

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

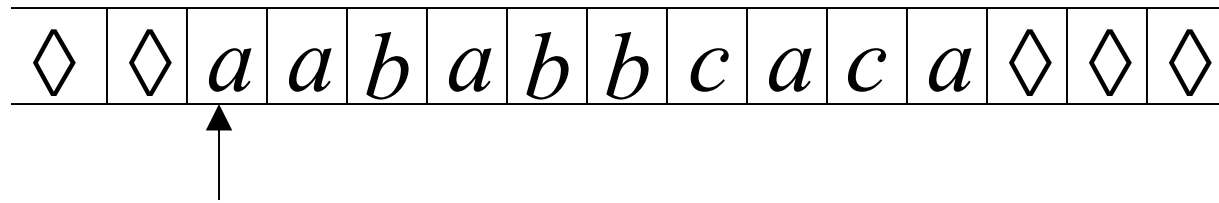
When we say: There exists an algorithm

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

Variations of the Turing Machine

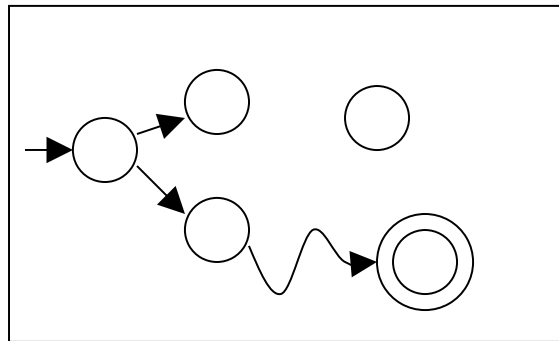
The Standard Model

Infinite Tape



Read-Write Head (Left or Right)

Control Unit



Deterministic

Variations of the Standard Model

- Turing machines with:
- Stay-Option
 - Semi-Infinite Tape
 - 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

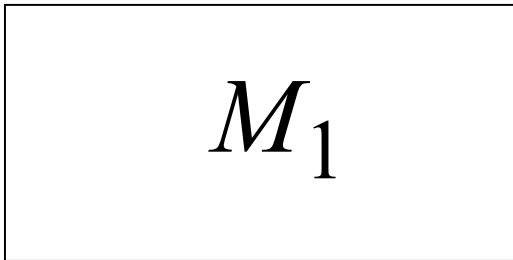
there is a machine M_2 of second class

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

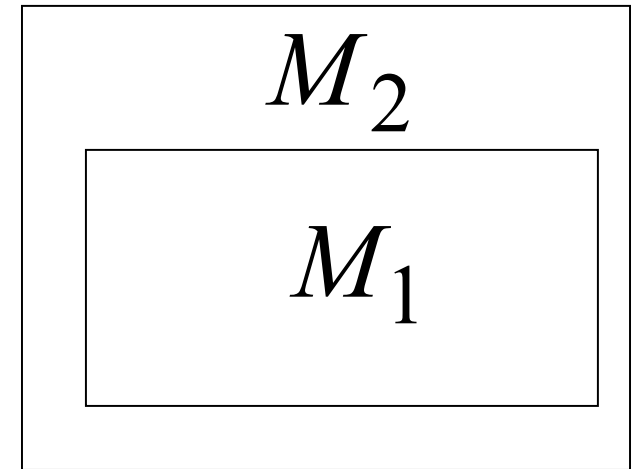
and vice-versa

Simulation: A technique to prove same power.
Simulate the machine of one class
with a machine of the other class

First Class
Original Machine

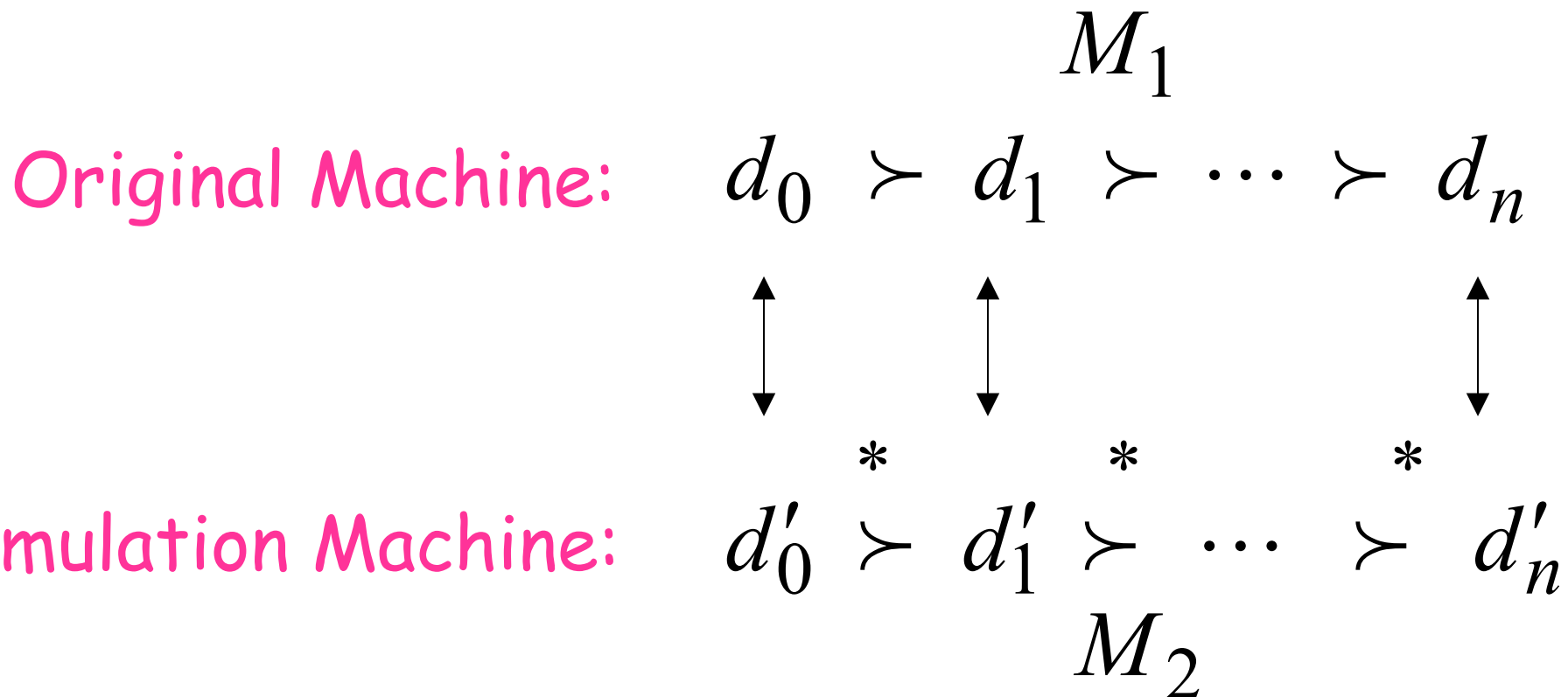


Second Class
Simulation Machine



simulates M_1

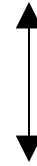
Configurations in the Original Machine M_1
 have corresponding configurations
 in the Simulation Machine M_2



Accepting Configuration

Original Machine:

d_f



Simulation Machine:

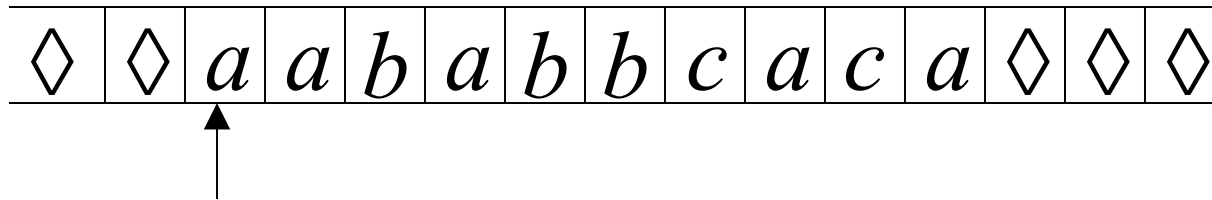
d'_f

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

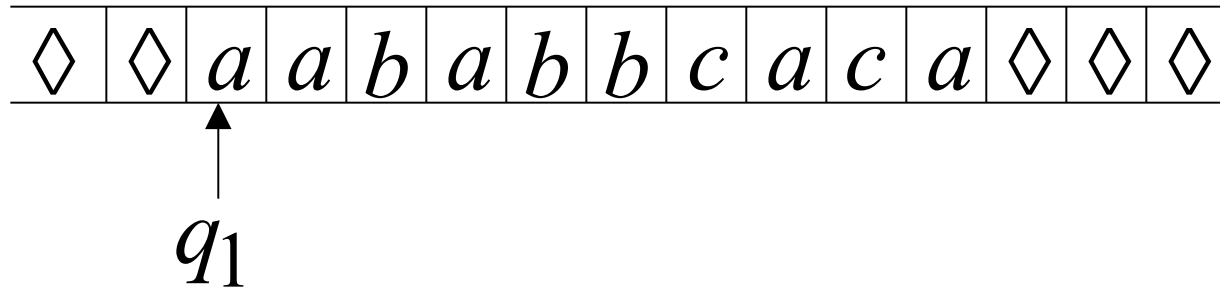


Left, Right, Stay

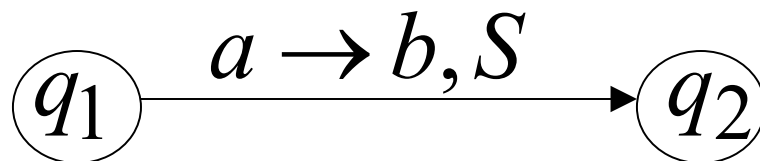
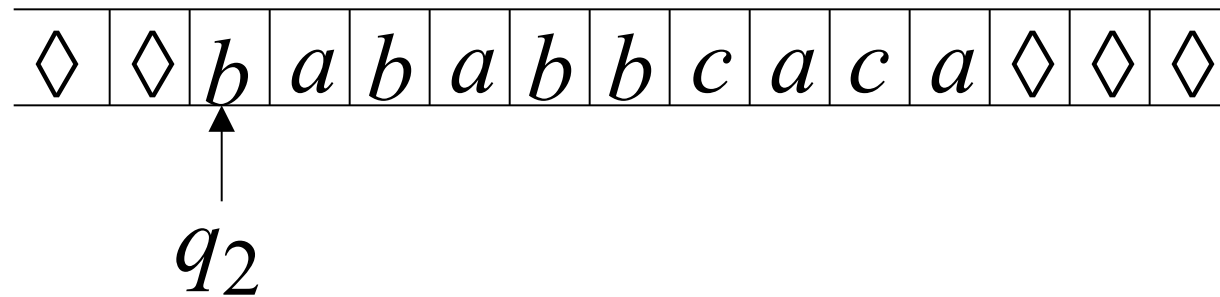
L,R,S: possible head moves

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

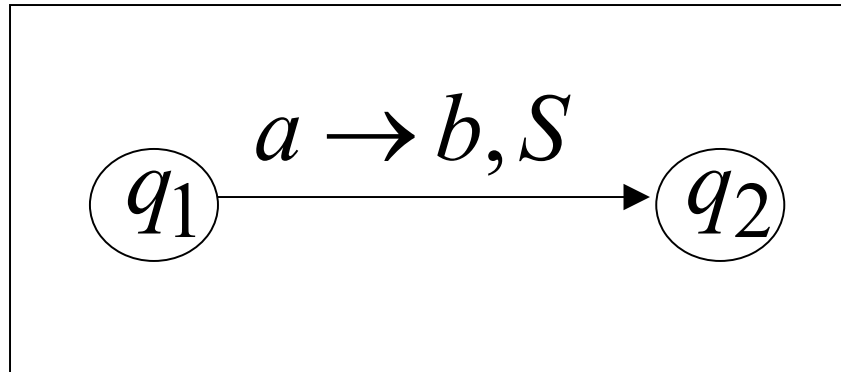
simulate Standard Turing machines

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

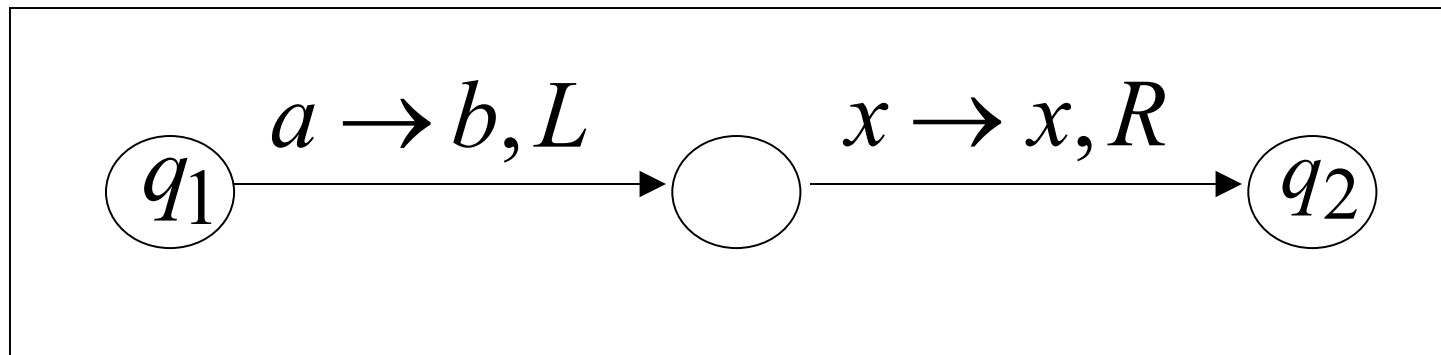
2. Standard Turing machines simulate Stay-Option machines

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

Stay-Option Machine



Simulation in Standard Machine

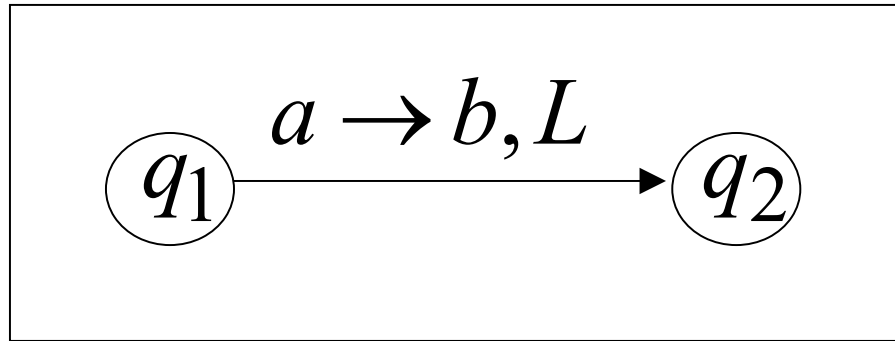


Busch's
simulation

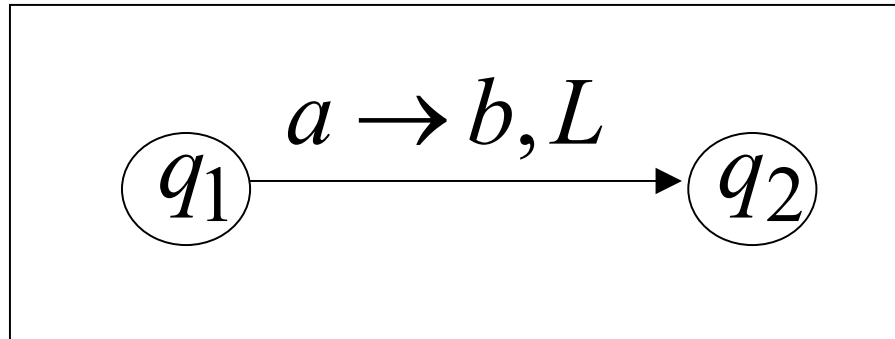
For every possible tape symbol x

For other transitions nothing changes

Stay-Option Machine



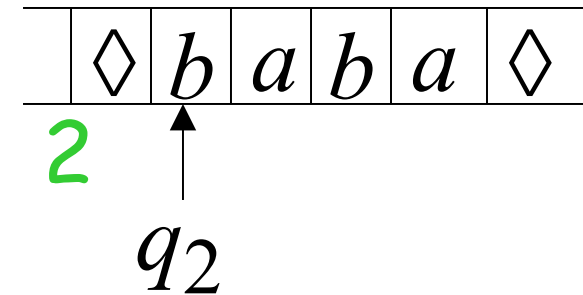
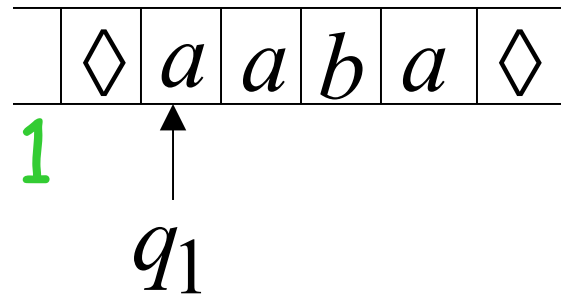
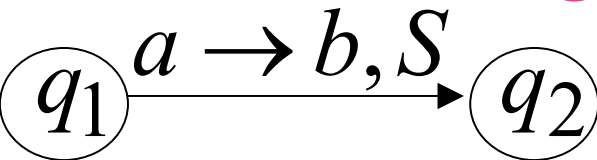
Simulation in Standard Machine



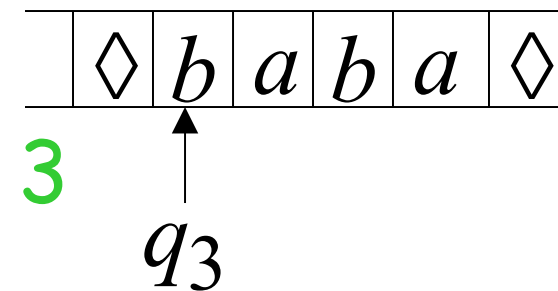
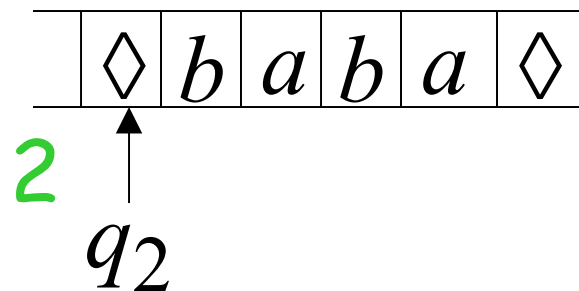
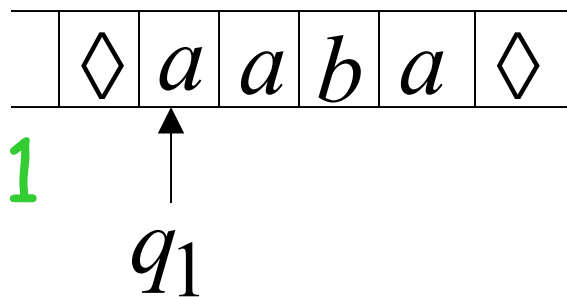
Similar for Right moves

example of simulation

Stay-Option Machine:



Simulation in Standard Machine:



END OF PROOF

A useful trick: Multiple Track Tape

helps for more complicated simulations

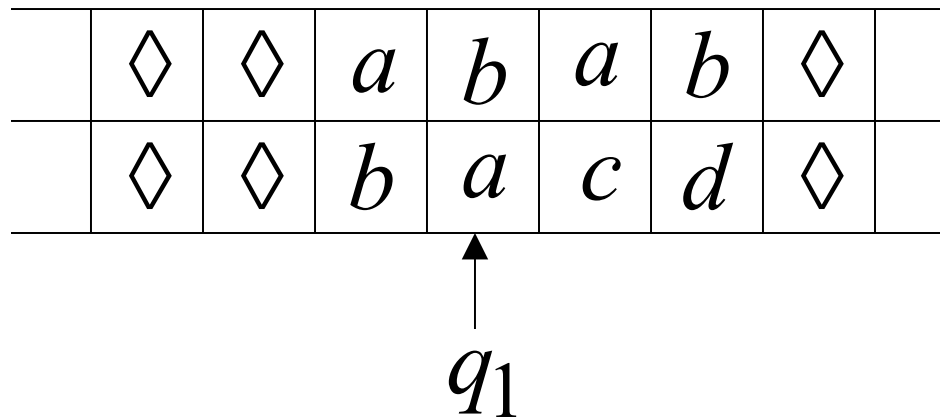
One Tape

	◇	◇	<i>a</i>	<i>b</i>	<i>a</i>	<i>b</i>	◇		track 1
	◇	◇	<i>b</i>	<i>a</i>	<i>c</i>	<i>d</i>	◇		track 2

One head

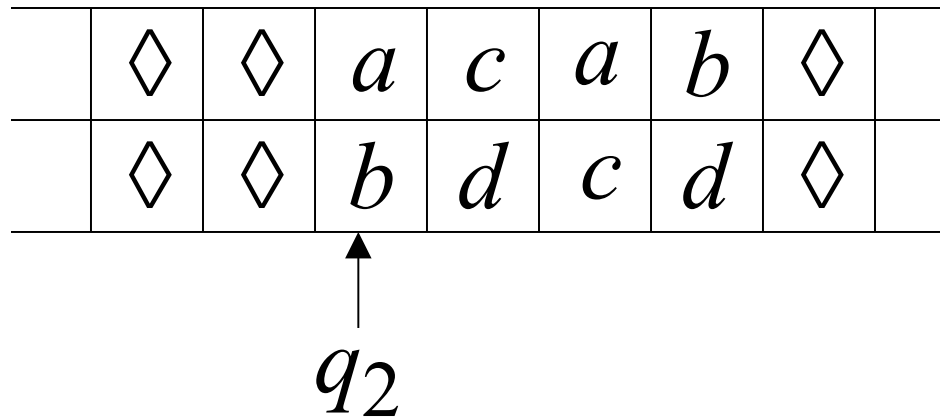
One symbol (*a*, *b*)

It is a standard Turing machine, but each tape alphabet symbol describes a pair of actual useful symbols



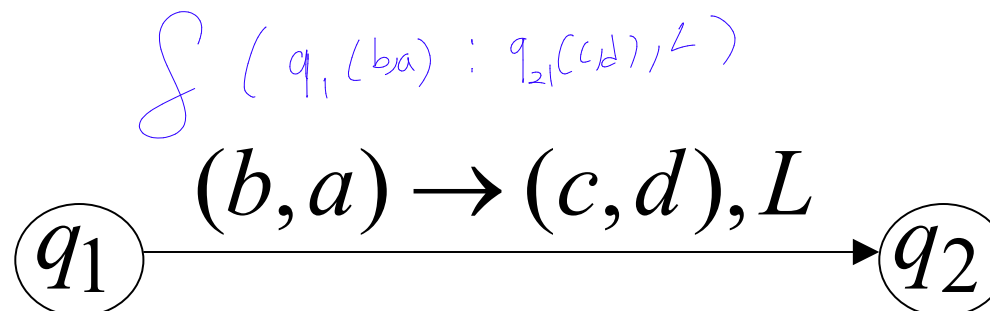
track 1

track 2



track 1

track 2

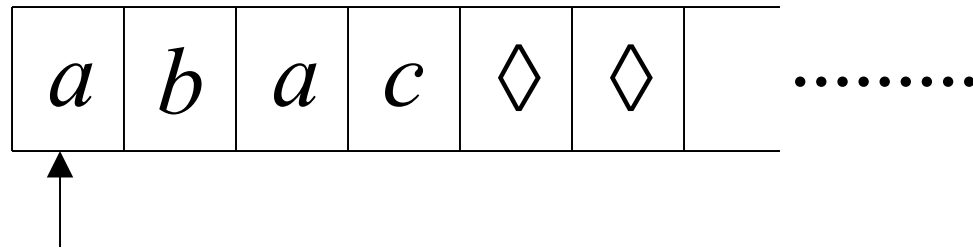


$$\Sigma = \{ (a,b), (c,d), (c,a), (b,d), \dots \}$$

Alfabeti: a, b, c, d

Semi-Infinite Tape

The head extends infinitely only to the right



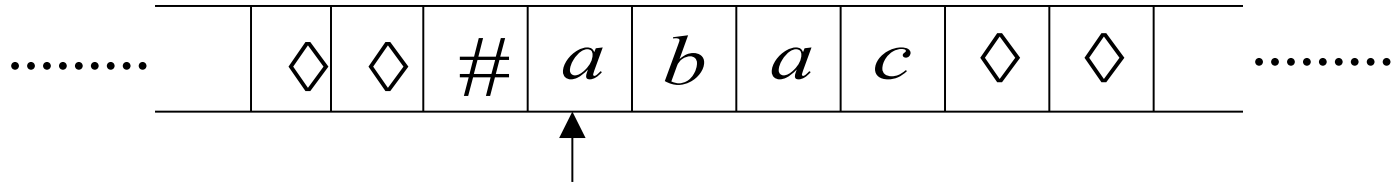
- Initial position is the leftmost cell
- When the head moves left from the border, it returns back to leftmost position

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:

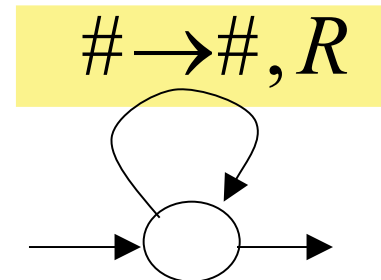


Standard Turing Machine

Semi-Infinite machine modifications

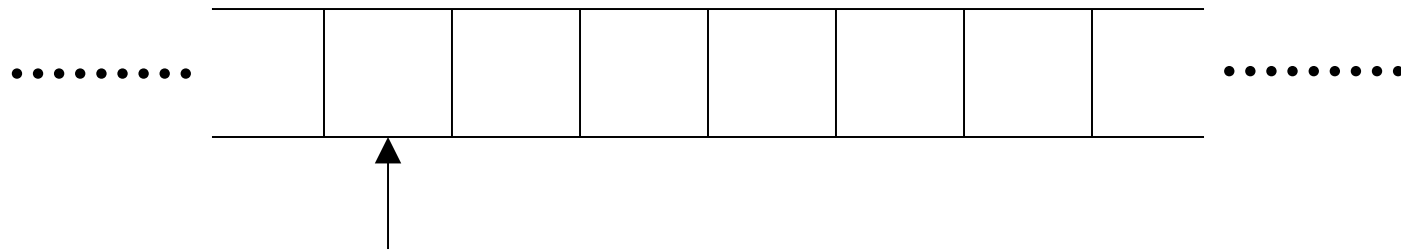
a. insert special symbol $\#$
at left of input string

b. Add a self-loop
to every state
(except states with no
outgoing transitions)

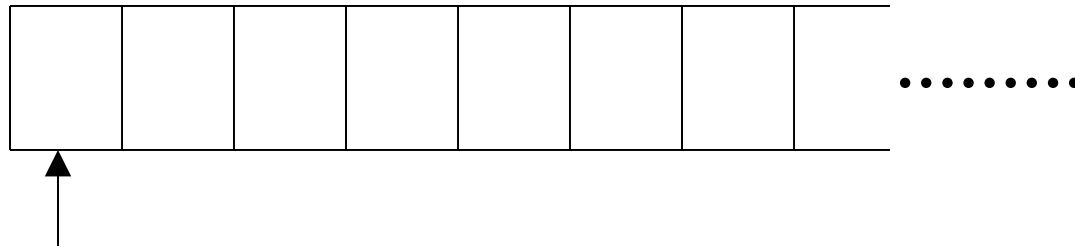


2. Semi-Infinite tape machines simulate Standard Turing machines:

Standard machine



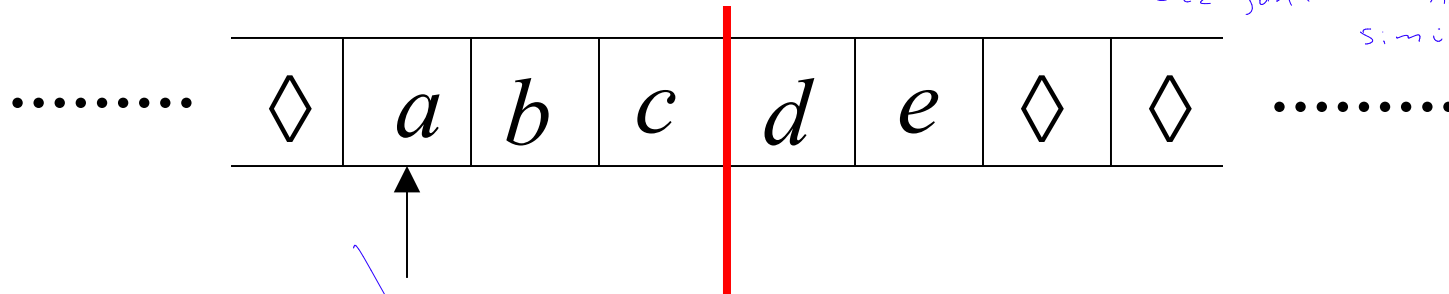
Semi-Infinite tape machine



Squeeze infinity of both directions
to one direction

Standard machine

→ Sensez makineyle
Etkün sonlu makine
simüle edilebilir



reference point

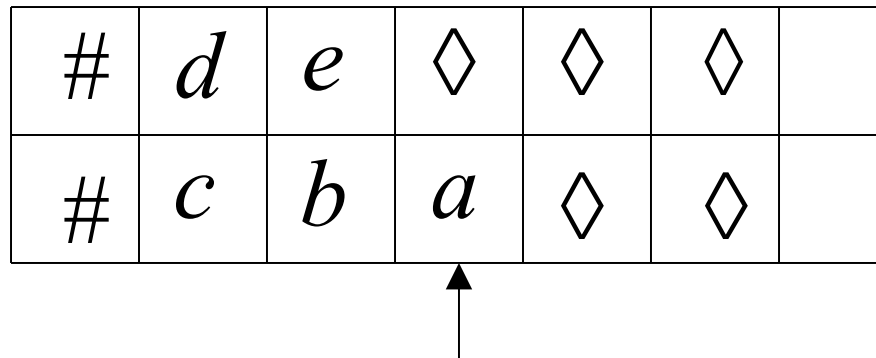
Bu yaklaşımla simüle edilebilir: 2

Bu yaklaşım

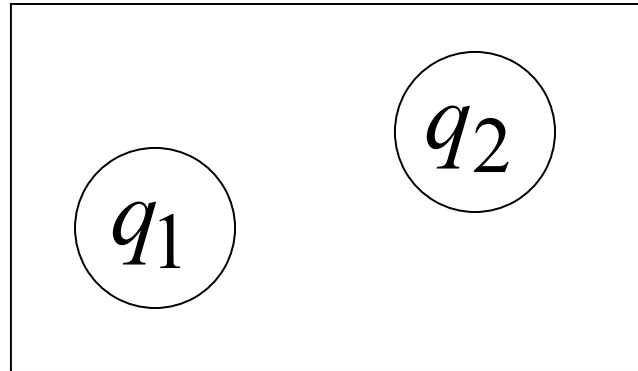
Semi-Infinite tape machine with two tracks

Right part

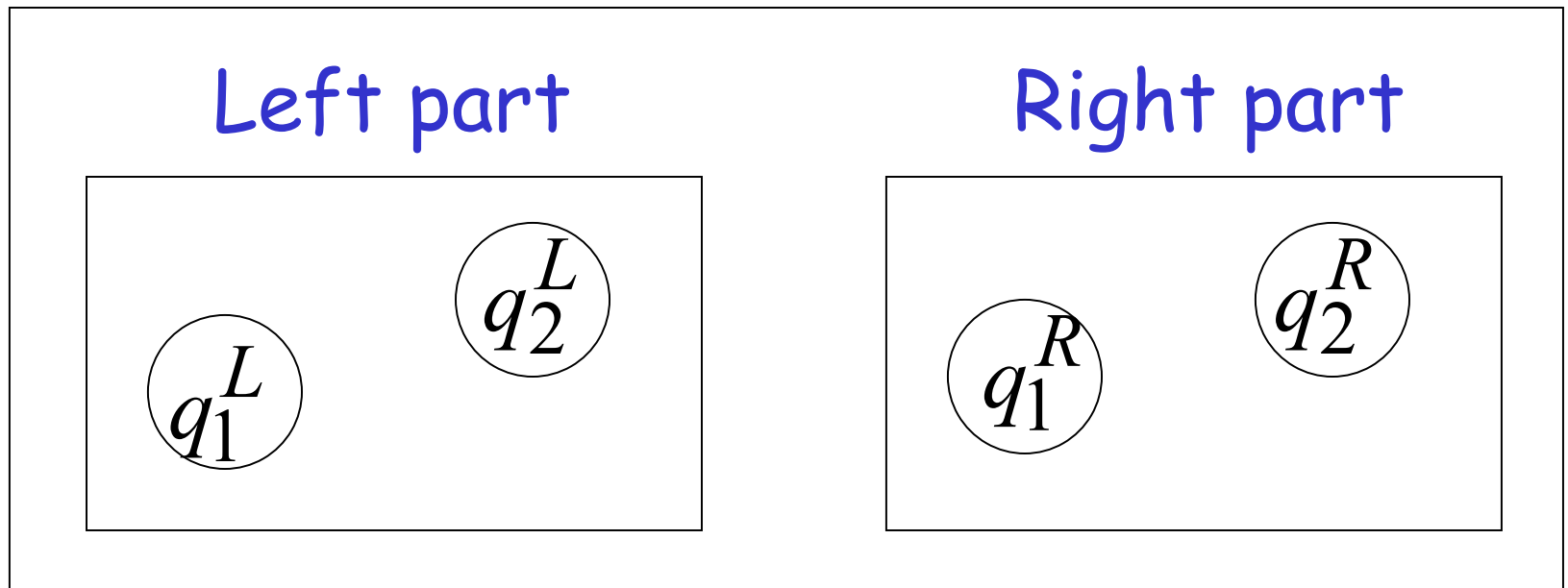
Left part



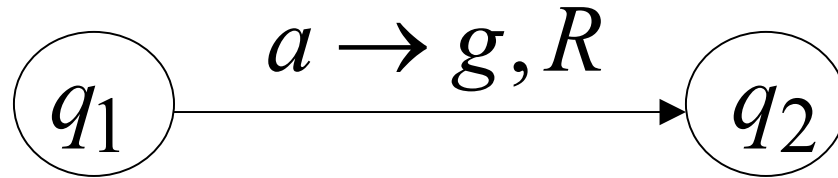
Standard machine



Semi-Infinite tape machine

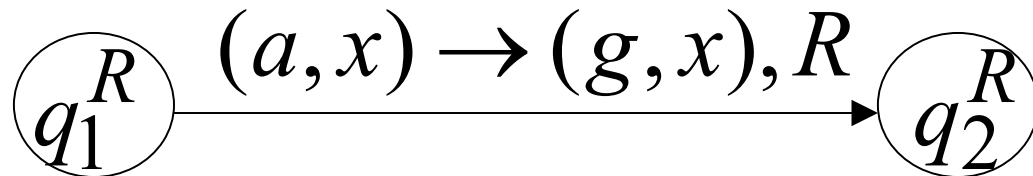


Standard machine

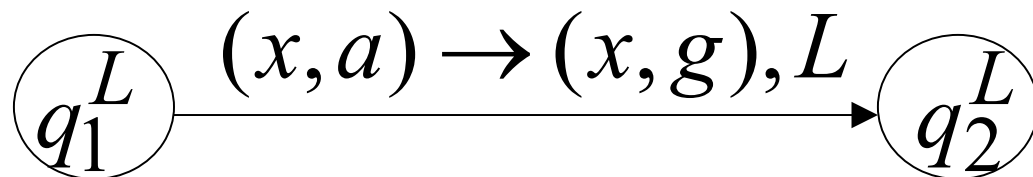


Semi-Infinite tape machine

Right part



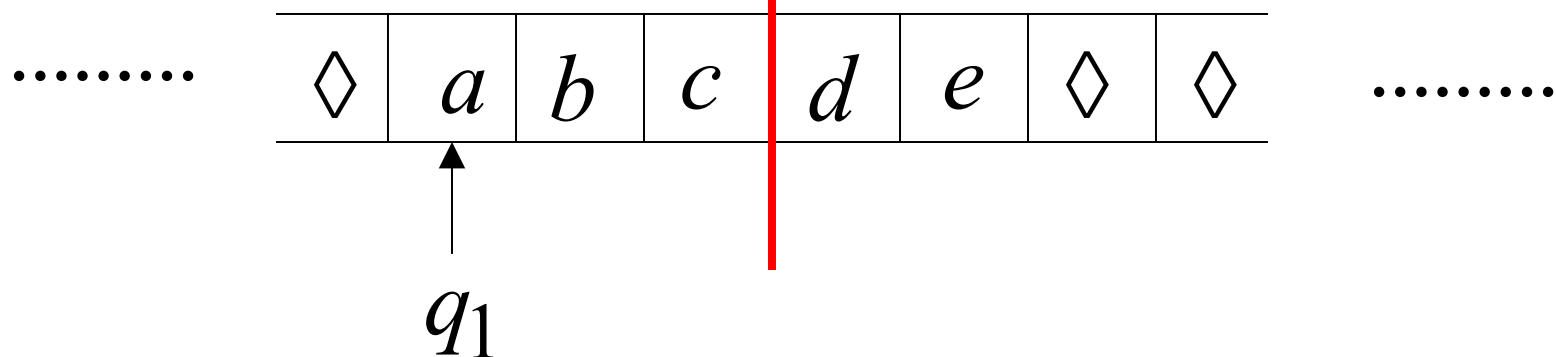
Left part



For all tape symbols x

Time 1

Standard machine



Semi-Infinite tape machine

Right part

#	d	e	\diamond	\diamond	\diamond	
---	-----	-----	------------	------------	------------	--

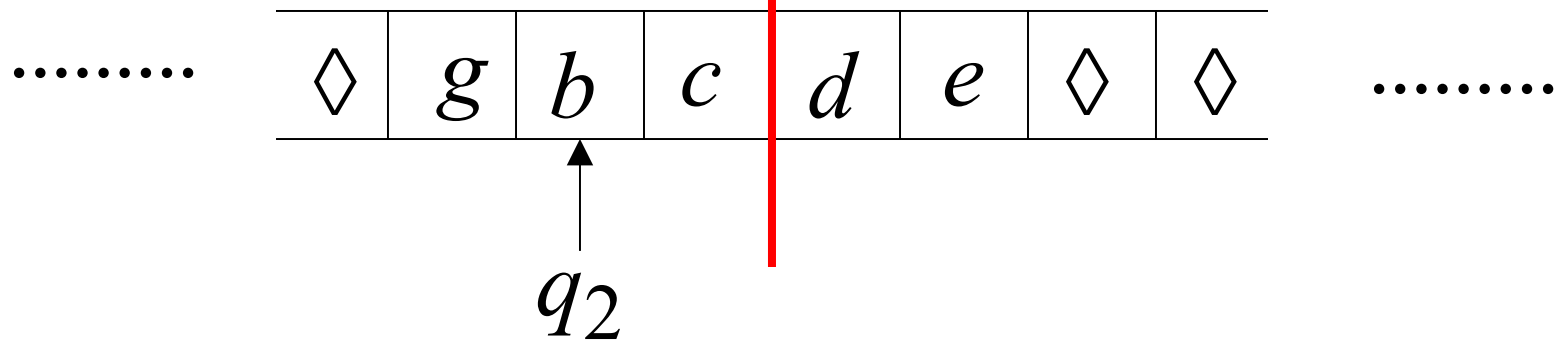
Left part

#	c	b	a	\diamond	\diamond	
---	-----	-----	-----	------------	------------	--

q_1^L

Time 2

Standard machine



Semi-Infinite tape machine

Right part

#	<i>d</i>	<i>e</i>	◇	◇	◇	
---	----------	----------	---	---	---	--

.....

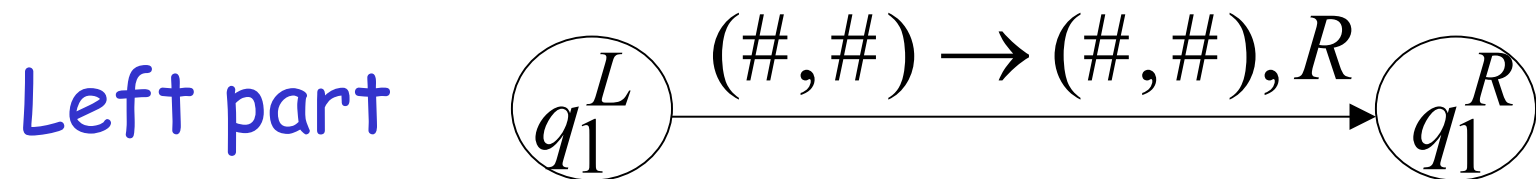
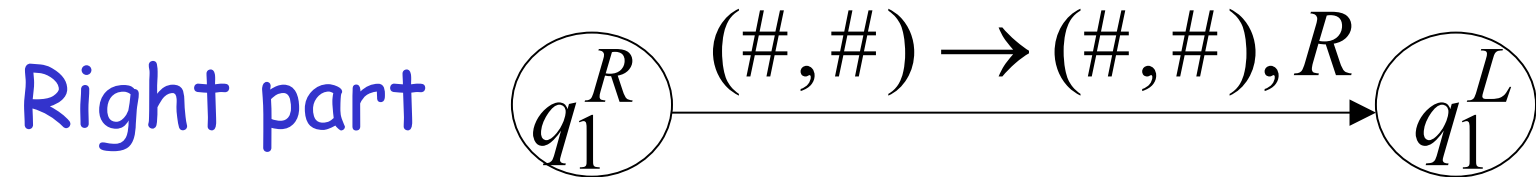
Left part

#	<i>c</i>	<i>b</i>	<i>g</i>	◇	◇	
---	----------	----------	----------	---	---	--

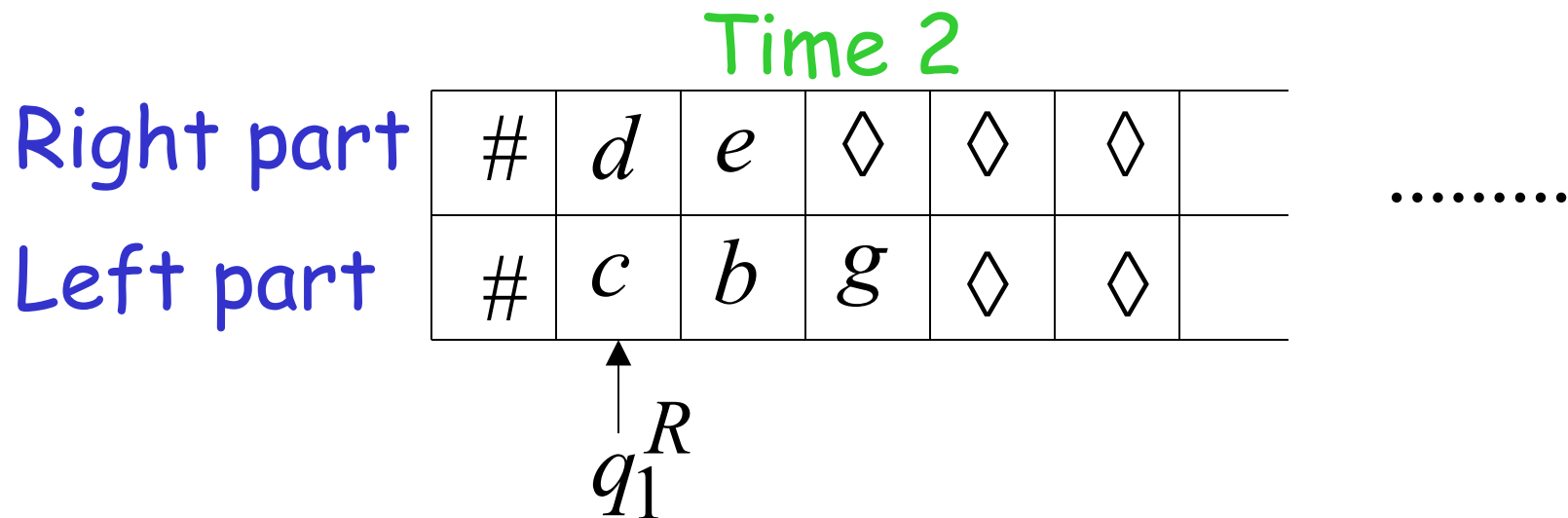
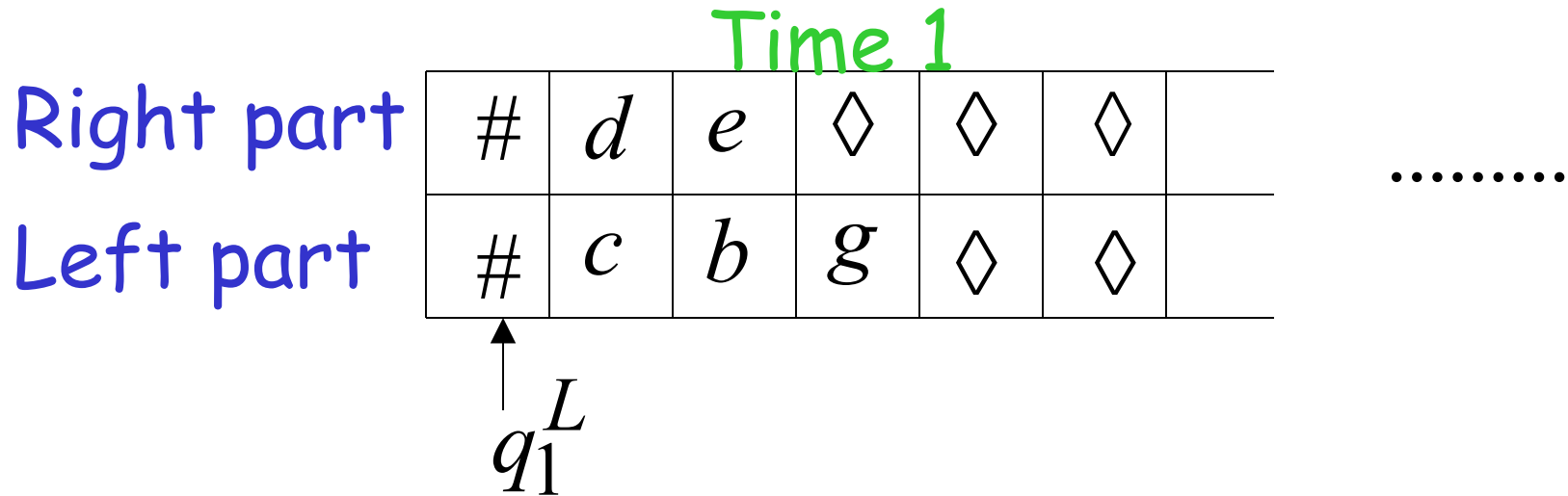
q_2^L

At the border:

Semi-Infinite tape machine

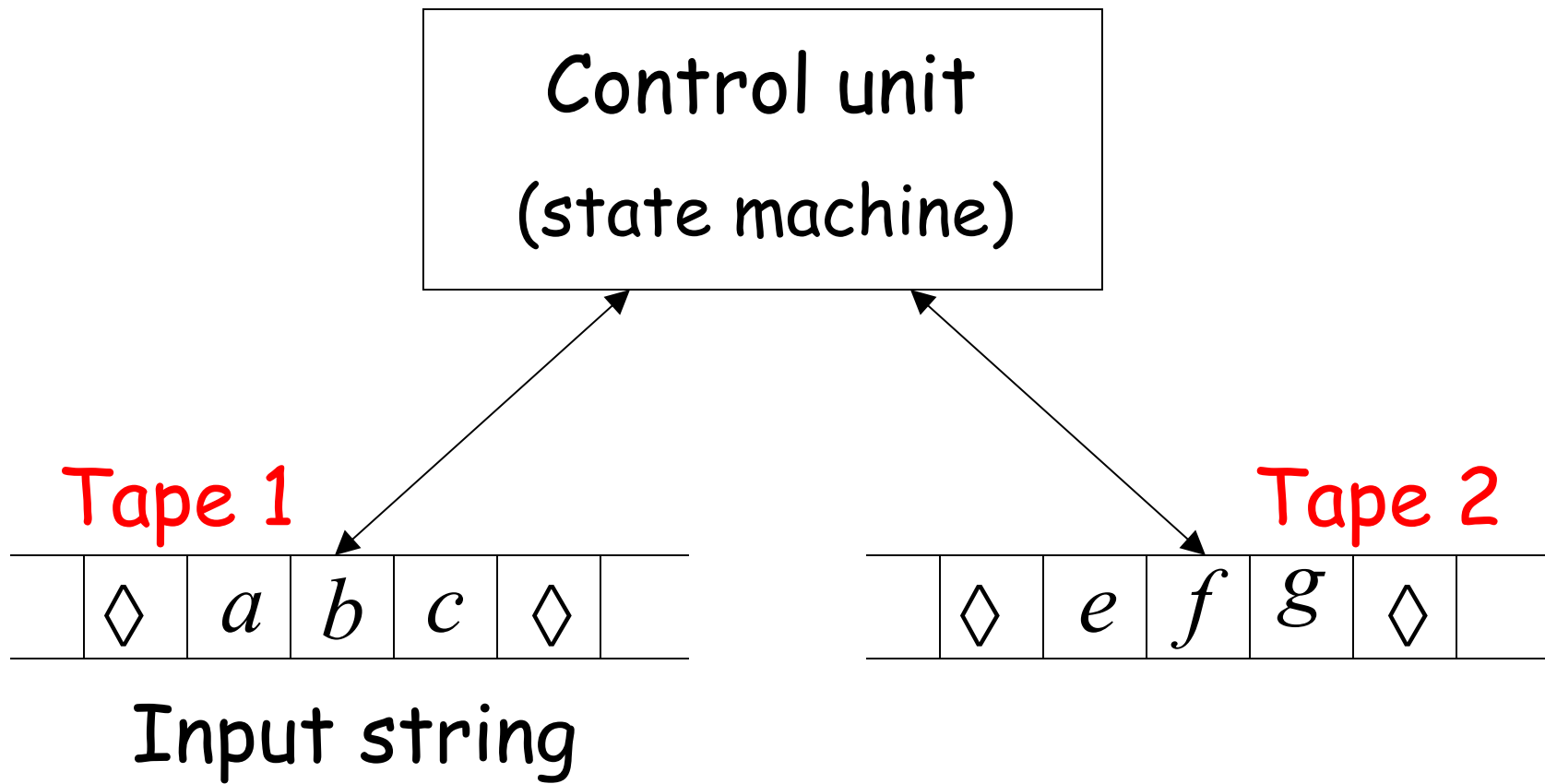


Semi-Infinite tape machine

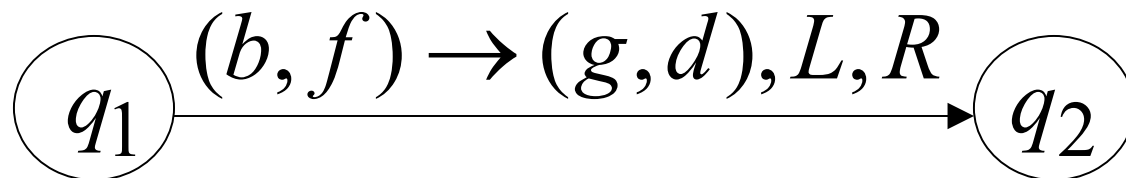
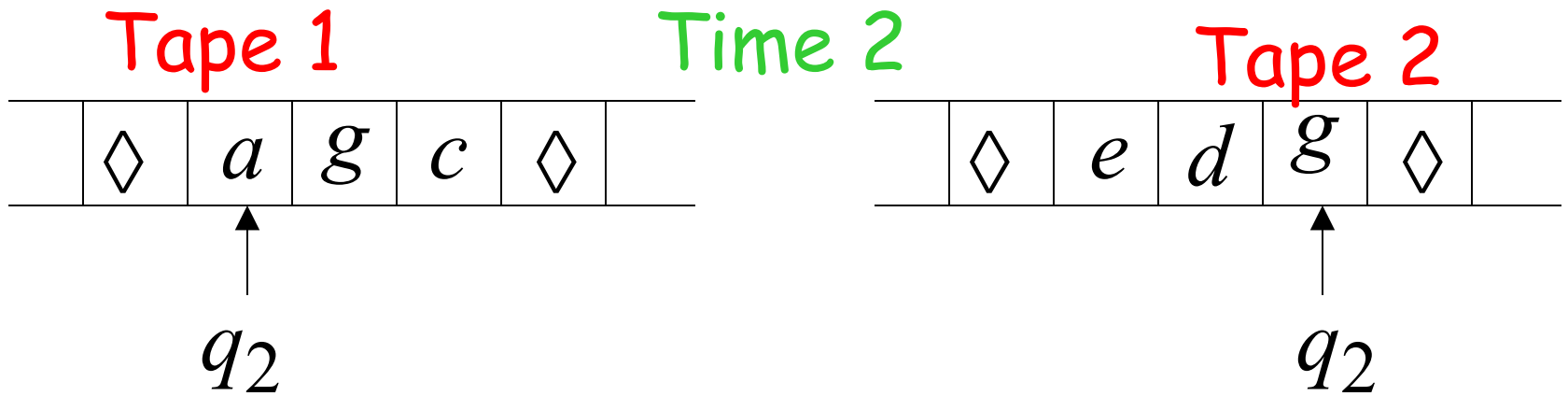
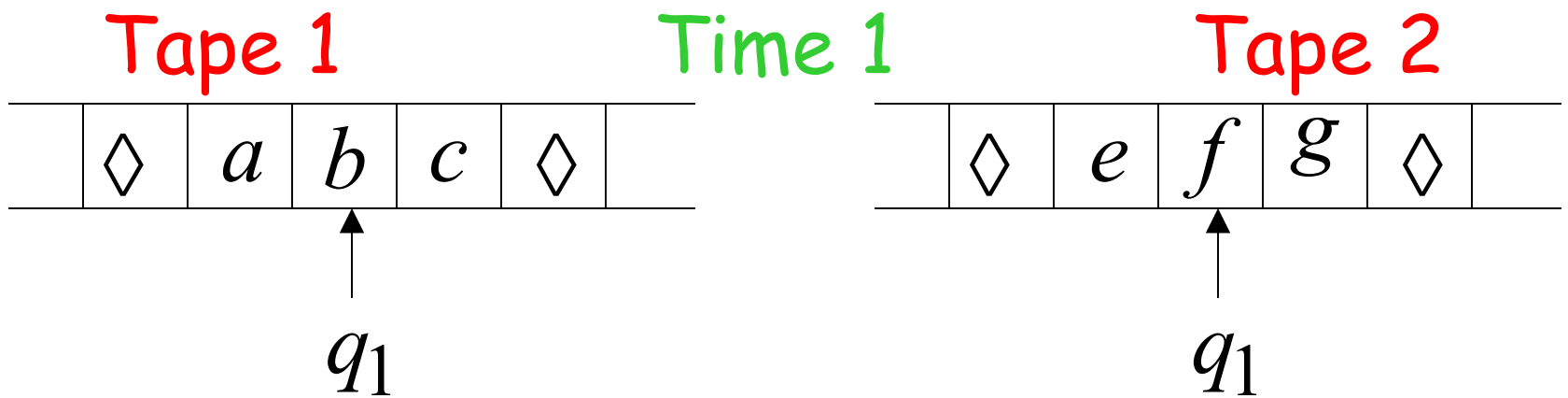


END OF PROOF

Multi-tape Turing Machines



Input string appears on Tape 1



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 one tape

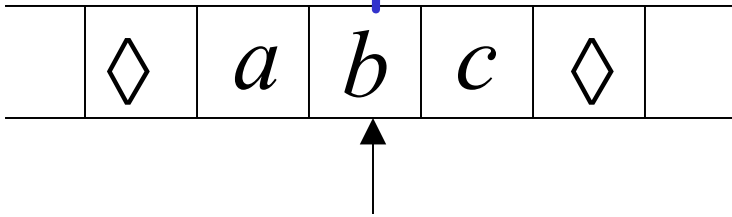
2. Standard Turing machines simulate Multi-tape machines:

Standard machine:

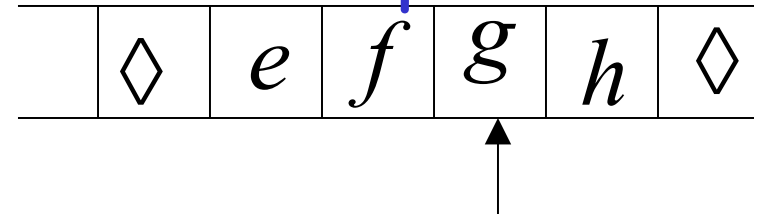
- Uses a multi-track tape to simulate the multiple tapes
- A tape of the Multi-tape machine corresponds to a pair of tracks

Multi-tape Machine

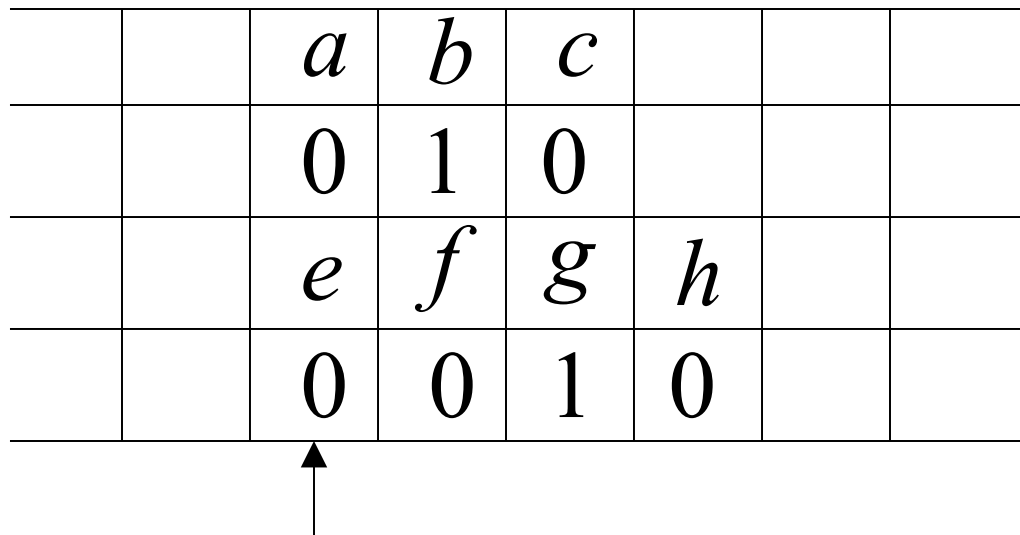
Tape 1



Tape 2



Standard machine with four track tape



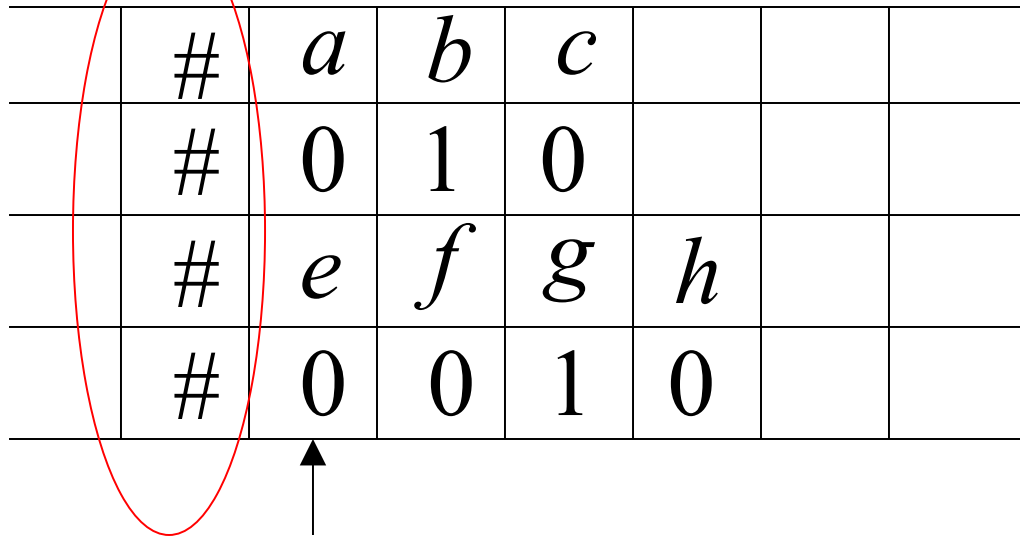
Tape 1

head position

Tape 2

head position

Reference point



The diagram shows a 4x8 grid representing a multi-tape Turing machine tape. The first column contains the '#' symbol in all four rows. The second column contains 'a', '0', 'e', and '0'. The third column contains 'b', '1', 'f', and '0'. The fourth column contains 'c', '0', 'g', and '1'. The fifth column contains '0', 'h', and '0'. The remaining three columns are empty. A red oval encircles the first column, and a black arrow points to the '0' in the second row, second column. This '0' is the reference point.

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

Tape 1

head position

Tape 2

head position

Repeat for each Multi-tape state transition:

1. Return to reference point
1. dan geyzer, update sonra digeri update edip agni adimlere ugguluyor.
2. Find current symbol in Track 1 and update
3. Return to reference point
4. Find current symbol in Tape 2 and update

END OF PROOF

Same power doesn't imply same speed:

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

↓ unrecognizable

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

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

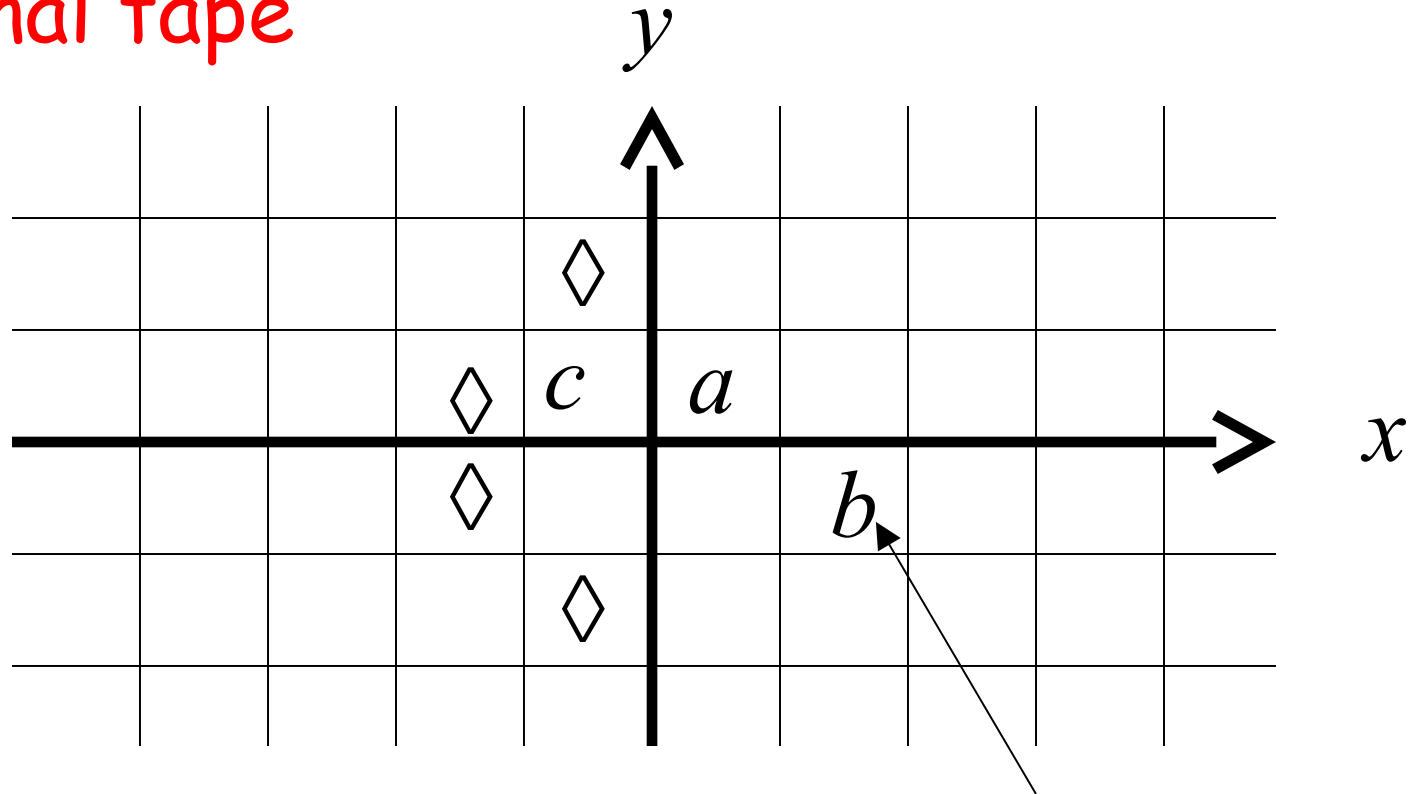
2-tape machine: $O(n)$ time

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

2. Compare a^n on tape 1
and b^n on 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

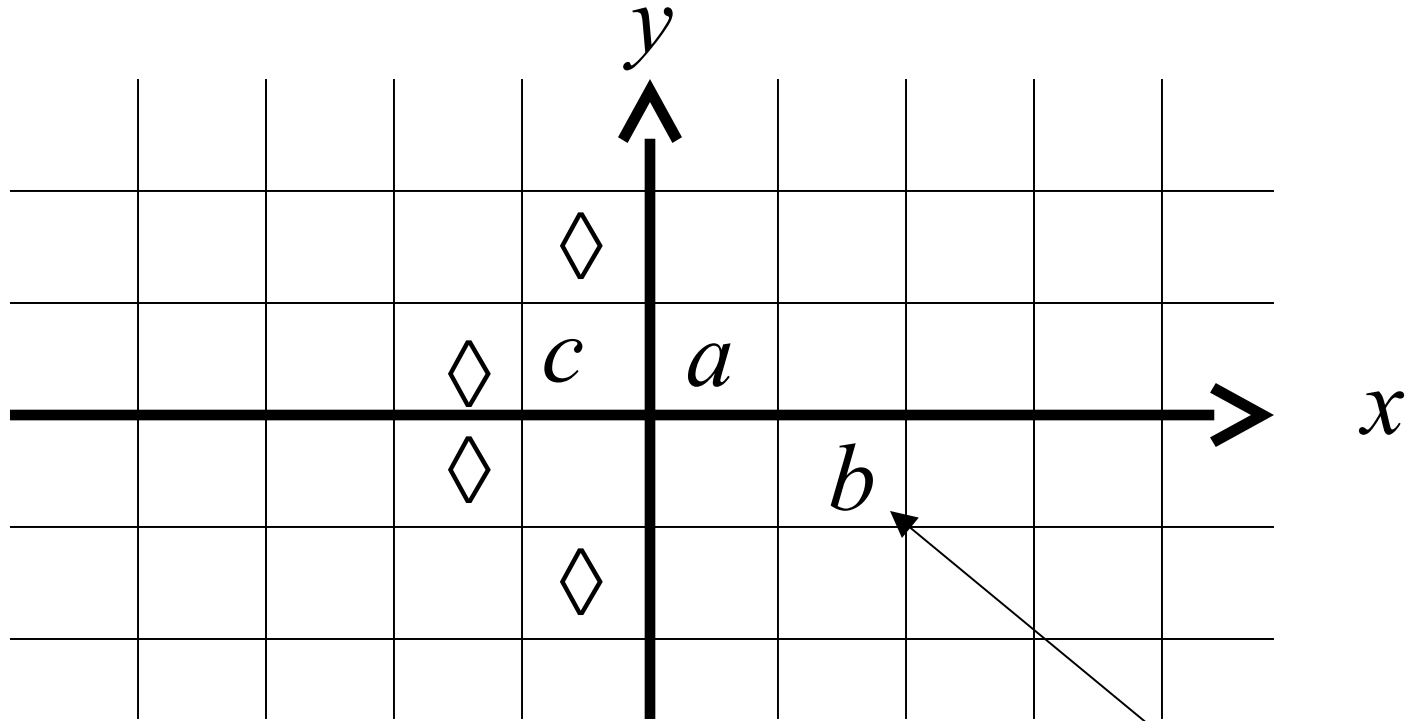
Trivial: Use one dimension

2. Standard Turing machines simulate Multidimensional machines

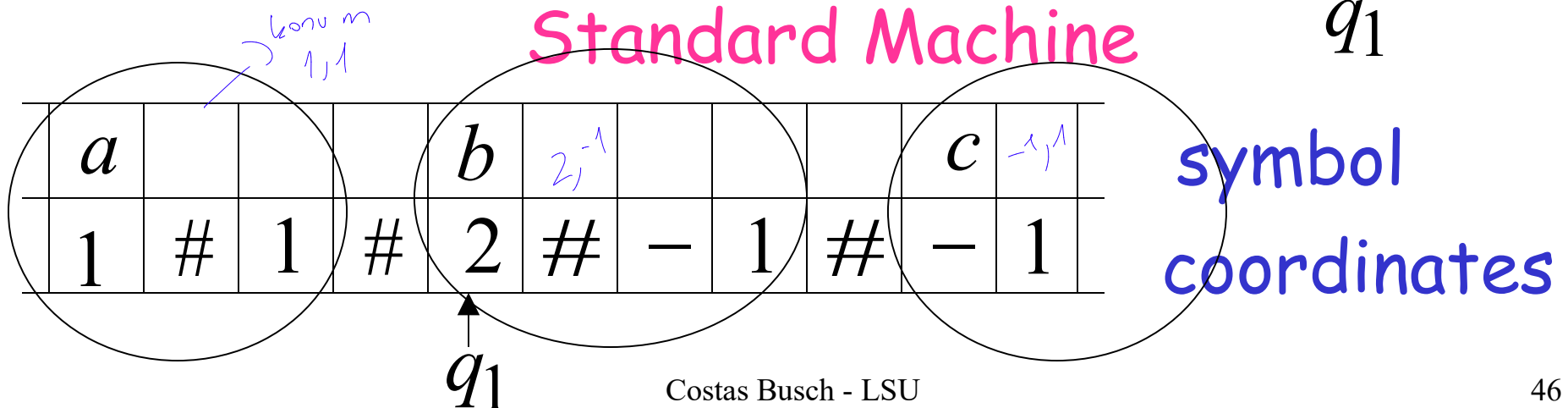
Standard machine:

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

2-dimensional machine



Standard Machine



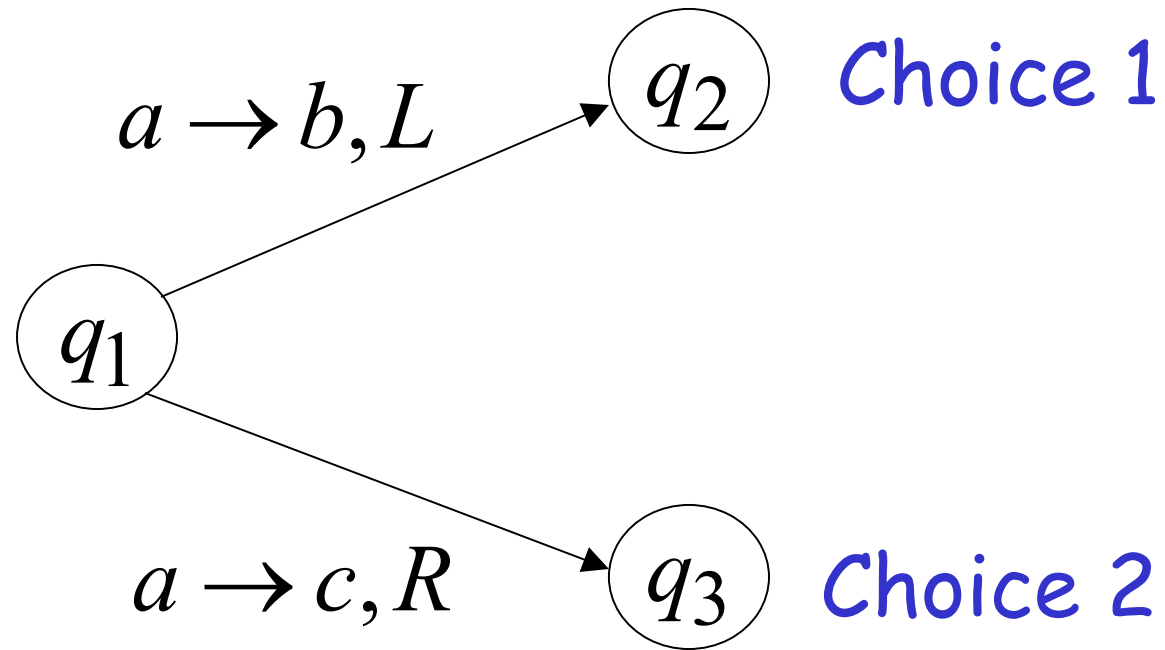
Standard machine:

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

1. Update current symbol
2. Compute coordinates of next position
3. Find next position on tape

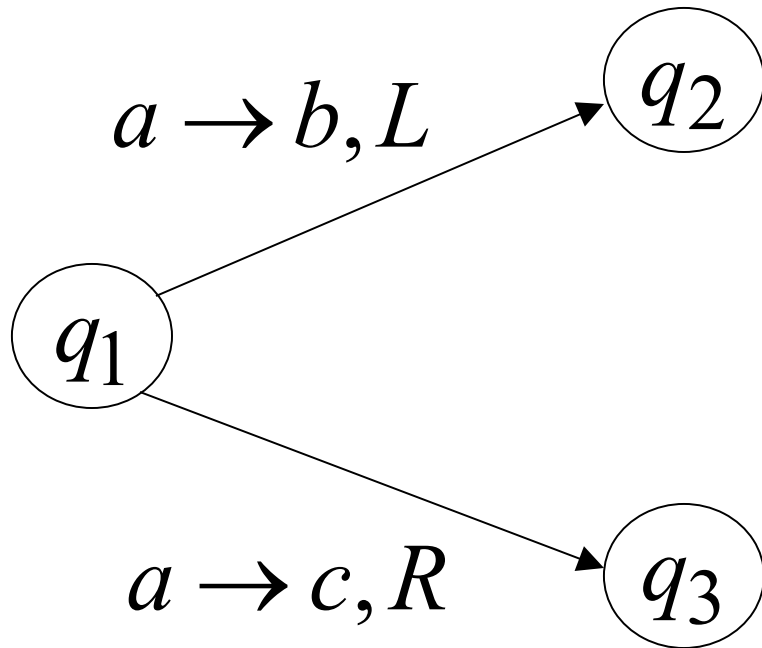
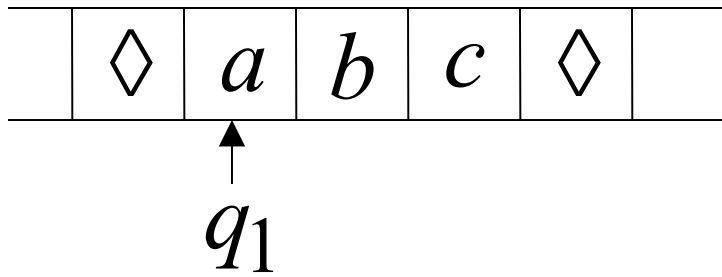
END OF PROOF

Nondeterministic Turing Machines



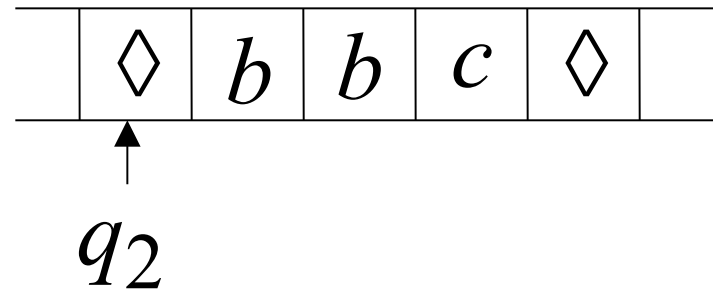
Allows Non Deterministic Choices

Time 0

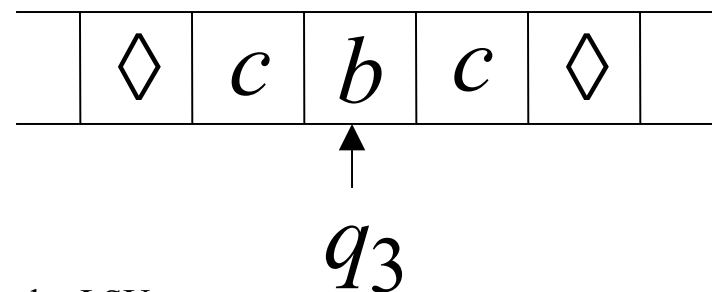


Time 1

Choice 1



Choice 2



Input string w is accepted if
there is a computation:

*

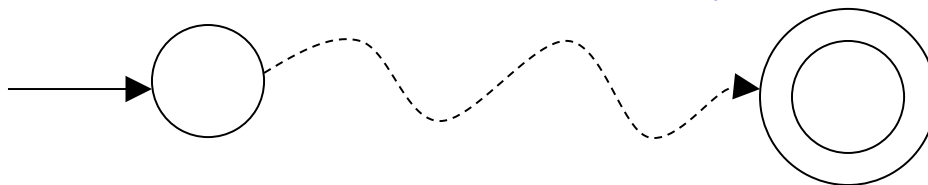
$$q_0 w \succ x q_f y$$

Initial configuration

Final Configuration

Any accept state

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

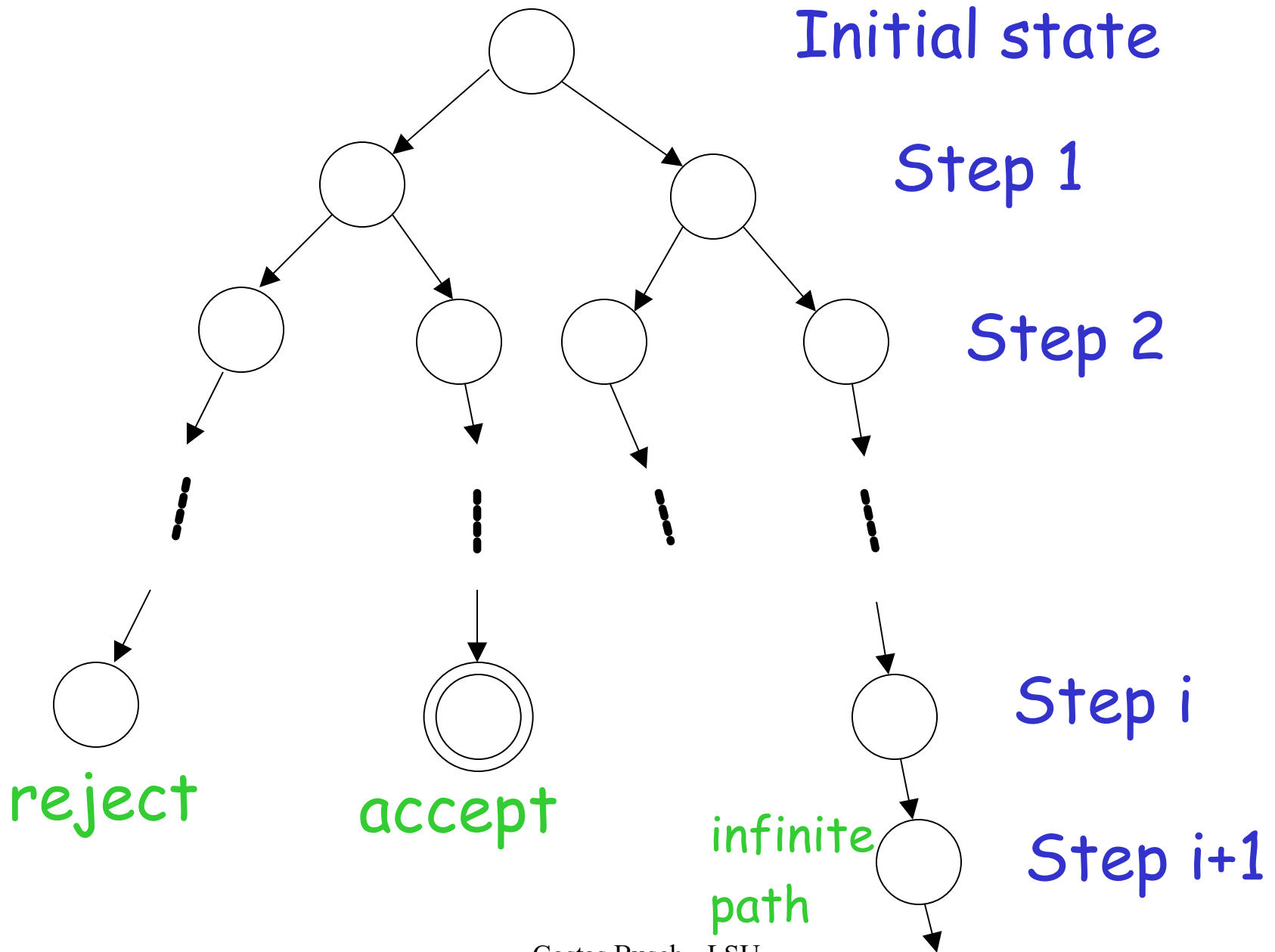
Trivial: every deterministic machine
is also nondeterministic

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

Deterministic machine:

- Uses a 2-dimensional tape
(equivalent to standard Turing machine with one tape)
- Stores all possible computations of the non-deterministic machine on the 2-dimensional tape

All possible computation paths

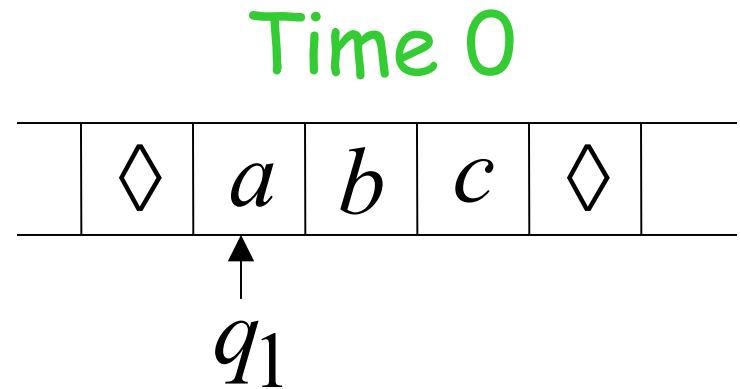
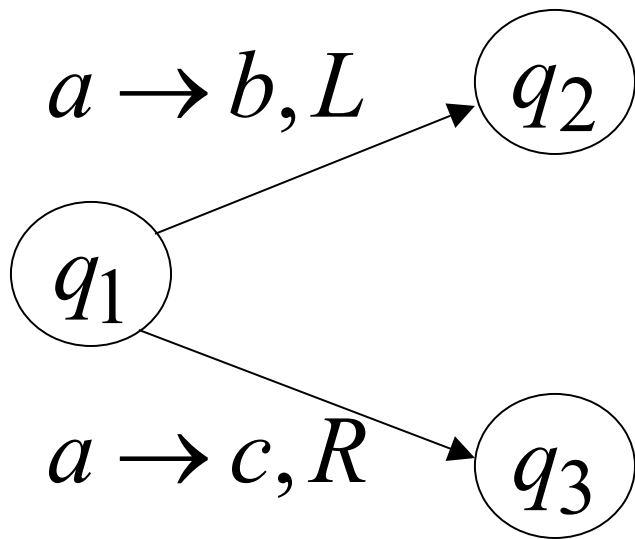


The Deterministic Turing machine
simulates all possible computation paths:

- simultaneously
- step-by-step
- with **breadth-first** search

depth-first may result getting stuck at exploring
an infinite path before discovering the accepting path

NonDeterministic machine



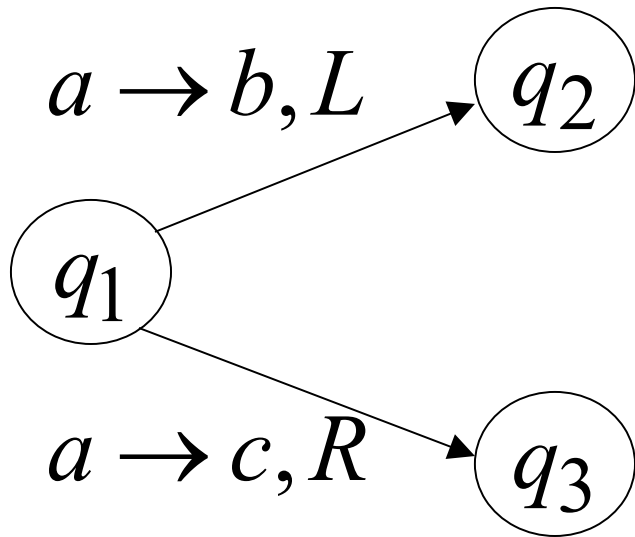
Deterministic machine

	#	#	#	#	#	#	
	#	<i>a</i>	<i>b</i>	<i>c</i>	#		
	#	<i>q</i> ₁			#		
	#	#	#	#	#		

current
configuration

NonDeterministic machine

Time 1



	◇	<i>b</i>	<i>b</i>	<i>c</i>	◇	
--	---	----------	----------	----------	---	--

Choice 1

q_2

	◇	<i>c</i>	<i>b</i>	<i>c</i>	◇	
--	---	----------	----------	----------	---	--

Choice 2

q_3

Deterministic machine

	#	#	#	#	#	#	
#		<i>b</i>	<i>b</i>	<i>c</i>	#		
#	q_2				#		
#		<i>c</i>	<i>b</i>	<i>c</i>	#		
#			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

If the non-deterministic machine accepts the input string:

The deterministic machine accepts and halts too

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

If the non-deterministic machine does not accept the input string:

1. The simulation halts if all paths reach a halting state

OR

2. The simulation never terminates if there is a never-ending path (infinite loop)

In either case the deterministic machine rejects too (1. by halting or 2. by simulating the infinite loop)

END OF PROOF