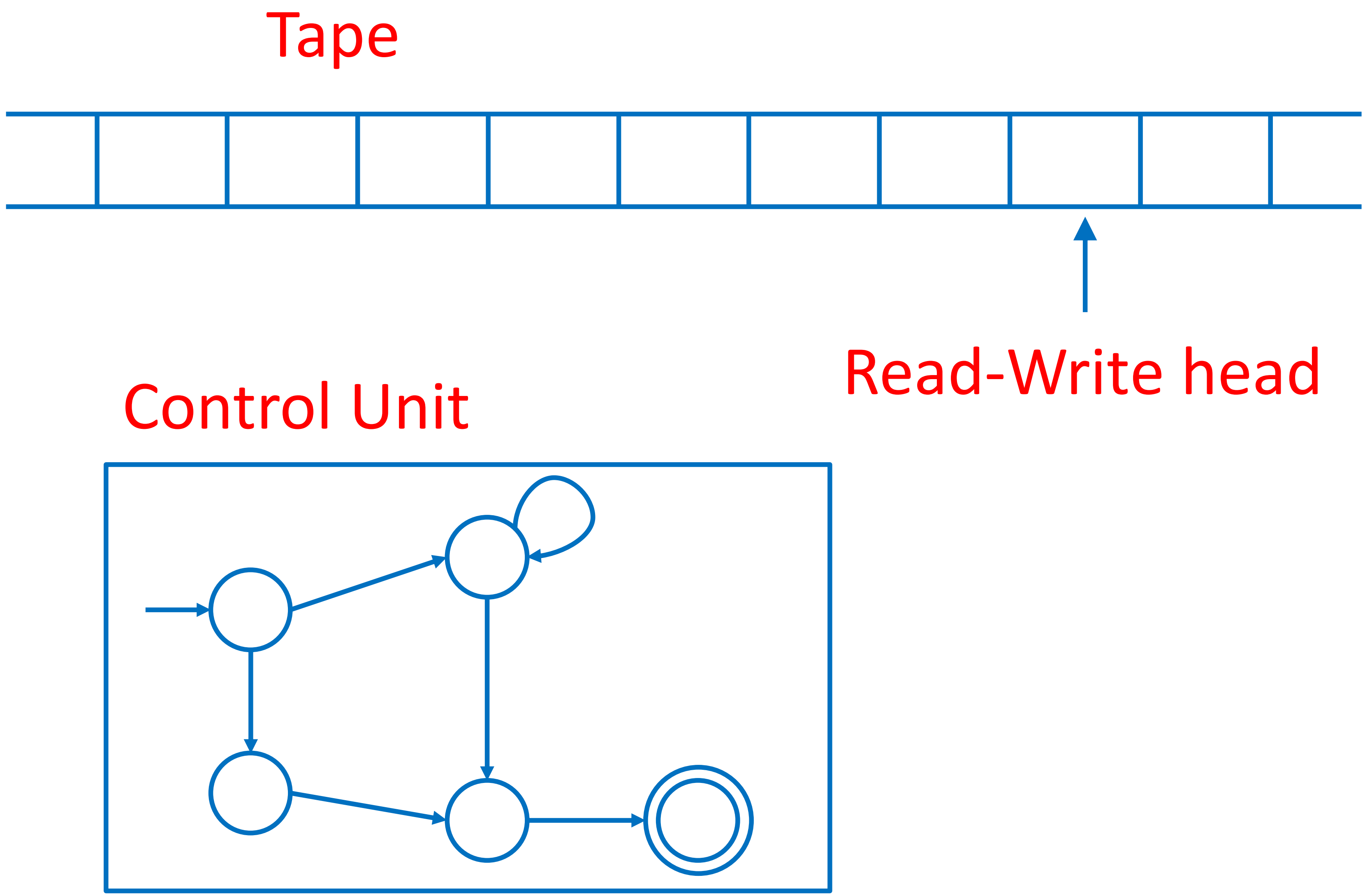
Turing Machines As Transducers

* اَللّٰهُمَّ إِنَّا نَسْأَلُكَ لِسَانًا رَطِبًا بِذِكْرِكَ وَقَلْبًا مُفْعَمًا بِشُكْرِكَ وَبَدَنًا هَيِّنًا لَيِّنًا بِطَاعَتِكَ اَللّٰهُمَّ إِنَّا نَسْأَلُكَ إِيْمَانًا كَامِلًا

وَنَسْأَلُكَ قَلْبًا خَاشِعًا وَنَسْأَلُكَ عِلْمًا نَافِعًا وَنَسْأَلُكَ يَقِيْنًا صَادِقًا وَنَسْأَلُكَ دِيْنًا قَيِّمًا وَنَسْأَلُكَ الْعَافِيَةَ مِنْ كُلِّ بَلِيَّةٍ وَنَسْأَلُكَ تَمَامَ الْغِنَى عَنِ النَّاسِ وَهَبْ لَنَا حَقِيْقَةَ الْإِيْمَانِ بِكَ حَتَّى لَا نَخَافَ وَلَا نَرْجُوْ غَيْرَكَ وَلَا نَعْبُدَ شَيْئًا سِوَاكَ وَاجْعَلْ يَدَكَ مَبْسُوْطَةً عَلَيْنَا وَعَلَى أَهْلِيْنَا وَأَوْلَادِنَا وَمَنْ مَعَنَا بِرَحْمَتِكَ وَلَا تَكِلْنَا إِلَى أَنْفُسِنَا طَرْفَةَ عَيْنٍ وَلَا أَقَلَّ مِنْ ذٰلِكَ يَا نِعْمَ الْمُجِيْبُ.

A Standard Turing Machine



A Turing Machine as a Transducer

Transducer:

- The input for a computation will be all the nonblank symbols on the tape at the initial time.
- At the conclusion of the computation, the output will be whatever is on the tape.

$$q_0 w \vdash^* q_f f(w)$$

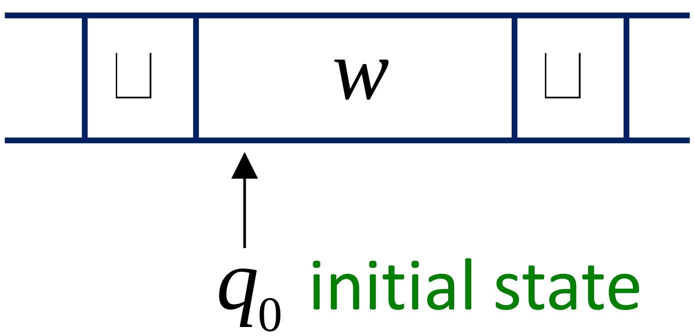
A Computable Functions

Definition: A function f with domain D is said to be Turing-computable or just computable if there exists some Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ such that

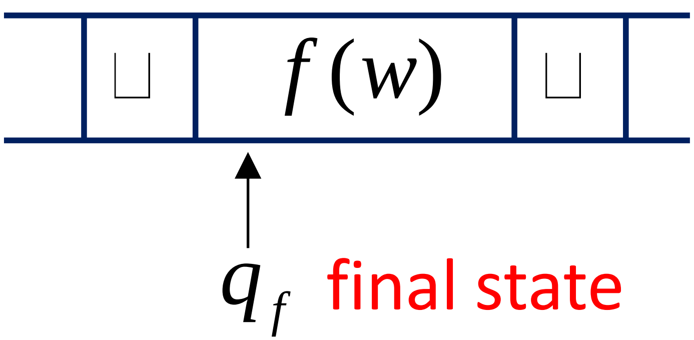
$$q_0 w \vdash_M^* q_f f(w), \qquad q_f \in F$$

for all $w \in D$.

Initial configuration



Final configuration



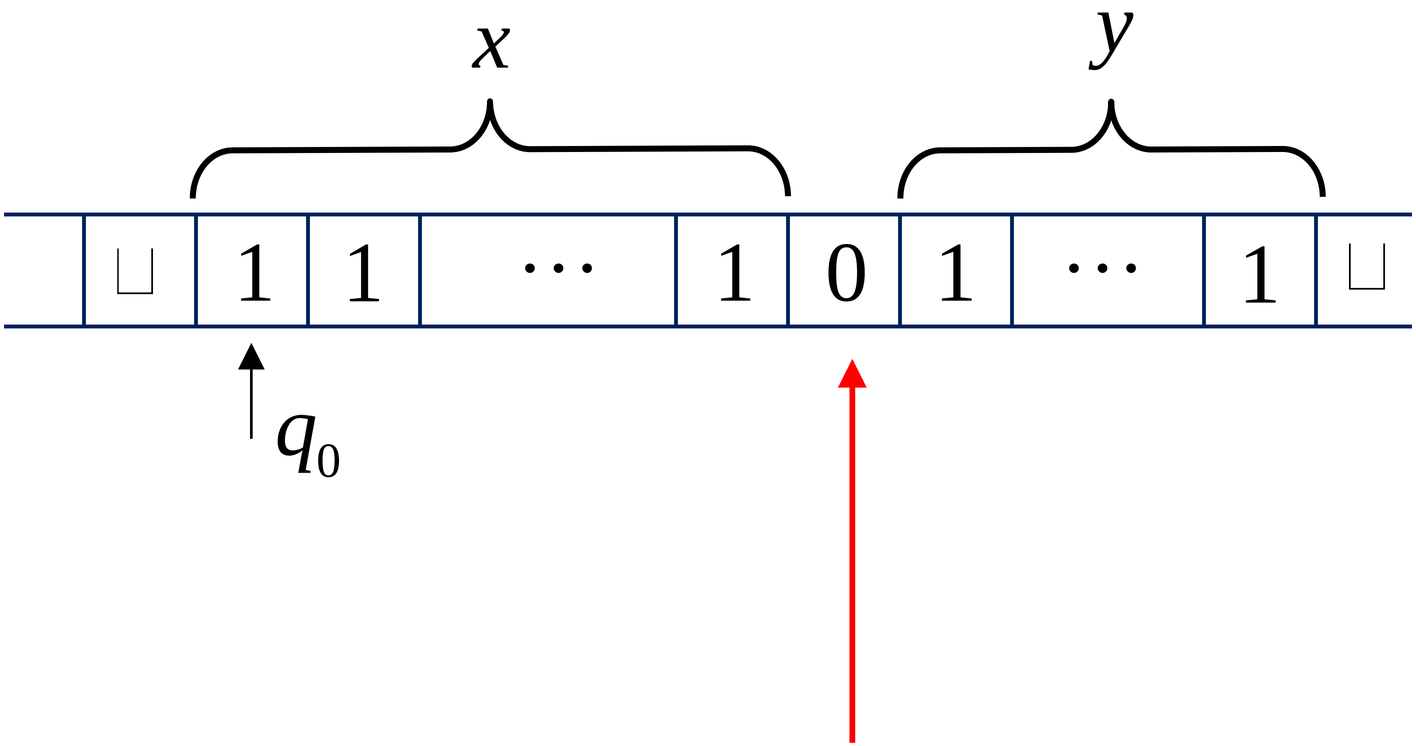
A Computable Functions

Example: Given two positive integers x and y, design a Turing machine that computes $x + y$.

Solution:

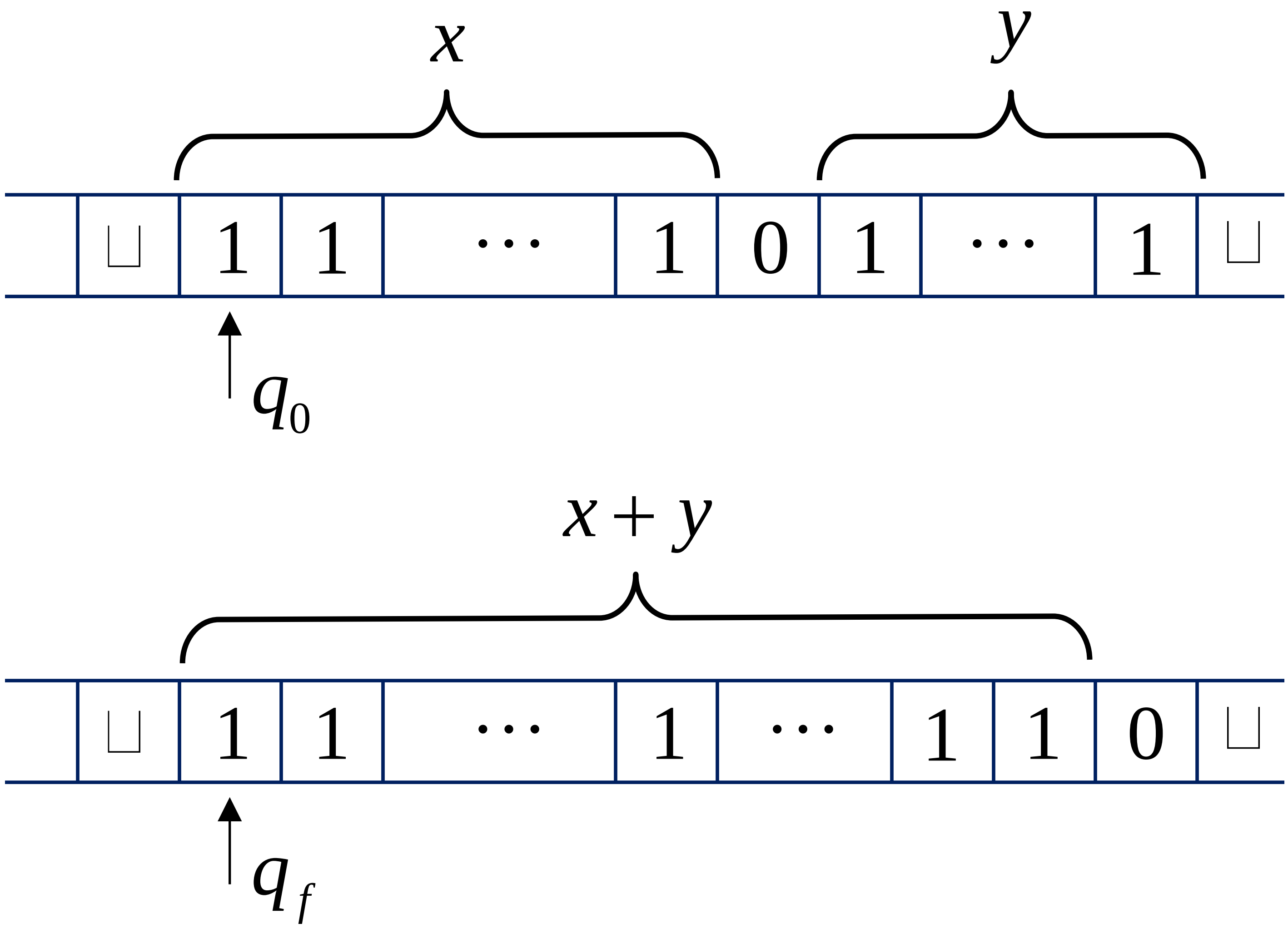
- x and y are represented by $w(x) \in \{x\}^+$ and $w(y) \in \{y\}^+$ such that $|w(x)| = x$ and $|w(y)| = y$.
- $q_0 w(x) 0 w(y) \vdash^* q_f w(x + y) 0$

A Computable Functions



The 0 is the delimiter that separates the two numbers

A Computable Functions



A Computable Functions

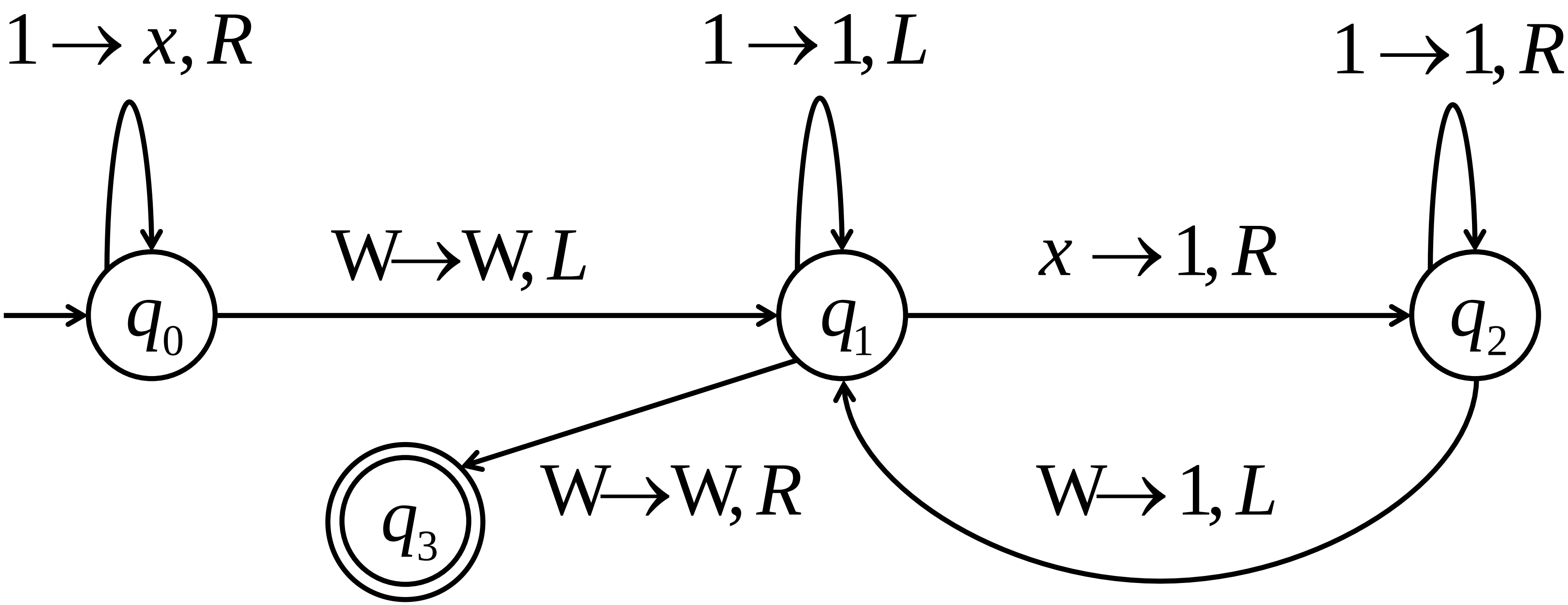
Example: Given a positive integer x, design a Turing machine that computes $2x$ ($w(x) \in \{x\}^+$ with $w(x)| = x$).

Solution: To solve the problem, we implement the following process:

1. Replace every 1 by an x.
2. Find the rightmost x and replace it with 1.
3. Travel to the right end of the current nonblank region and create a 1 there.
4. Repeat Steps 2 and 3 until there are no more x's.

A Computable Functions

Example: Given a positive integer x, design a Turing machine that computes $2x$ ($w(x) \in \{x\}^+$ with $w(x)| = x$).



A Computable Functions

Another Example:

Given positive integers x and y, design a Turing machine that will halt in a final state q_y if $x \geq y$, and that will halt in a non-final state q_n if $x < y$.

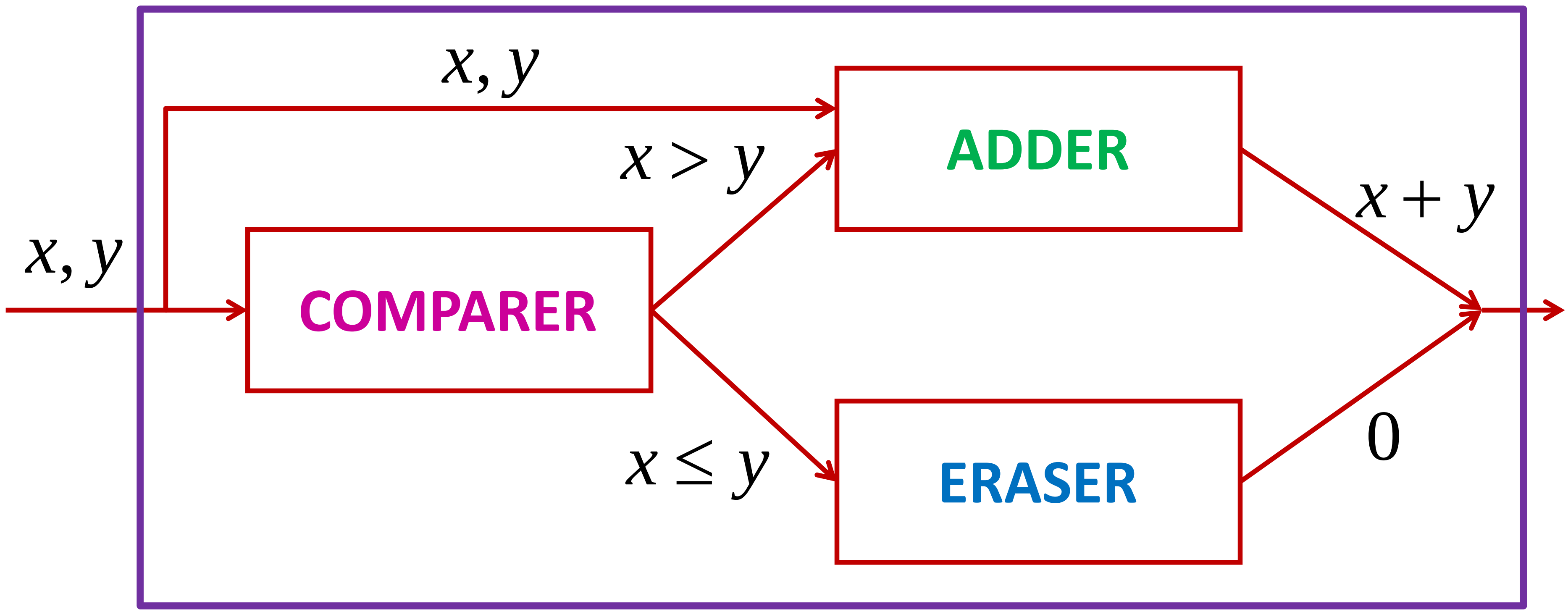
Combining Turing Machines

Combining Turing Machines

Example:

Design a Turing machine that computes the function

$$f(x) = \begin{cases} x + y, & \text{if } x \geq y \\ 0, & \text{if } x < y \end{cases}.$$



Turing’s Thesis

Turing Thesis

Question:

Do Turing machines have the same power with a digital computer?

Intuitive answer: YES

There is no formal answer!!!

Turing Thesis

Turing's thesis:

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

(1930)

Turing Thesis

Computer Science Law:

A computation is mechanical if and only if it can be performed by a Turing Machine

There is no known model of computation more powerful than Turing Machines

Turing Thesis

Definition of Algorithm:

An algorithm for function $f(x)$ is a Turing Machine which computes $f(x)$.

Turing Thesis

Algorithms are Turing Machines

When we say:

There exists an algorithm

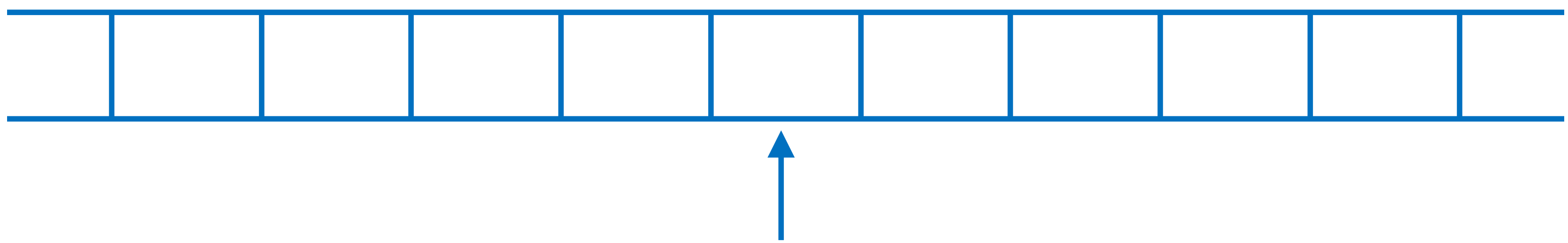
We mean:

There exists a Turing Machine that executes the algorithm

Variations of the Turing Machine

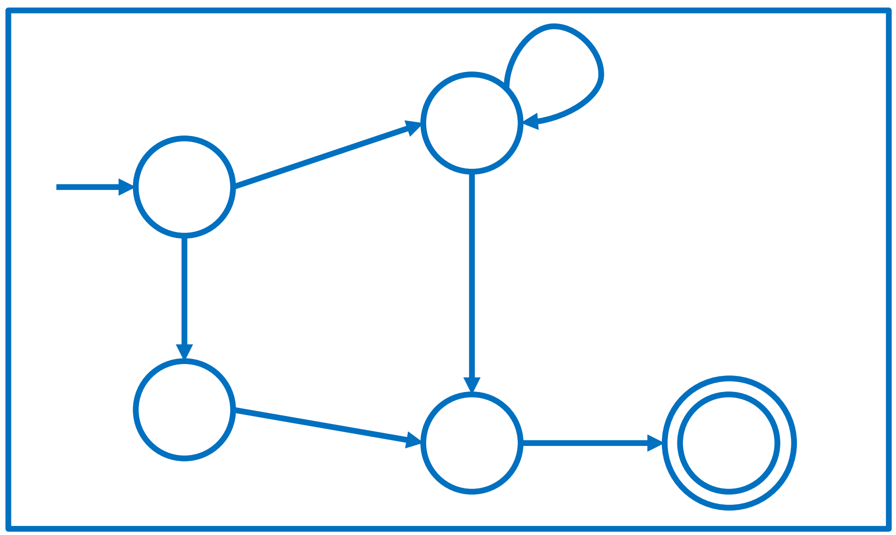
A Standard Turing Machine

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

Variations of the Standard Model

- The variations form different **Turing Machine Classes**

BUT

- Each class has the **same power** with the **Standard Model**

Variations of the Standard Model

- Same power of two classes means:

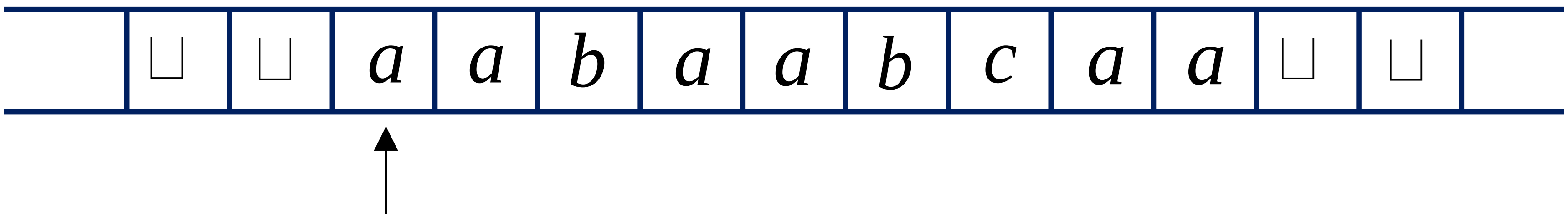
For any machine M_1 of the first class, there is a machine M_2 in the second class such that

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

and vice versa.

Stay-Option

The head can stay in the same position



Left, Right, Stay

L,R,S: moves

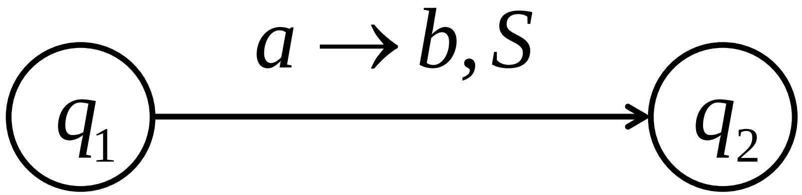
Stay-Option

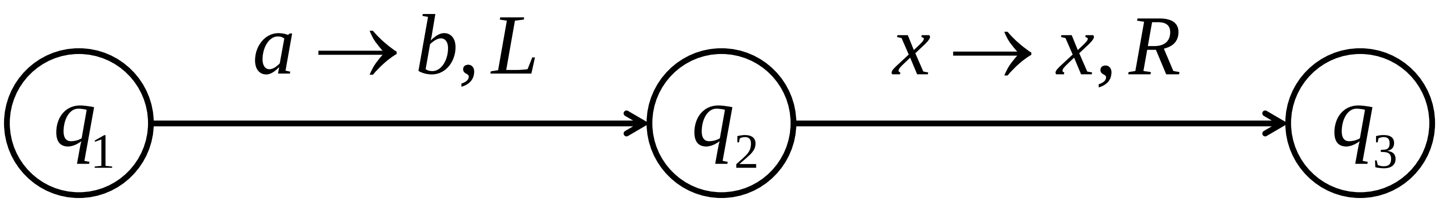
$$q_1 \xrightarrow{a \rightarrow b, L} q_2$$

$$q_1 \xrightarrow{a \rightarrow b, R} q_2$$

$$q_1 \xrightarrow{a \rightarrow b, S} q_2$$

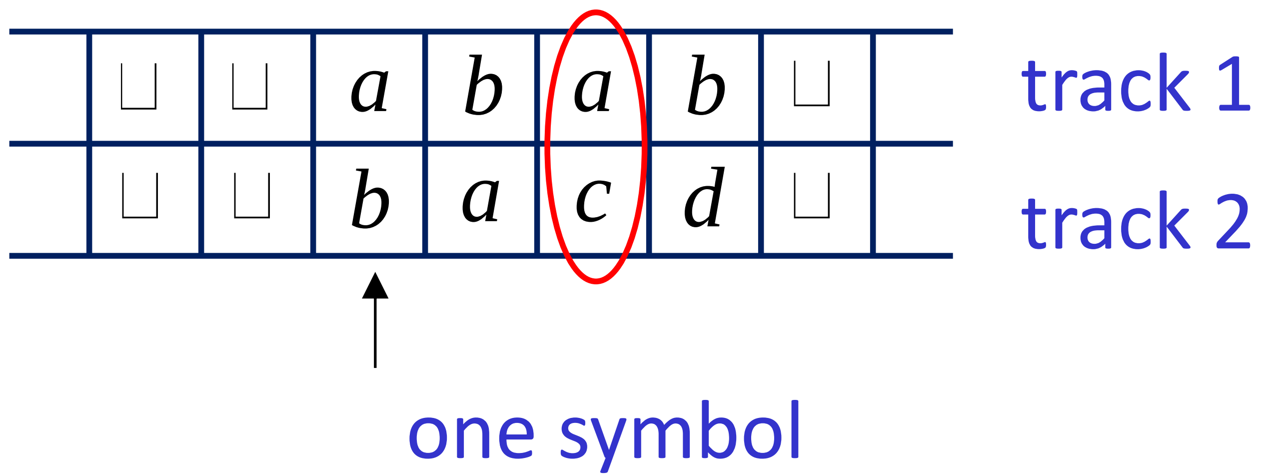
Stay-Option



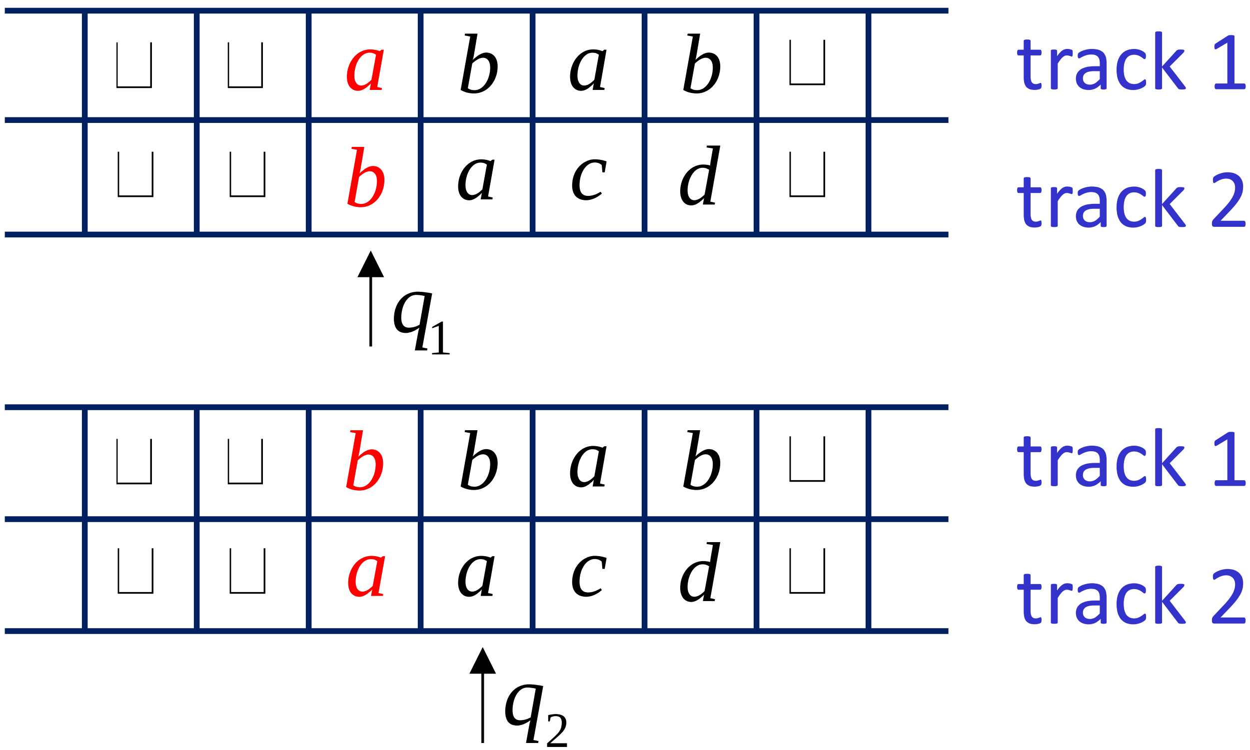


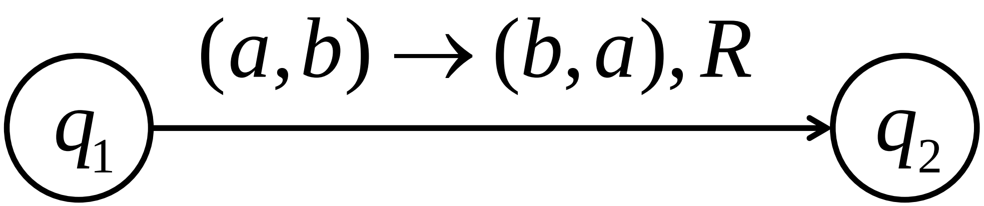
For every symbol $x \in \Sigma$

Multiple Track Tape

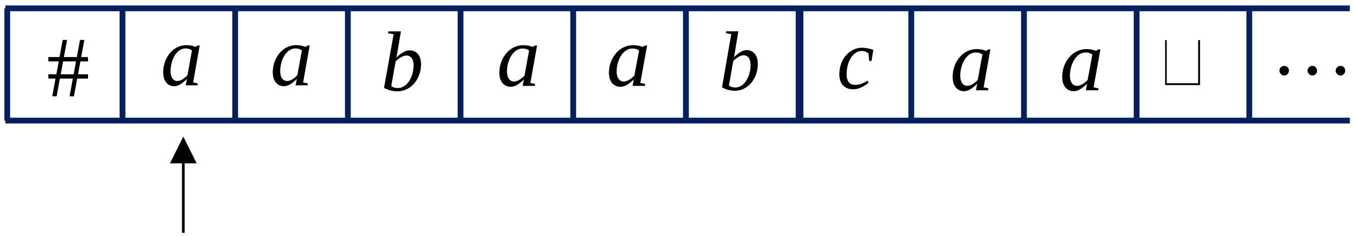


Multiple Track Tape

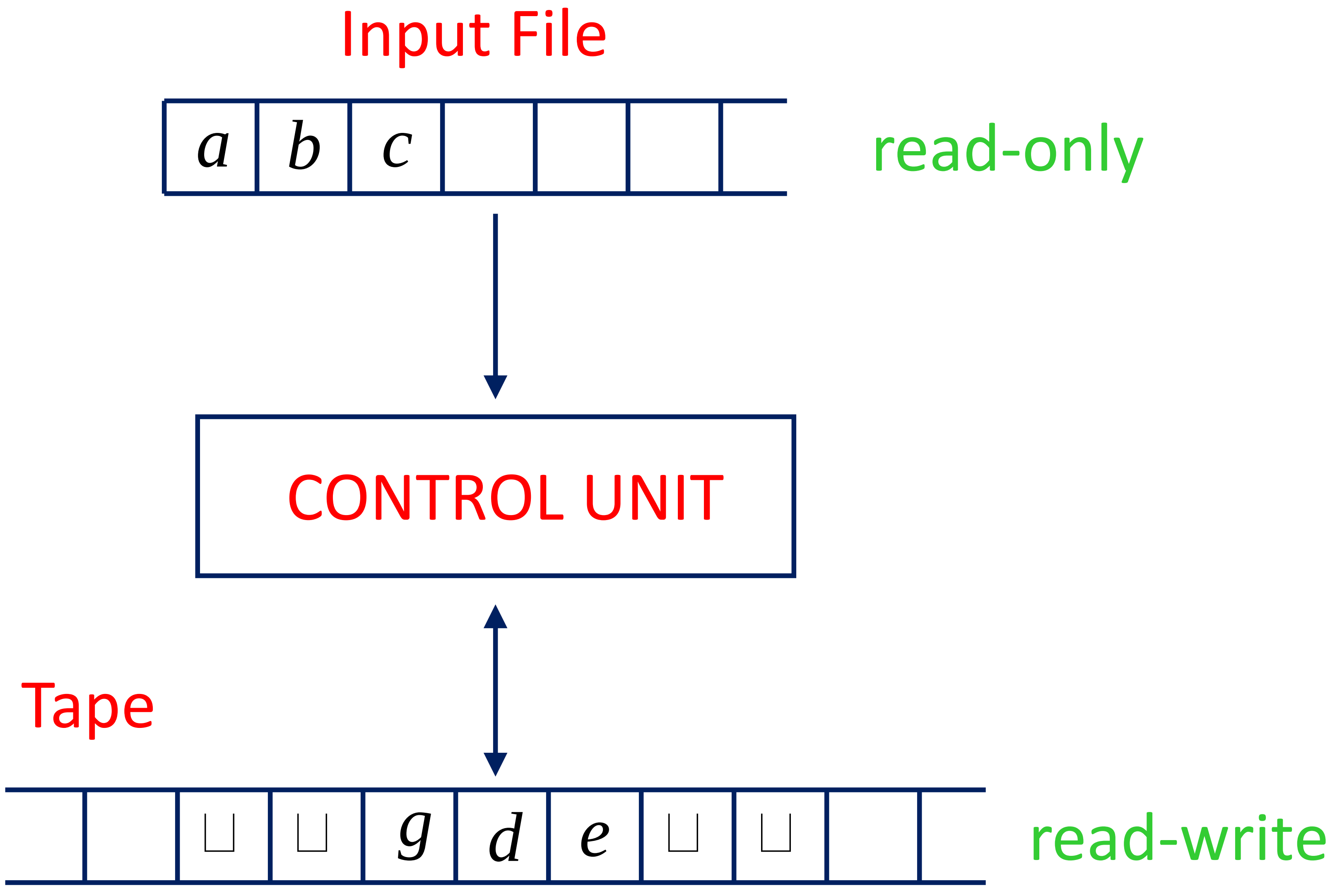




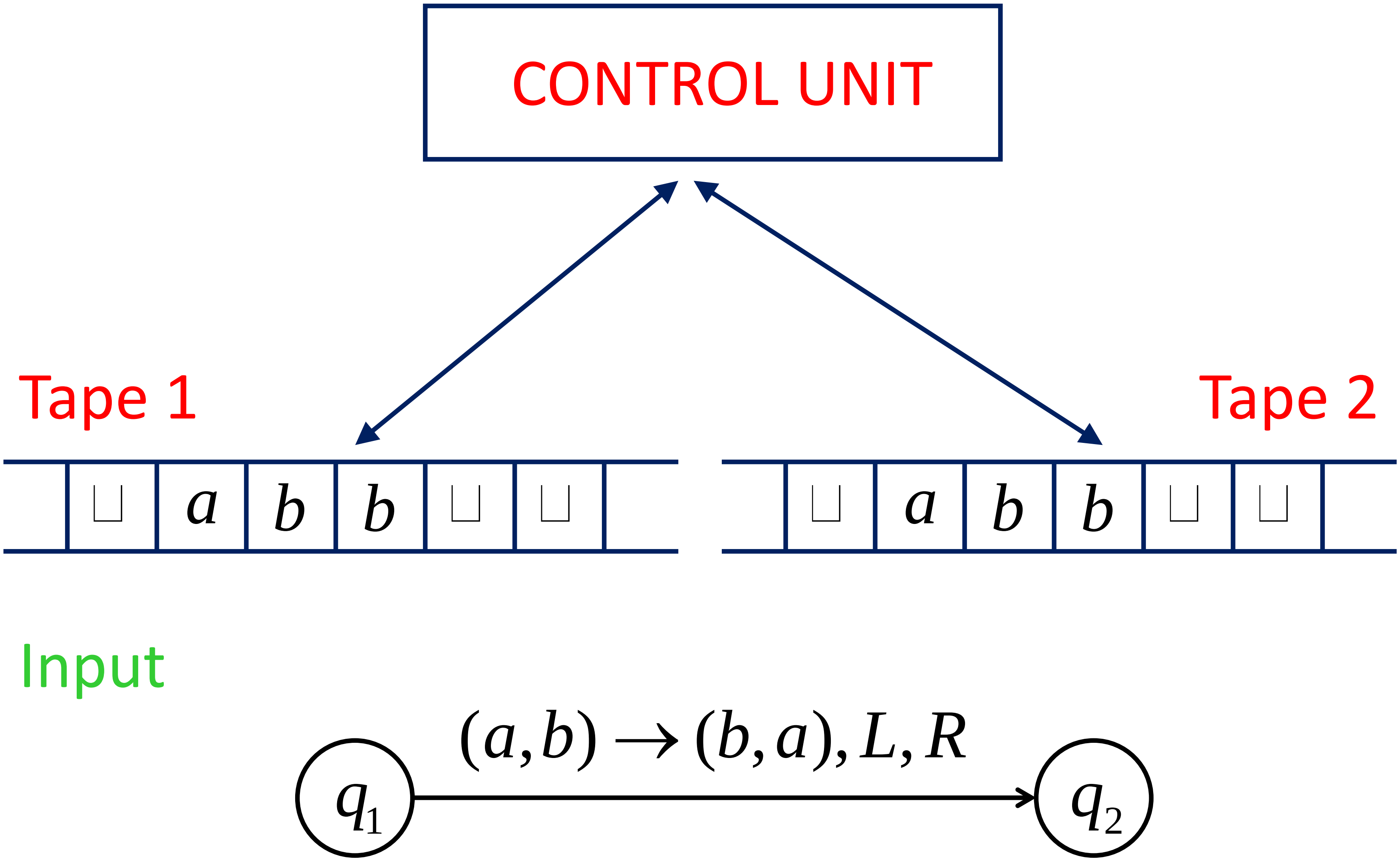
Semi-Infinite Tape



The Off-Line Machine



Multitape Turing Machines



Multitape Turing Machines

Same power doesn't imply same speed:

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

	Acceptance Time
Standard machine	n^2
Two-tape machine	n

Multitape Turing Machines

Standard machine:

Go back and forth n^2 times

Two-tape machine:

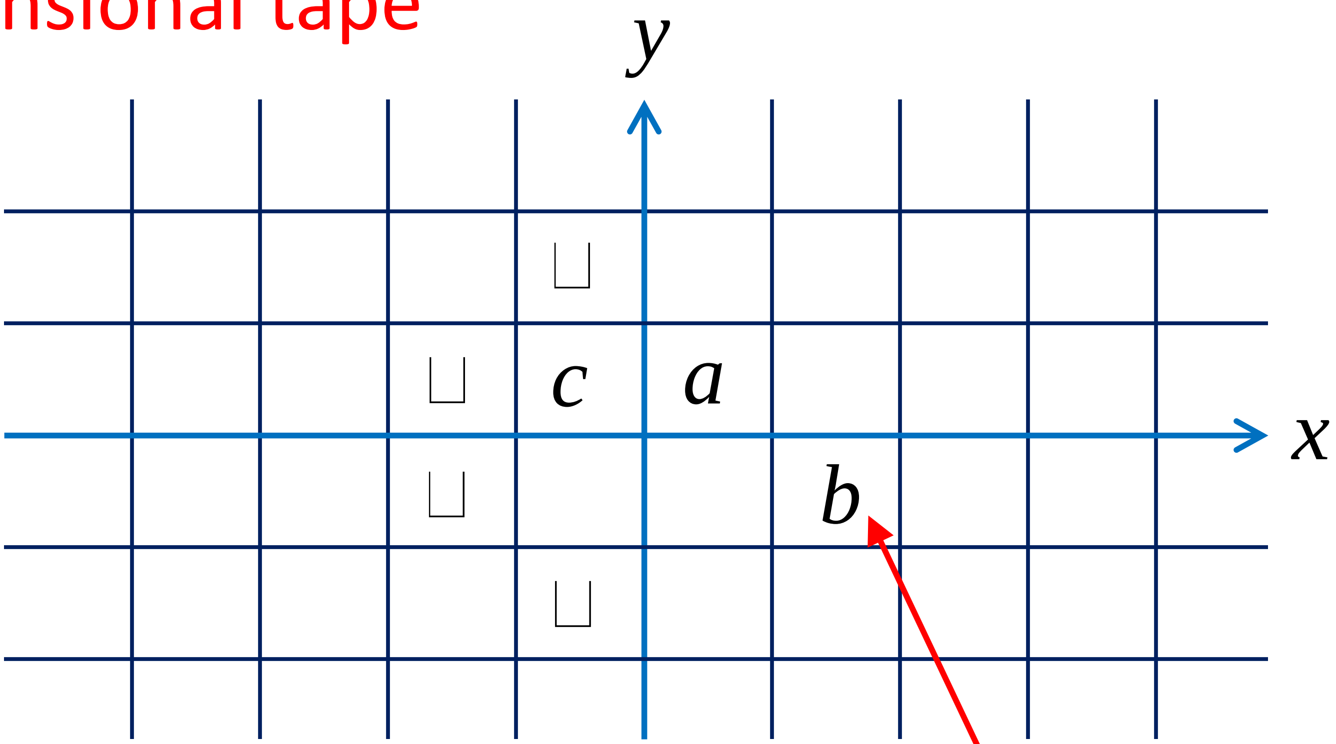
Copy b^n to tape 2 (n steps)

Leave a^n on tape 1 (n steps)

Compare tape 1 and tape 2 (n steps)

Multi-Dimensional Turing Machines

Two-dimensional tape



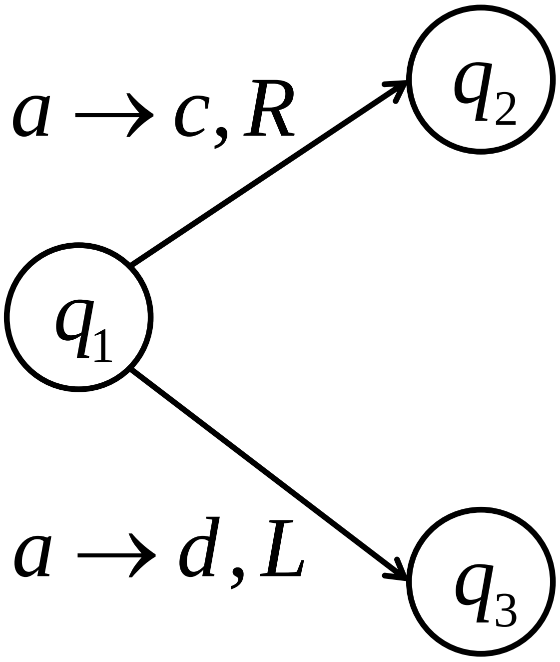
MOVES: L,R,U,D

U: up   D: down

HEAD

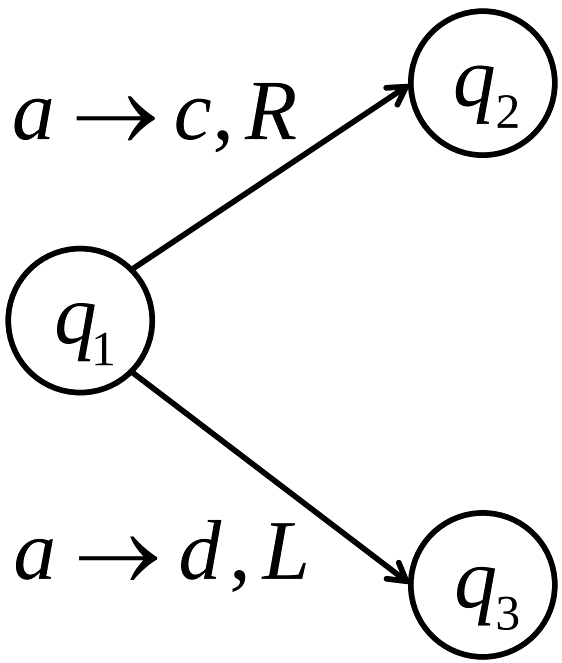
Position: (+2,-1)

Non-deterministic Turing Machines

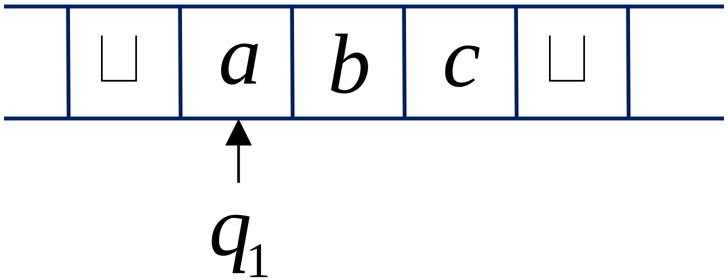


Non-deterministic Choice

Non-deterministic Turing Machines



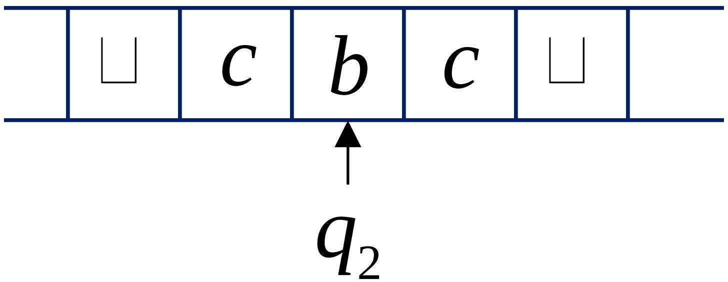
Time 0

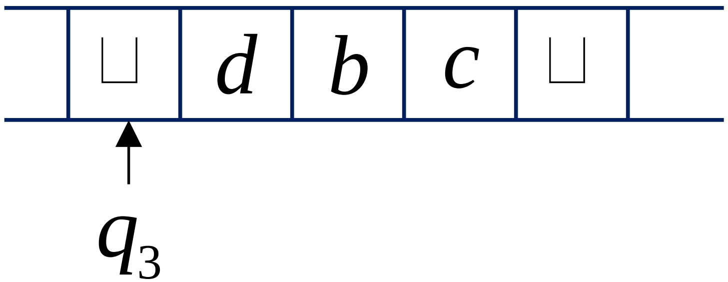


Choice 1

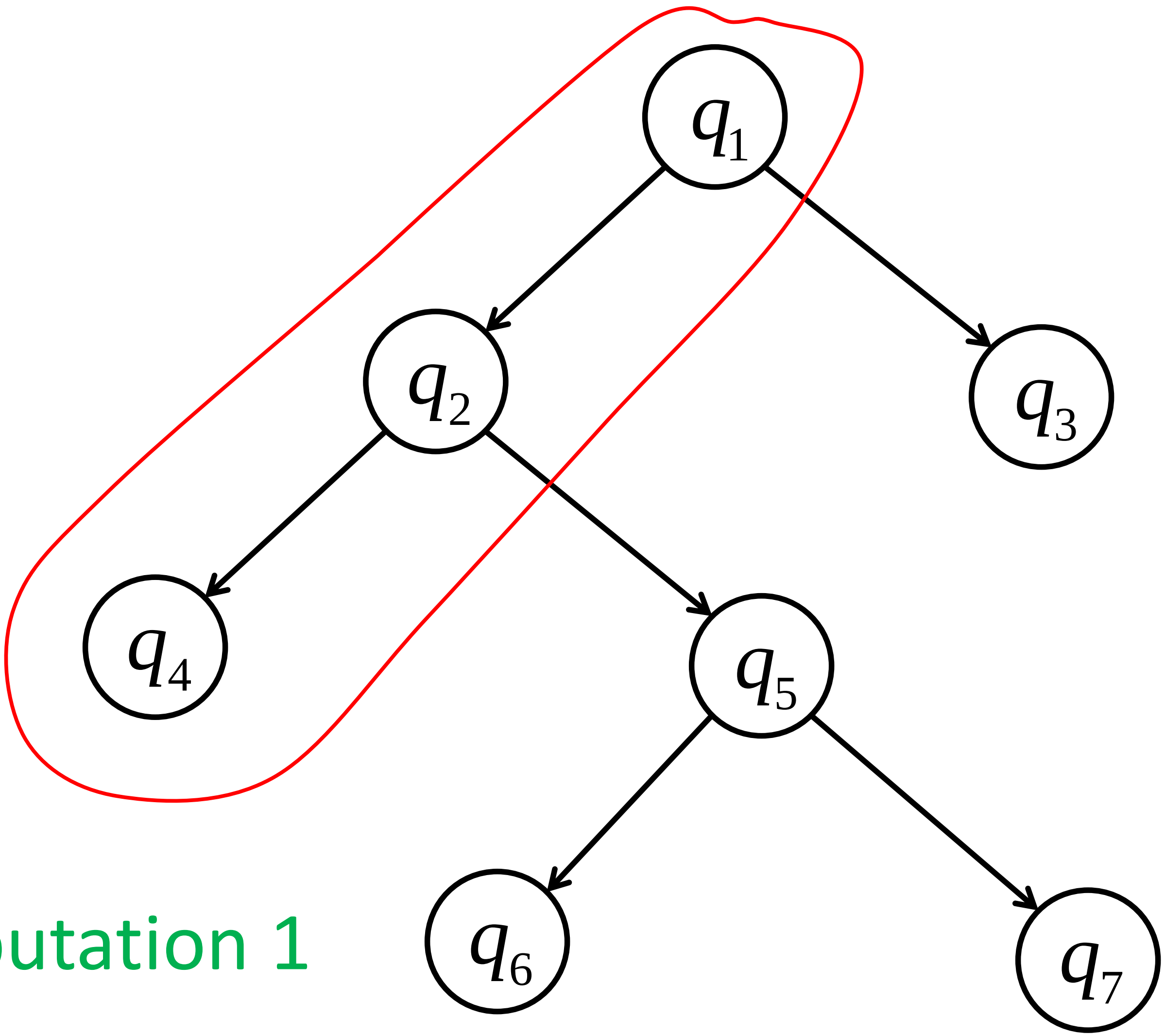
Time 1

Choice 2



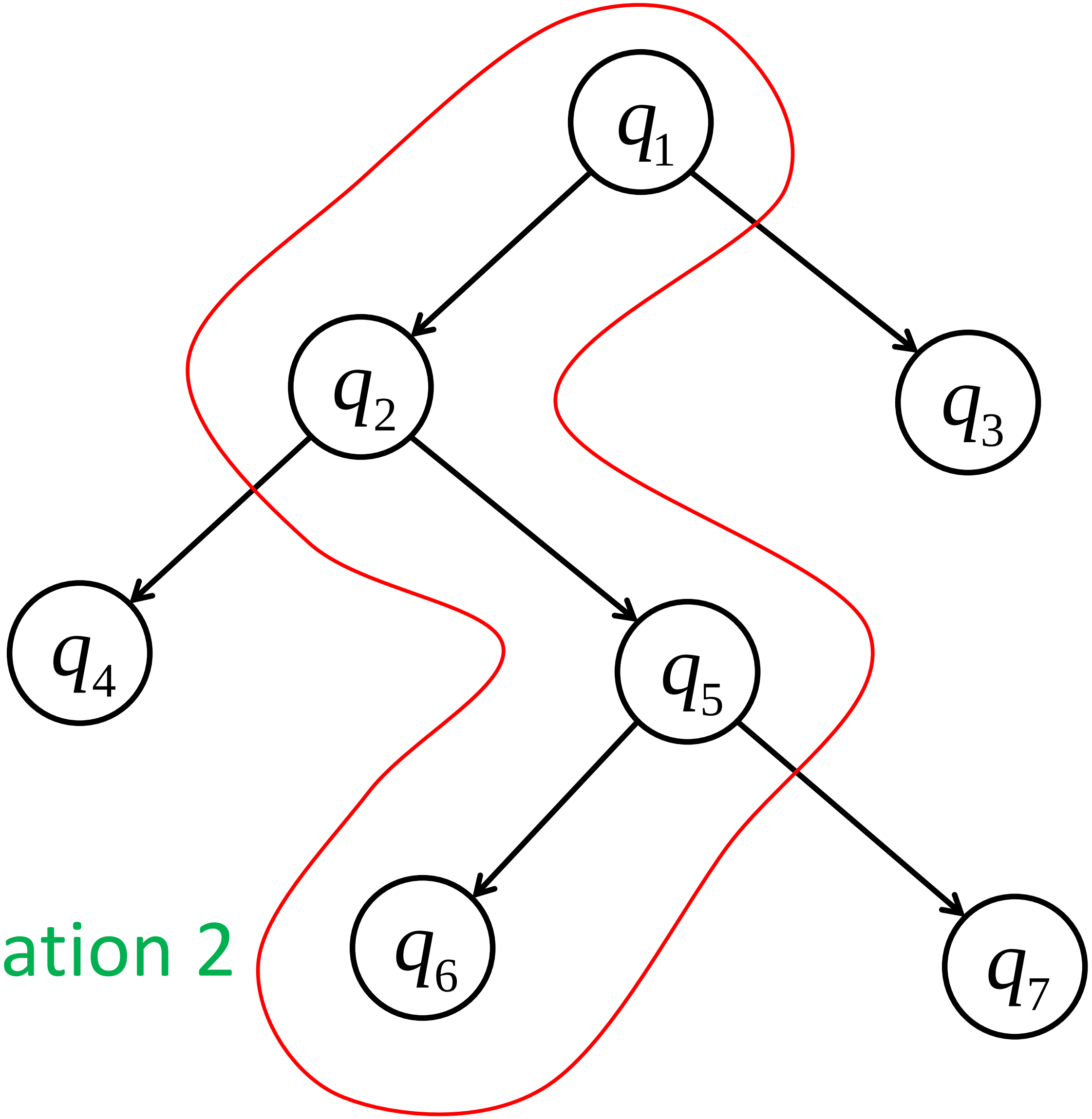


Non-deterministic Turing Machines



Computation 1

Non-deterministic Turing Machines



Computation 2

Non-deterministic Turing Machines

Input string w is accepted if

there is a possible computation

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

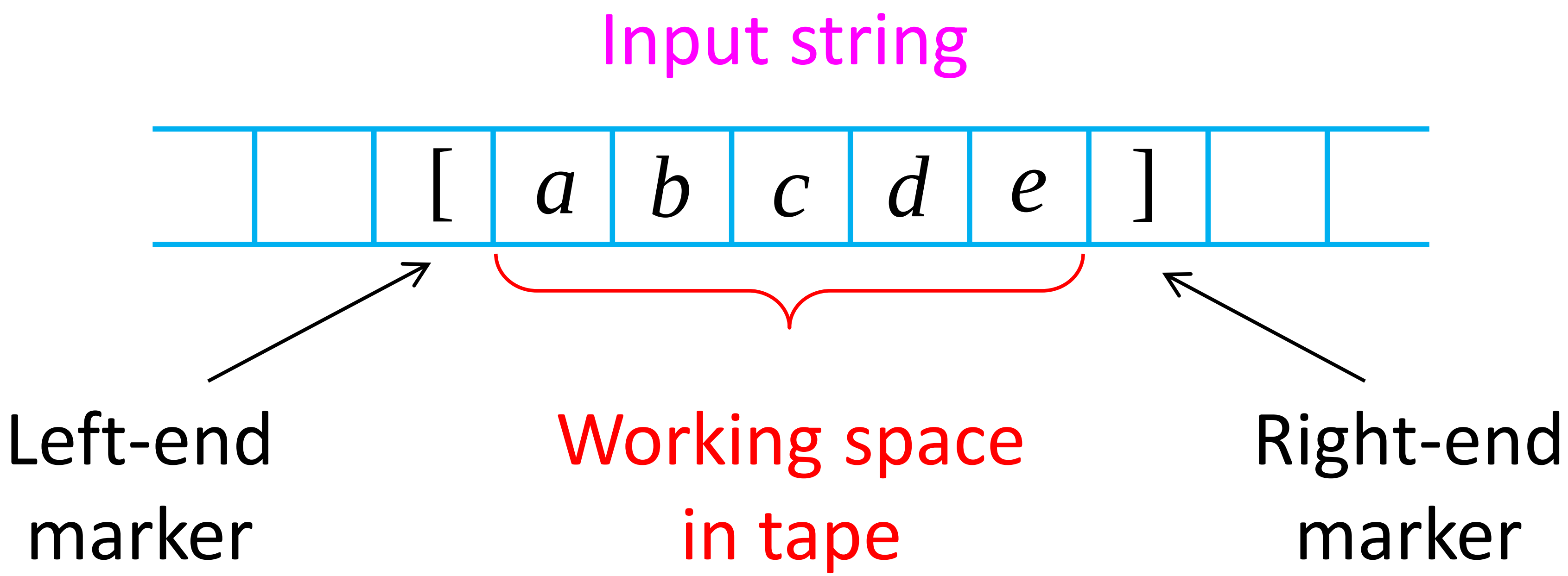
Initial configuration

Final Configuration

Final state

Linear Bounded Automata

- **Linear Bounded Automata (LBAs)** are the same as Turing Machines with one difference: the input string tape space is the only tape space allowed to use.



Linear Bounded Automata

- Example languages accepted by LBAs:

$$L = \{a^n b^n c^n\} \qquad\qquad L = \{a^{n!}\}$$

- LBA have less power than Turing Machines

Exercises

- Construct a nondeterministic Turing machine that accepts the language

$$L = \{ww : w \in \{a, b\}^+\}$$

- Construct a nondeterministic Turing machine that accepts the language

$$L = \{ww^R w : w \in \{a, b\}^+\}$$

Home Assignment/Quiz3 (10pts. 7 Jan 2023)

1. Show that $L = \{a^n b^k c^m : n = k + m\}$ is not regular language (use Pumping Lemma) and find a context-free grammar generating L.

2. Eliminate useless productions from the grammar.

$$S \to aAS \mid aBC \mid aDE, \qquad A \to bAA \mid bBF \mid bDG \mid b$$

$$B \to bAB \mid bBH \mid bDI \mid a, \qquad C \to b$$

$$D \to bAD \mid bBJ \mid bDE, \qquad K \to aAK \mid aBL \mid aDG$$

$$M \to aAM \mid aBN \mid aDI$$

3. Convert grammar into Chomsky Normal Form

$$S \to XaY \mid bc, \qquad X \to bac, \qquad Y \to bbSX$$

4. Find a context-free grammar generating $L = \{a^n b^m : n \leq m \leq 2n\}$

5. Show that $L = \{a^n b^m c^m : n \geq m\}$ is not a context-free language (use Pumping Lemma) and construct a Turing machine accepting L.