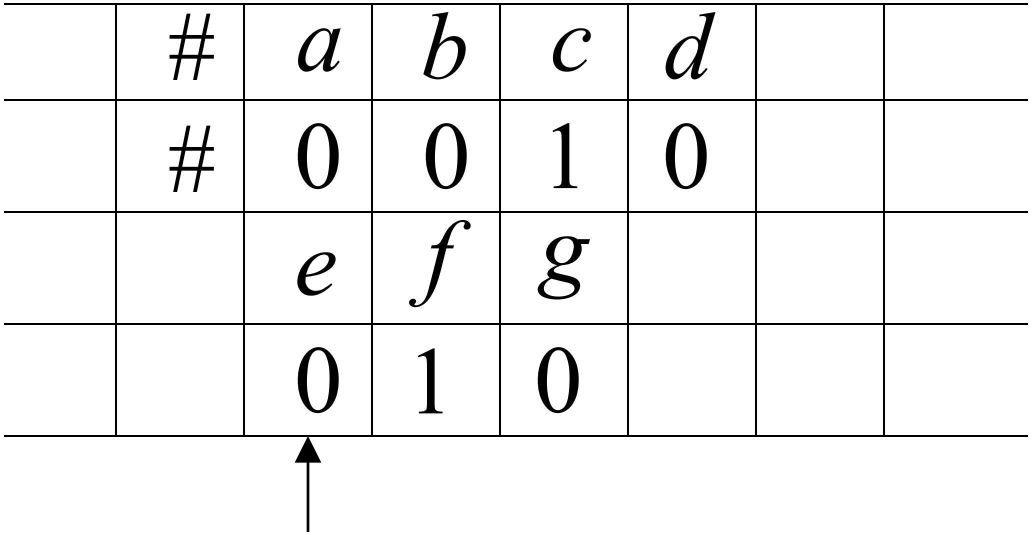
a	b	c	d			

Tape



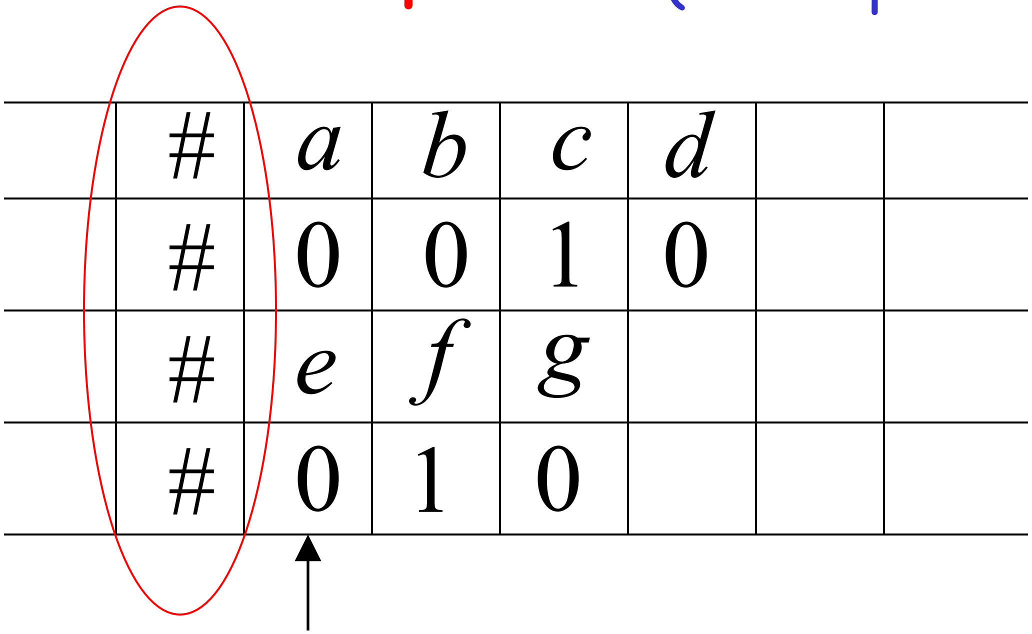
	◊	◊	e	f	g	◊	

Standard Machine -- Four track tape



	#	a	b	c	d			Input File
	#	0	0	1	0			head position
		e	f	g				Tape
		0	1	0				head position

Reference point (uses special symbol #)



	#	a	b	c	d		
	#	0	0	1	0		
	#	e	f	g			
	#	0	1	0			

Input File

head position

Tape

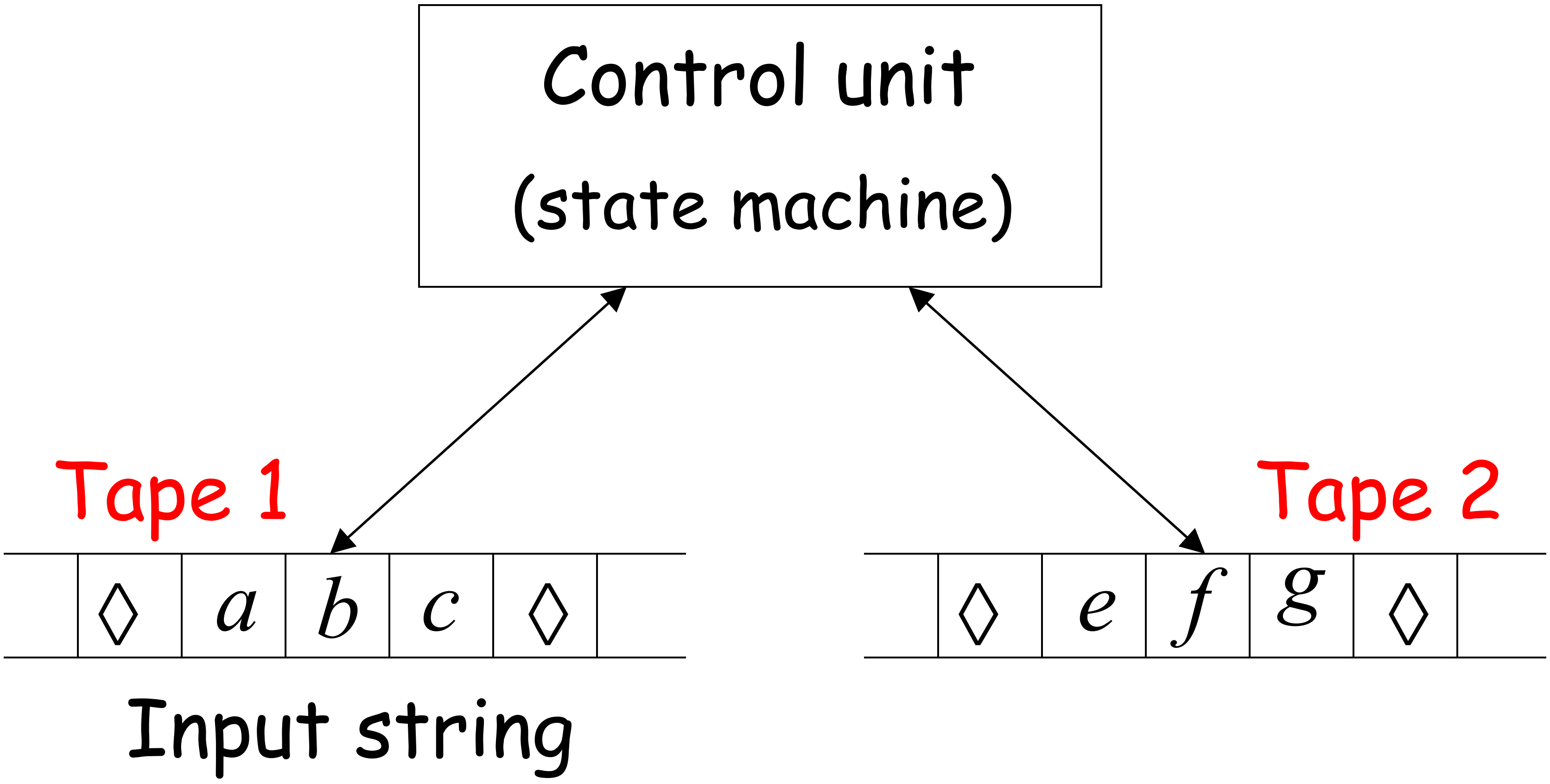
head position

Repeat for each state transition:

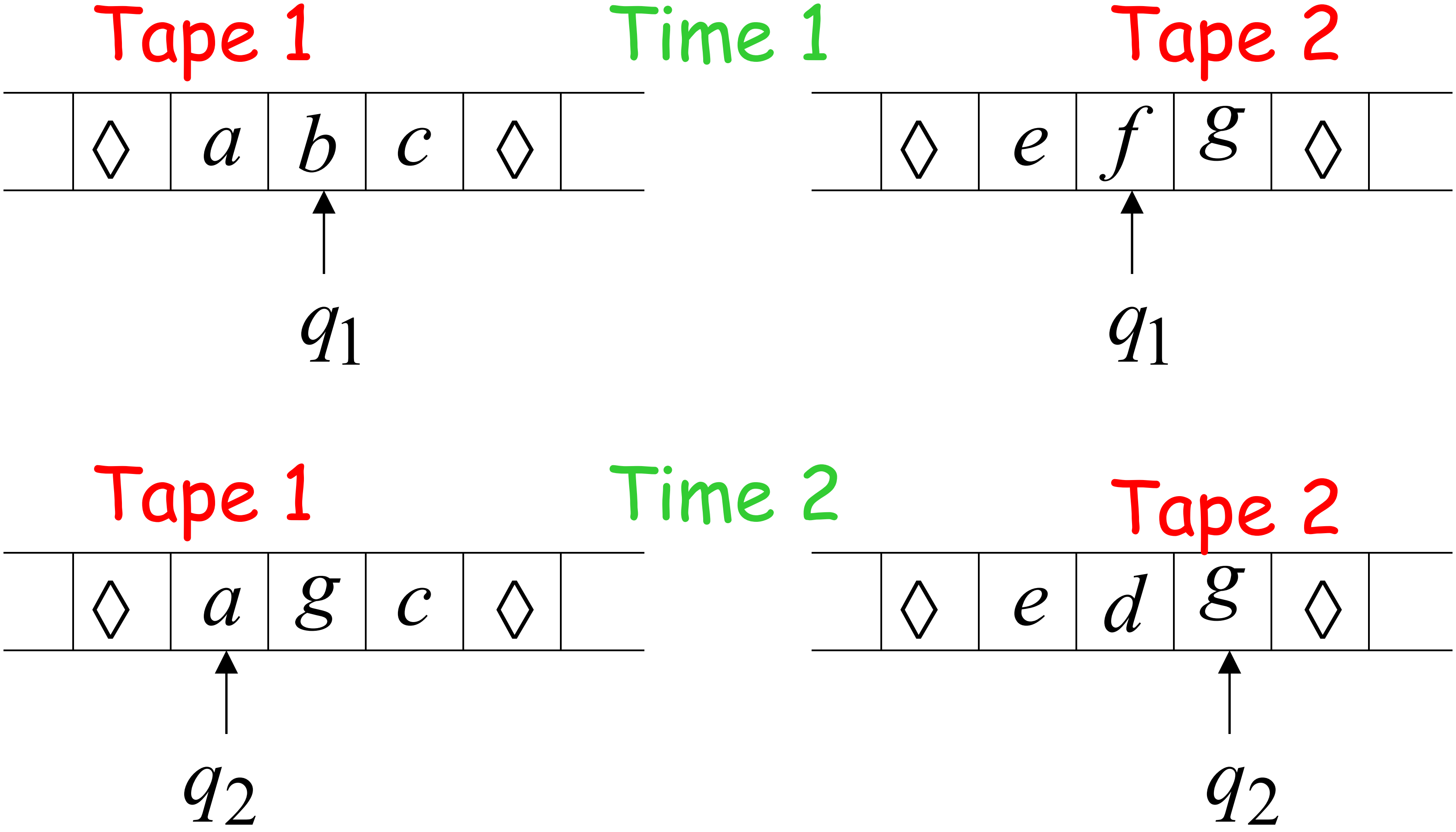
1. Return to reference point
2. Find current input file symbol
3. Find current tape symbol
4. Make transition

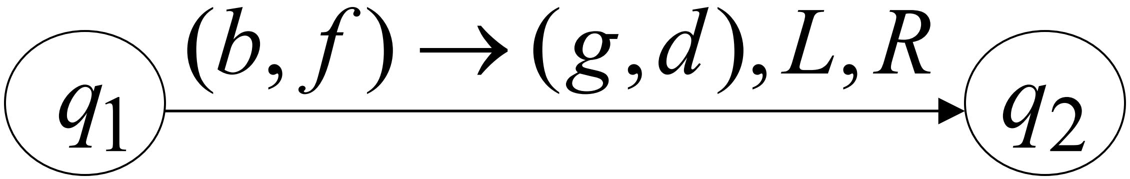
END OF PROOF

Multi-tape Turing Machines



Input string appears on Tape 1


Tape 1
Time 1
Tape 2
◊ a b c ◊
◊ e f g ◊
q_1
q_1
Tape 1
Time 2
Tape 2
◊ a g c ◊
◊ e d g ◊
q_2
q_2


q_1
$(b, f) \rightarrow (g, d), L, R$
q_2

Theorem: Multi-tape machines have the same power with Standard Turing machines

Proof: 1. Multi-tape machines simulate Standard Turing machines

2. Standard Turing machines simulate Multi-tape machines

1. Multi-tape machines simulate Standard Turing Machines:

Trivial: Use just one tape

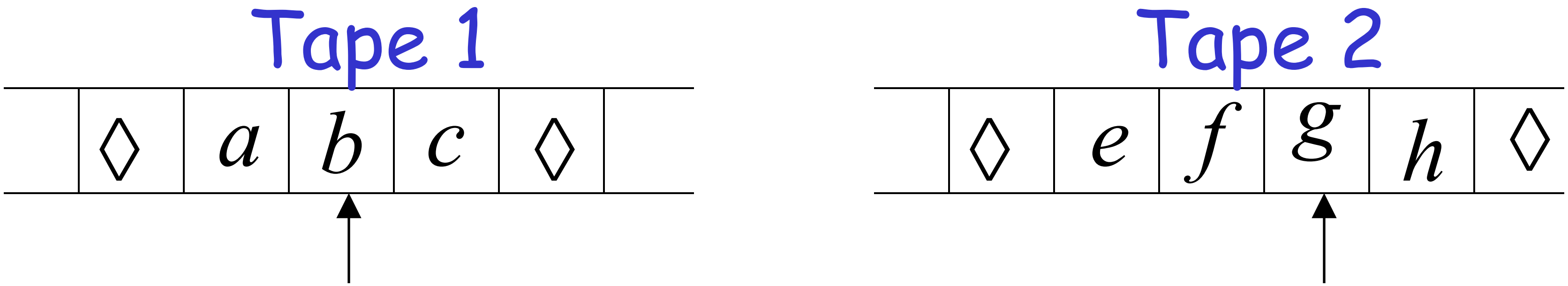
2. Standard Turing machines simulate Multi-tape machines:

Standard machine:

- Uses a multi-track tape to simulate the multiple tapes

- A tape of the Multi-tape machine corresponds to a pair of tracks

Multi-tape Machine



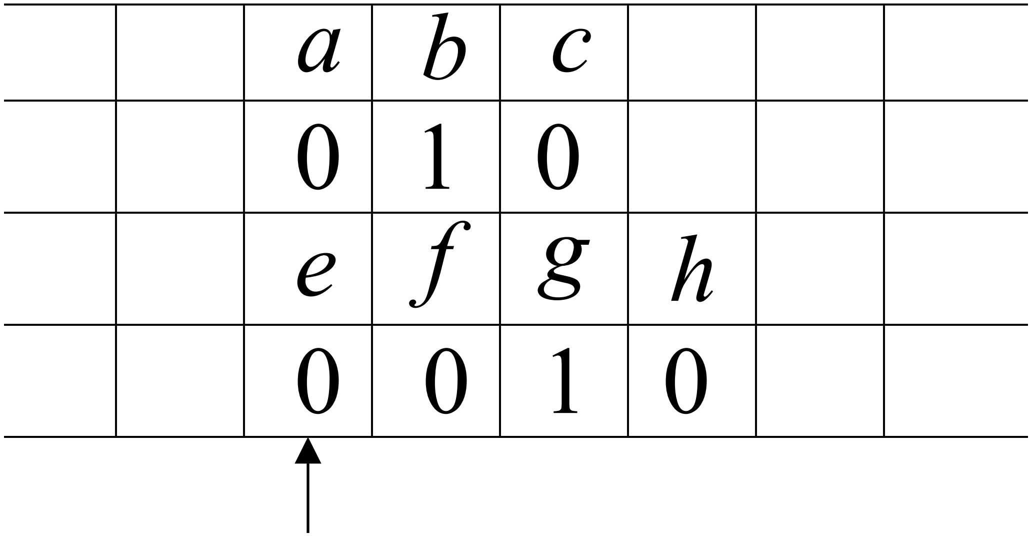
Tape 1

	◊	a	b	c	◊	

Tape 2

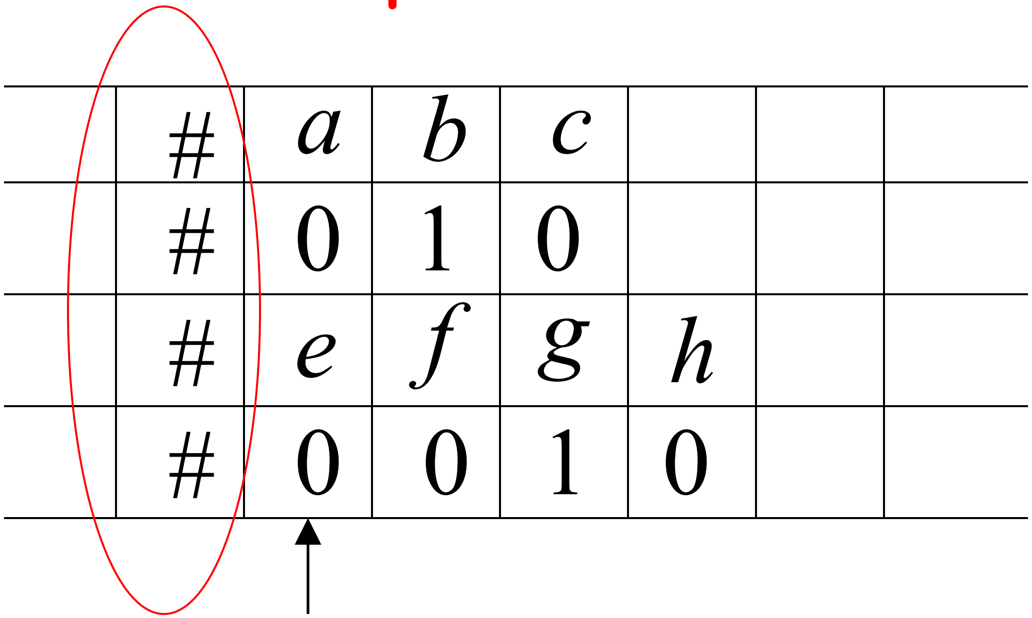
	◊	e	f	g	h	◊

Standard machine with four track tape



		a	b	c				Tape 1
		0	1	0				head position
		e	f	g	h			Tape 2
		0	0	1	0			head position

Reference point



	#	a	b	c				Tape 1
	#	0	1	0				head position
	#	e	f	g	h			Tape 2
	#	0	0	1	0			head position

Repeat for each state transition:

1. Return to reference point
2. Find current symbol in Tape 1
3. Find current symbol in Tape 2
4. Make transition

END OF PROOF

Same power doesn't imply same speed:

$$L = \{a^n b^n\}$$

Standard Turing machine: $O(n^2)$ time

Go back and forth $O(n^2)$ times
to match the a's with the b's

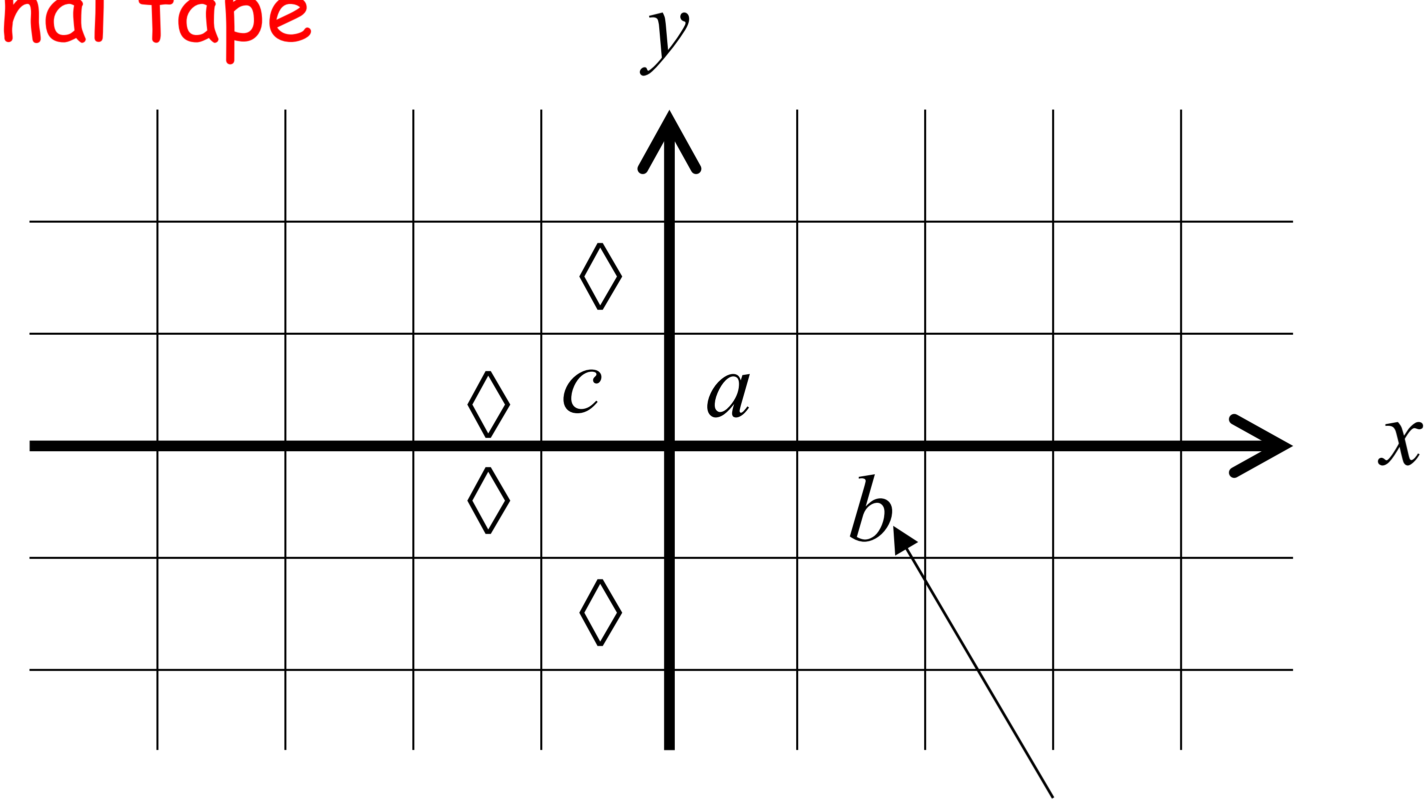
2-tape machine: $O(n)$ time

1. Copy b^n to tape 2 $(O(n)$ steps)

2. Compare a^n on tape 1
and b^n tape 2 $(O(n)$ steps)

Multidimensional Turing Machines

2-dimensional tape



MOVES: L,R,U,D

U: up D: down

HEAD

Position: +2, -1

Theorem: Multidimensional machines have the same power with Standard Turing machines

Proof: 1. Multidimensional machines simulate Standard Turing machines

2. Standard Turing machines simulate Multi-Dimensional machines

1. Multidimensional machines simulate Standard Turing machines

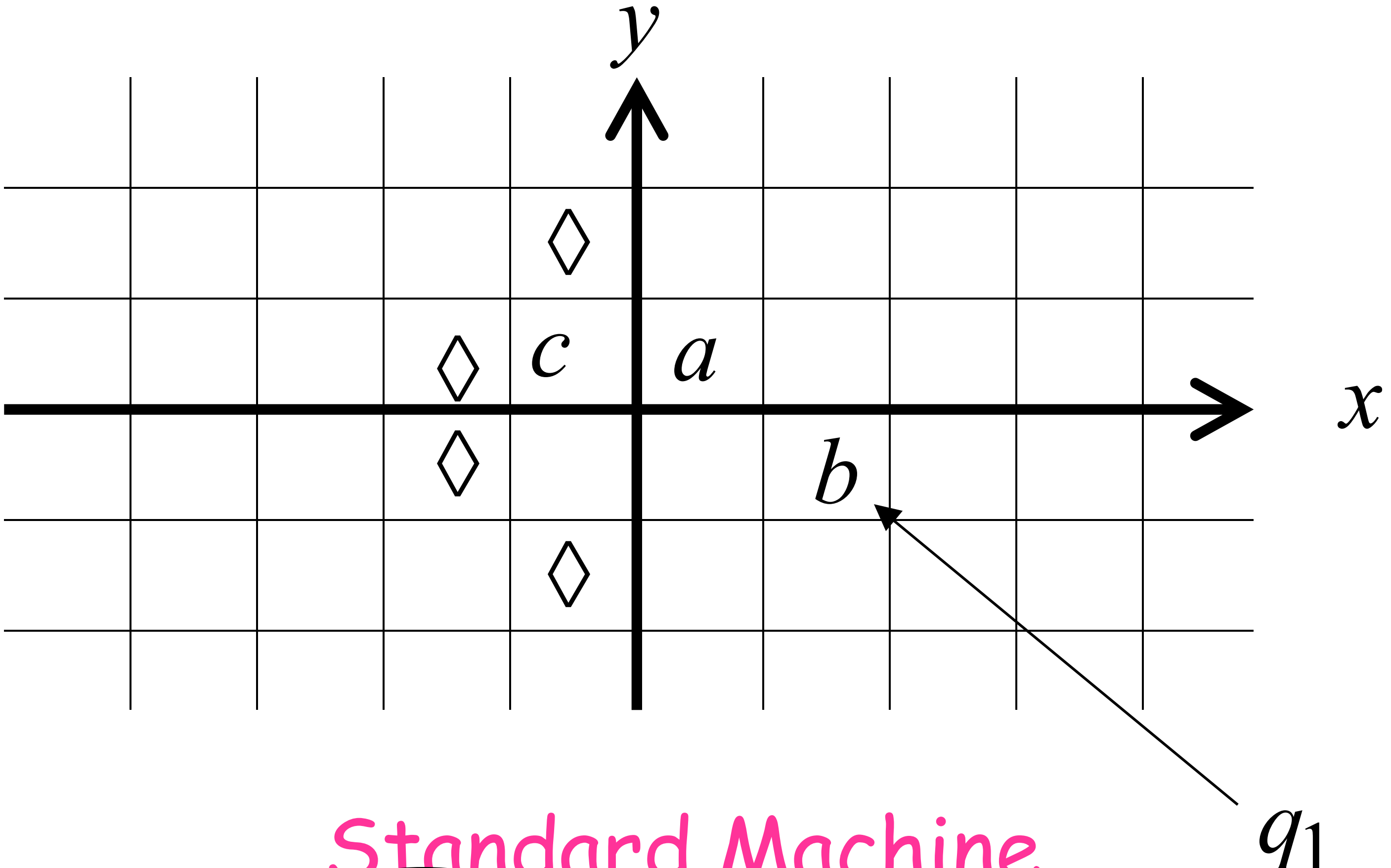
Trivial: Use one dimension

2. Standard Turing machines simulate Multidimensional machines

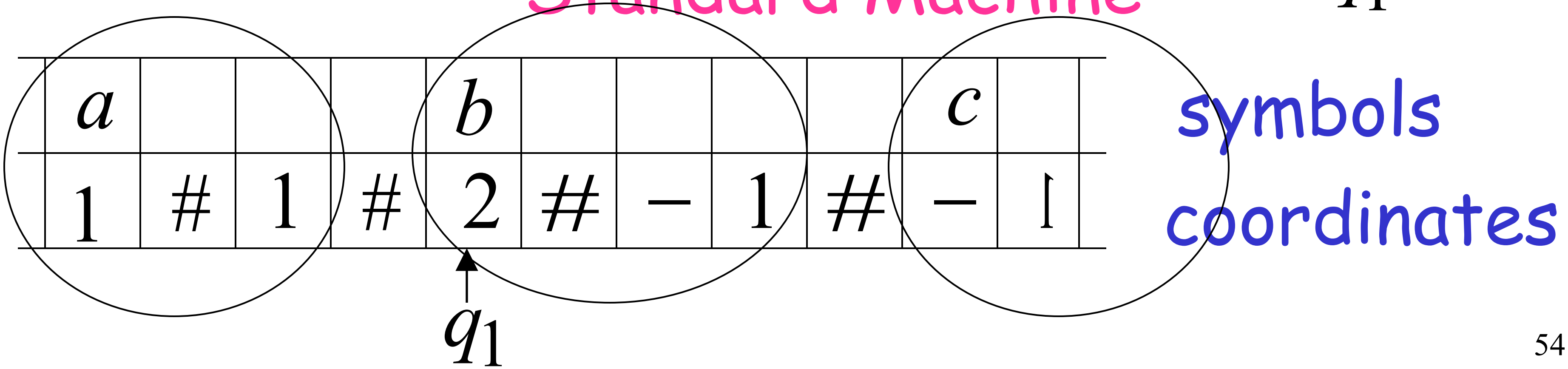
Standard machine:

- Use a two track tape
- Store symbols in track 1
- Store coordinates in track 2

2-dimensional machine



Standard Machine



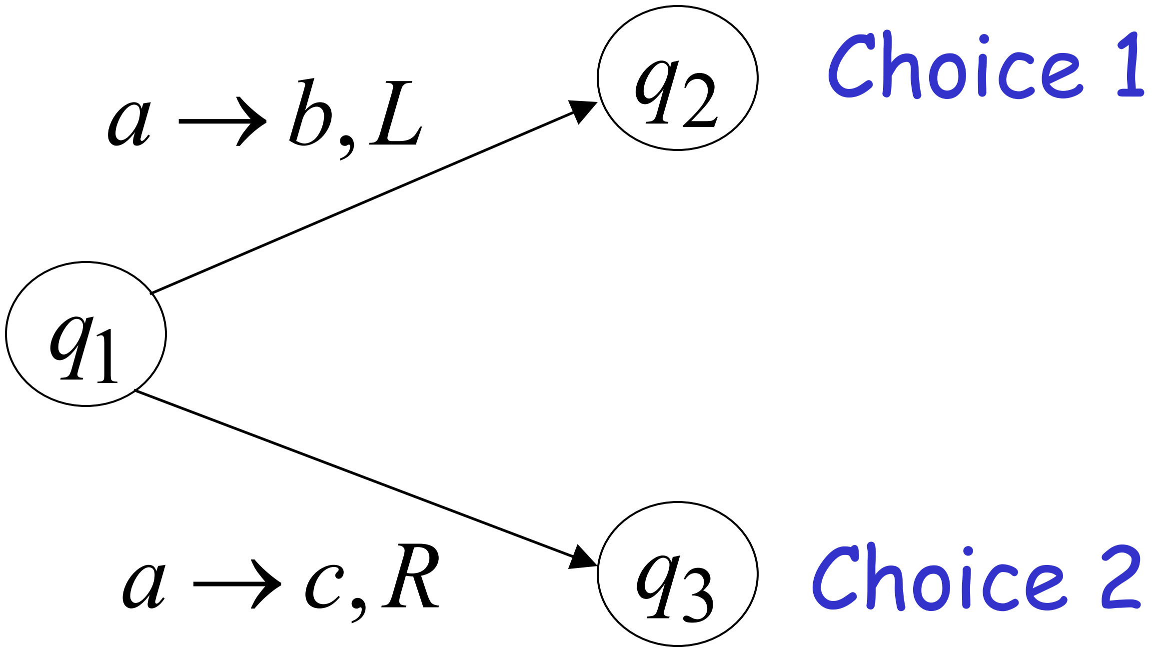
Standard machine:

Repeat for each transition followed in the 2-dimensional machine:

1. Update current symbol
2. Compute coordinates of next position
3. Go to new position

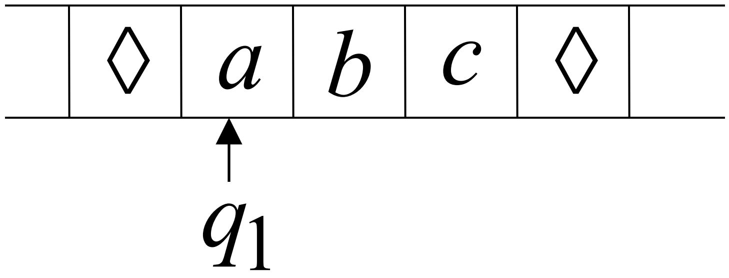
END OF PROOF

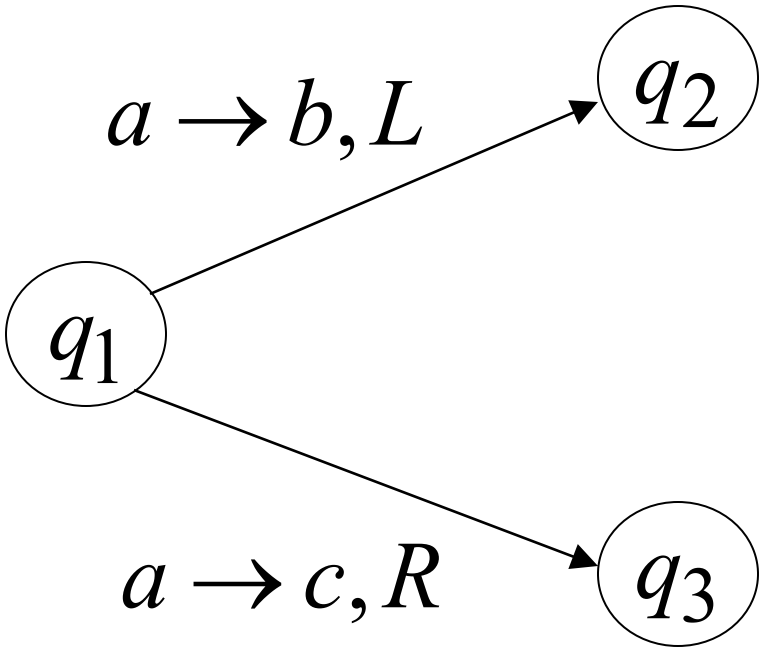
Nondeterministic Turing Machines



Allows Non Deterministic Choices

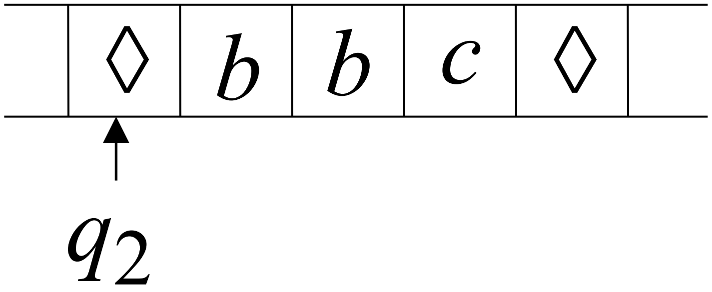
Time 0



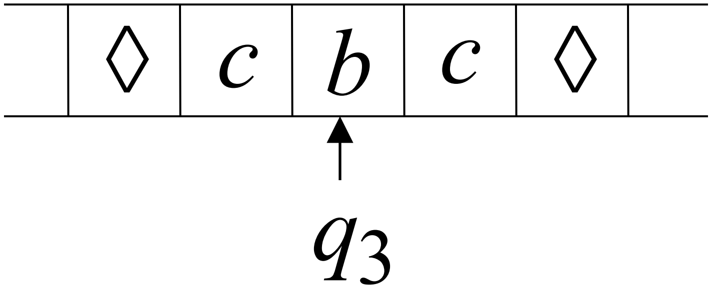


Time 1

Choice 1



Choice 2



Input string w is accepted if there is a computation:

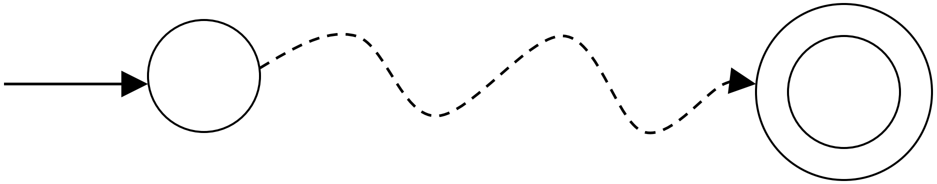
$$q_0 w \overset{*}{\succ} x \, q_f \, y$$

Initial configuration

Any accept state

Final Configuration

There is a computation:



Theorem: Nondeterministic machines have the same power with Standard Turing machines

Proof:

1. Nondeterministic machines simulate Standard Turing machines

2. Standard Turing machines simulate Nondeterministic machines

1. Nondeterministic Machines simulate Standard (deterministic) Turing Machines

Trivial: every deterministic machine is also nondeterministic

2. Standard (deterministic) Turing machines simulate Nondeterministic machines:

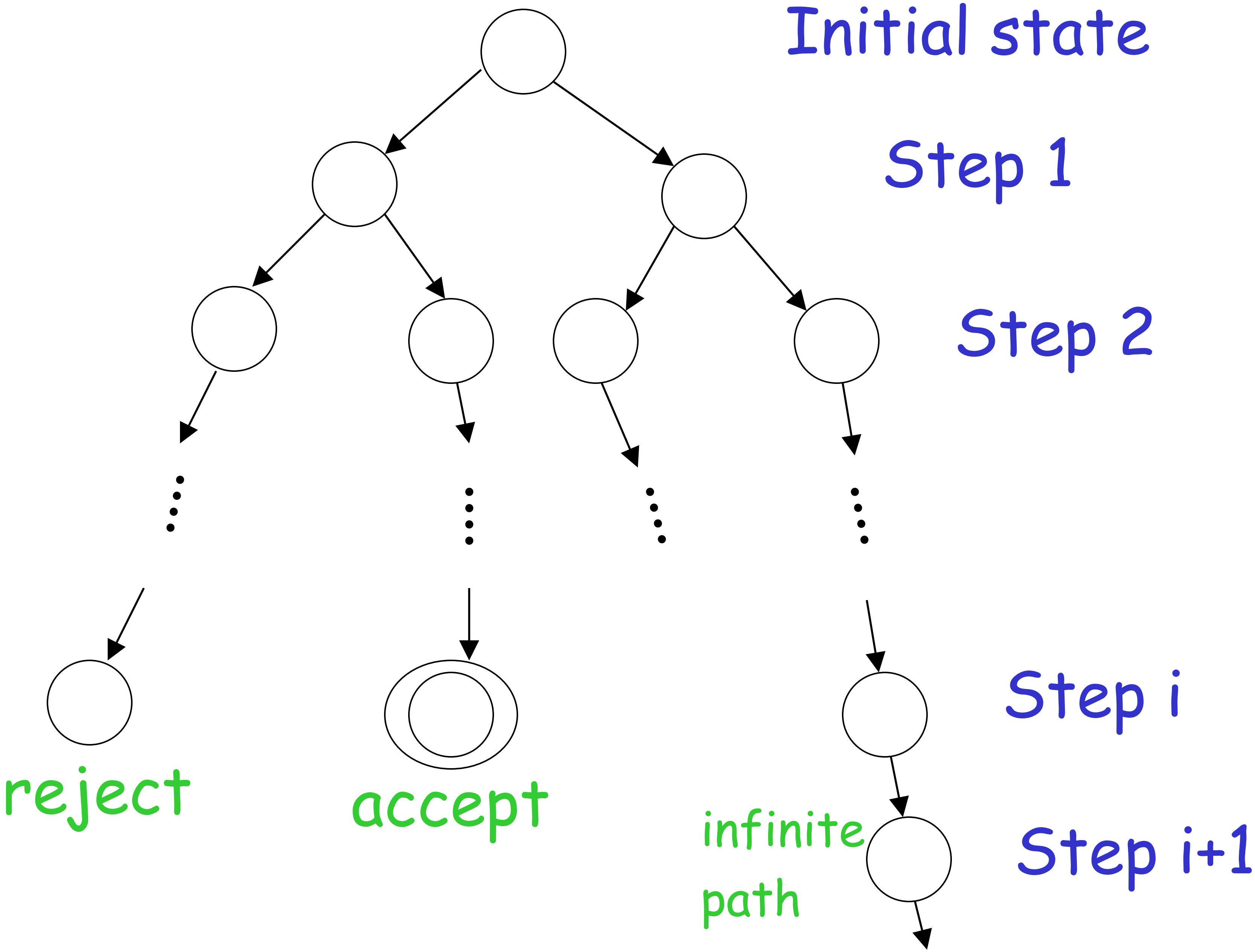
Deterministic machine:

- Uses a 2-dimensional tape

  (which is equivalent to 1-dimensional tape)

- Stores all possible computations of the non-deterministic machine on the 2-dimensional tape

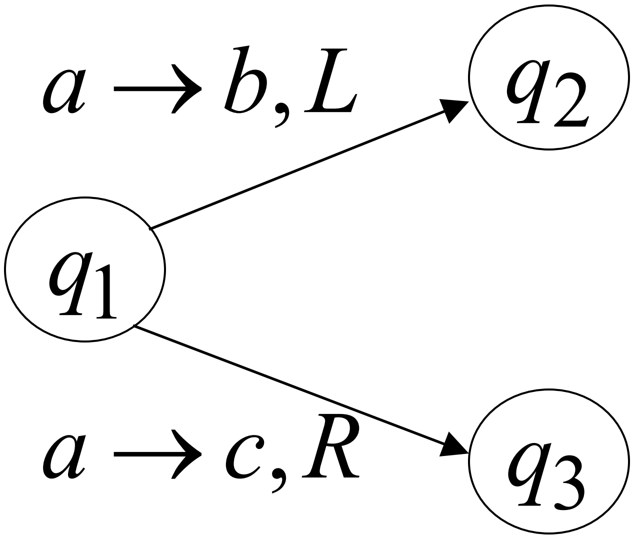
All possible computation paths



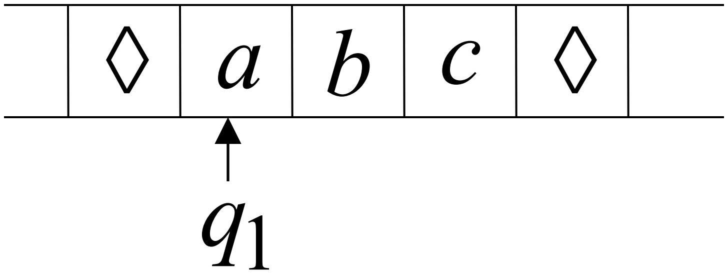
The Deterministic Turing machine simulates all possible computation paths:

- simultaneously
- step-by-step
- in a breadth-first search fashion

NonDeterministic machine



Time 0



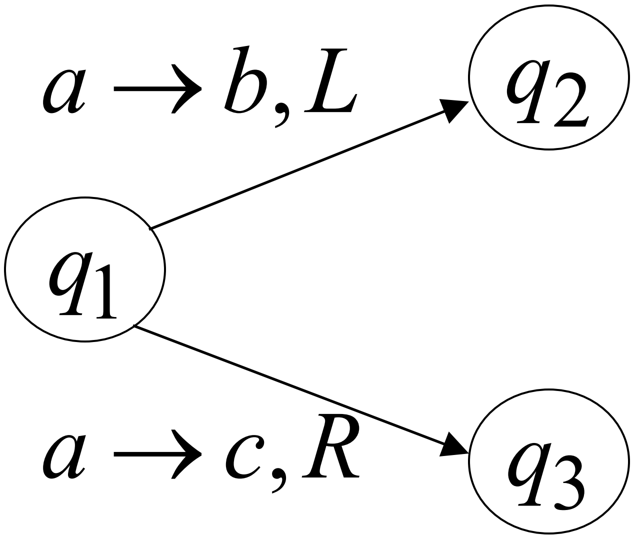
Deterministic machine

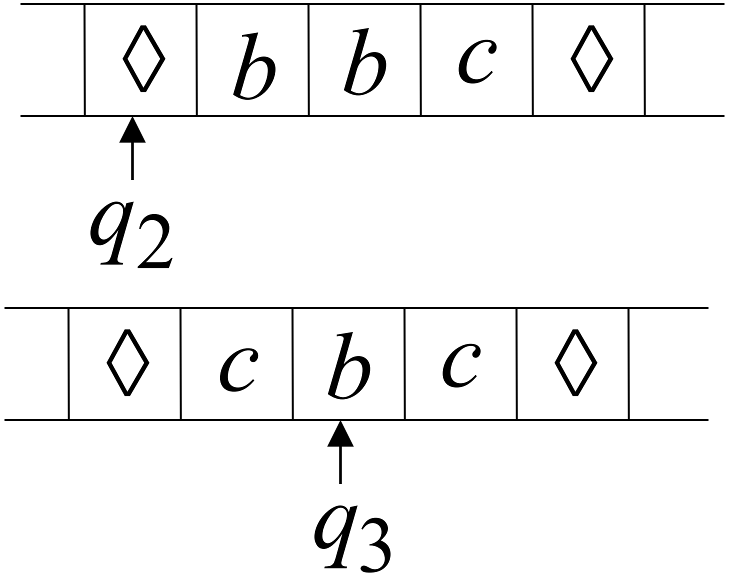
#	#	#	#	#	#
#	a	b	c	#	
#	q_1			#	
#	#	#	#	#	

current configuration

NonDeterministic machine

Time 1





Choice 1

Choice 2

Deterministic machine

	#	#	#	#	#	#	
#		b	b	c	#		
#	q_2				#		
#		c	b	c	#		
#			q_3		#		

Computation 1

Computation 2

Deterministic Turing machine

```
Repeat
    For each configuration in current step
    of non-deterministic machine,
    if there are two or more choices:
        1. Replicate configuration
        2. Change the state in the replicas
Until either the input string is accepted
    or rejected in all configurations
```

END OF PROOF

Remark:

The simulation takes in the worst case exponential time compared to the shortest accepting path length of the nondeterministic machine