Theory of Automata Computable Functions

Week 15

Contents

- Computation
  - Paradigms
  - Functions and their computations
  - Computable and non-computable functions
  - Computable Functions
- Computer Science Law
- TM for Computable Functions
- Alonzo Church's Thesis (1936)

What is Computation?

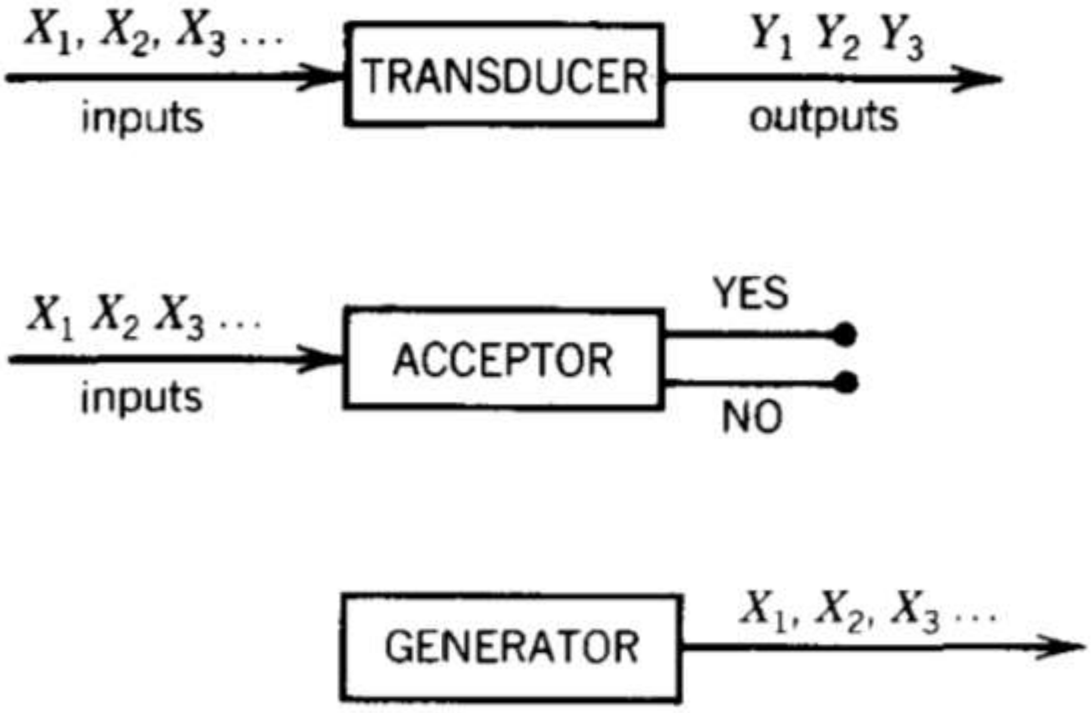
- **Paradigm instances of computation**
  - Operations on numeric data
  - Operations on character data
  - Function computation

Real computers are finite devices, not infinite, like the Turing machine – they can be understood by a circuit model of computation

Three Computational Paradigms

- Language acceptance (recognition)
- Transduction (transformation of an input into an appropriate output)
  - Function computation (natural numbers only)
- Language Generation (to list out all the words in the language, also known as enumerating the language)

- **Turing machines** implement each of these paradigms



$X_1, X_2, X_3 \ldots$
TRANSDUCER
$Y_1\ Y_2\ Y_3$
inputs
outputs
$X_1\ X_2\ X_3 \ldots$
ACCEPTOR
YES
NO
inputs
GENERATOR
$X_1, X_2, X_3 \ldots$

Functions and Their Computation

- Function
  - A correspondence between a collection of possible input values and a collection of output values
  - Each possible input is assigned a unique output
- Computing the function
  - Process of determining the particular output value that assigns to a given input

Computable and Non-computable Functions

- Computable functions

  Functions whose output values can be determined algorithmically from their input values

- Non-computable functions

  Functions that are so complex that there is no well-defined, step-by-step process for determining their output based on their input values

  The computation of these functions lies beyond the abilities of any algorithmic system

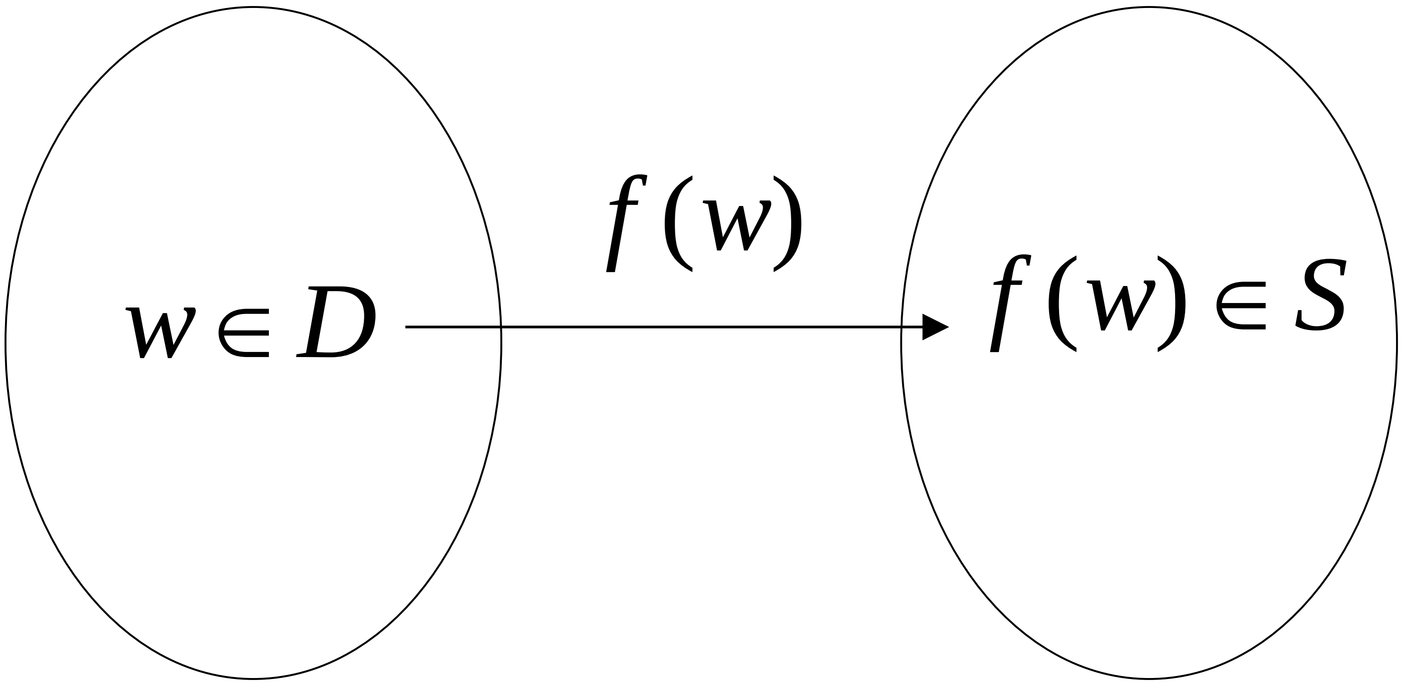
A Turing Machine

- Captures the essence of computational process
- Its computational power is as great as any algorithmic system
- If a problem can not be solved by a Turing machine, it can not be solved by any algorithmic system
- Represents a theoretical bound on the capabilities of actual machines

A function $f(w)$ has:

Domain: D

Result Region: S



Integer Domain

Decimal: 5

Binary: 101

Unary: 11111

We prefer **unary** representation:

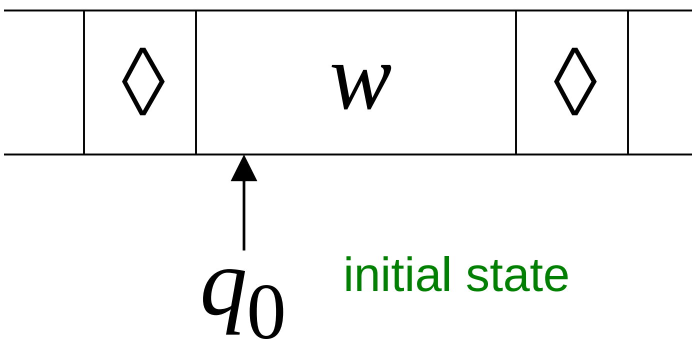
easier to manipulate with Turing machines

Computable Function

- If TM takes a sequence of numbers as input and leaves only one number as output, we say that the computer has acted like a mathematical ***function***. Any operation that is defined on all sequence of ***K*** numbers (for some ***K ≥1***) and that can be performed by a TM is called ***Turing Computable*** or just ***computable***.
- Example addition, subtraction, max, min, multiplication are computable function and defined for ***K*** = 2.
- Identity, successor are computable functions defined for K = 1, so there is only one input

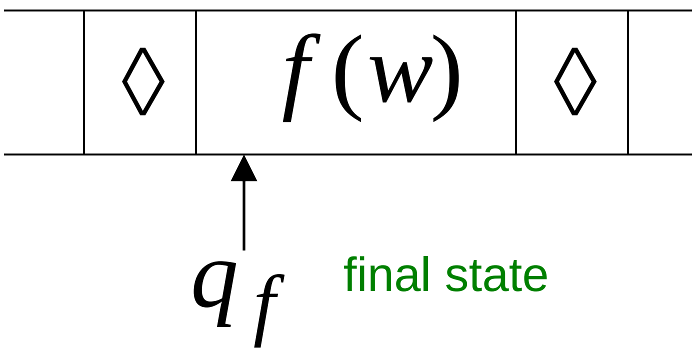
A function f is computable if
there is a Turing Machine M such that:

Initial configuration



initial state

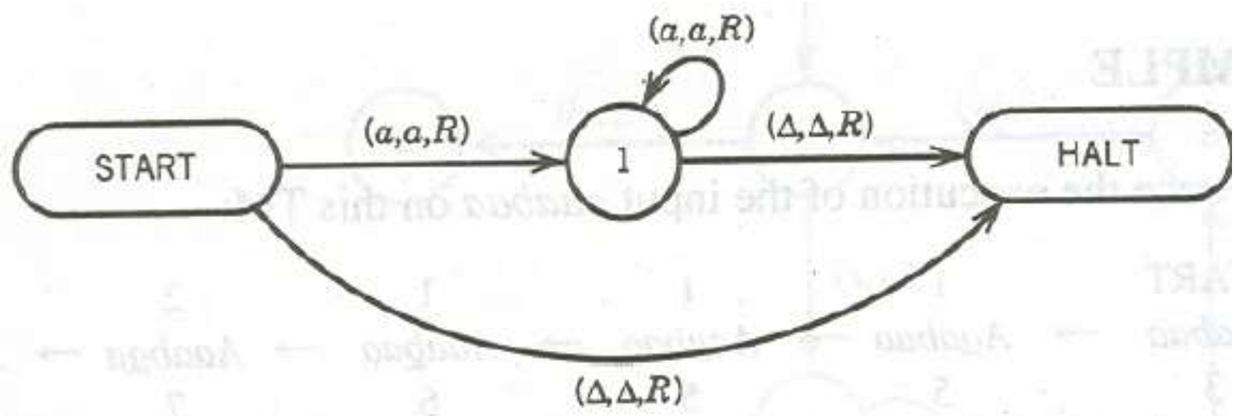
Final configuration

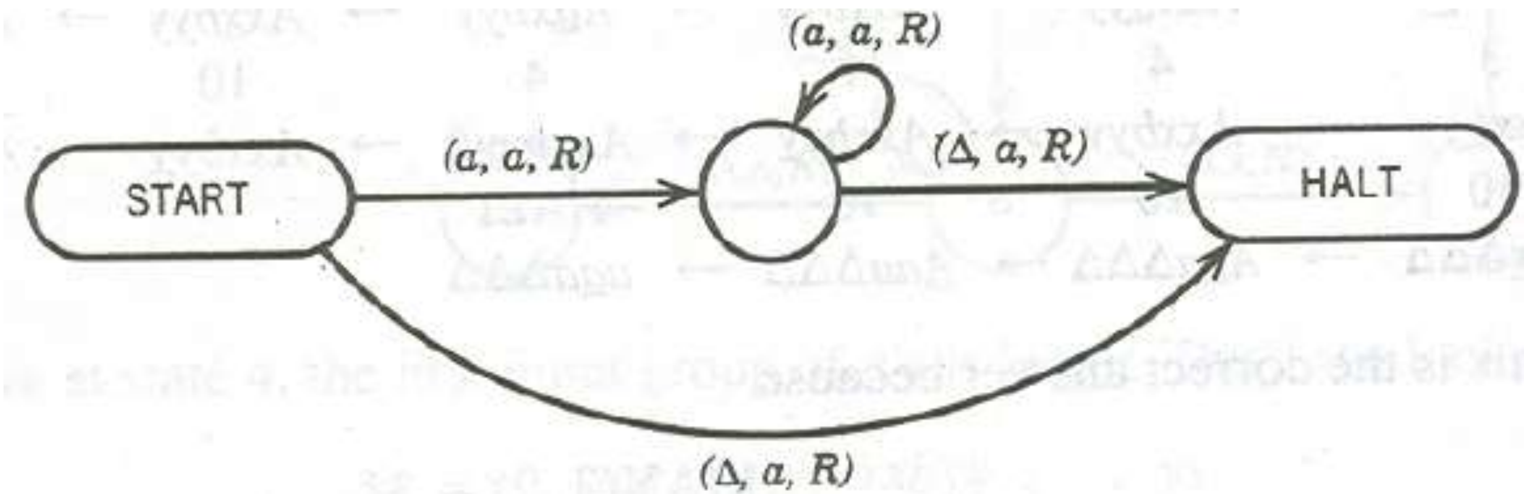


final state

For all $w \in D$ Domain

Identity TM





Successor TM

Example

The function $f(x, y) = x + y$ is computable

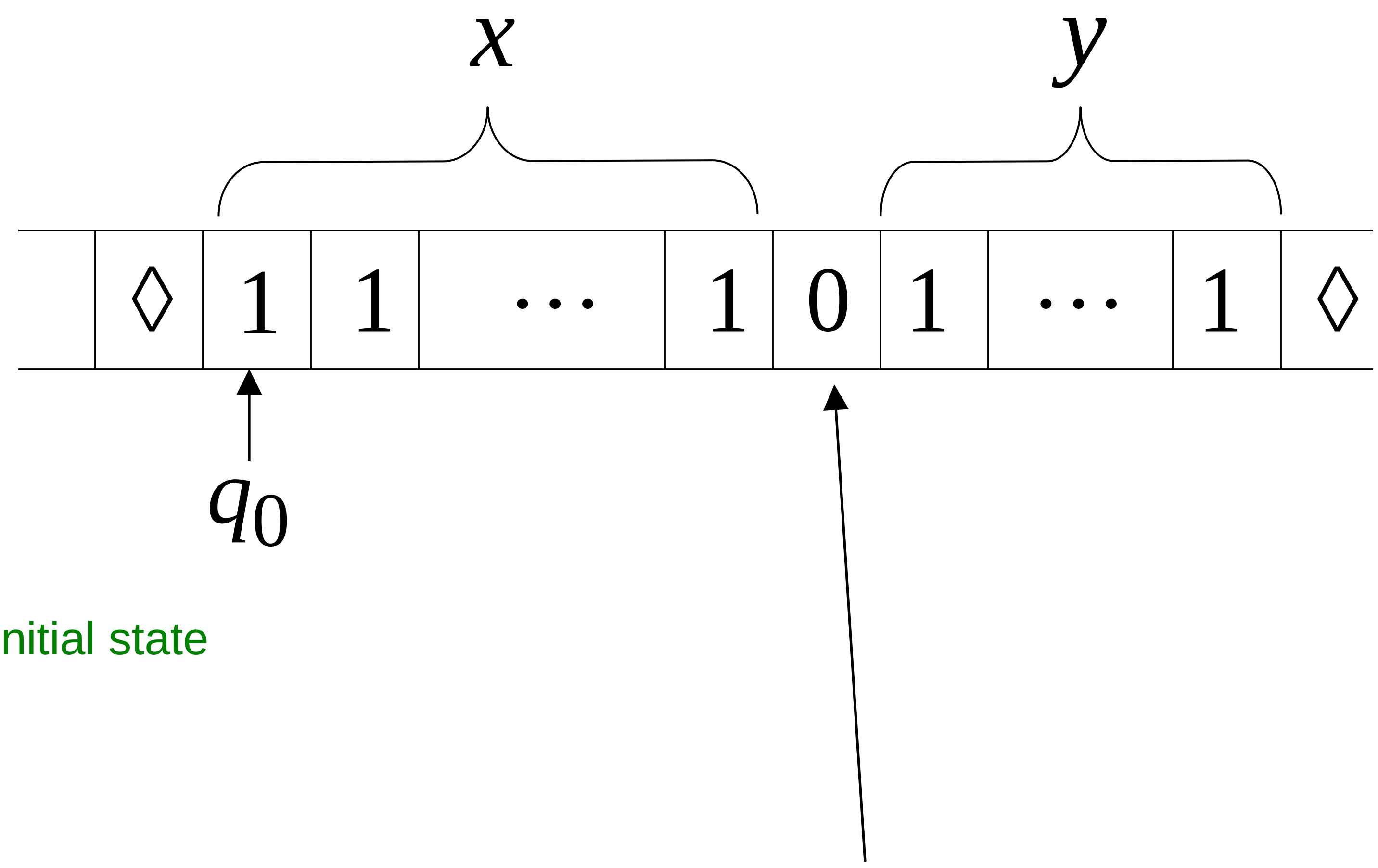
x, y are integers

Turing Machine:

Input string: $x0y$ unary

Output string: $xy0$ unary

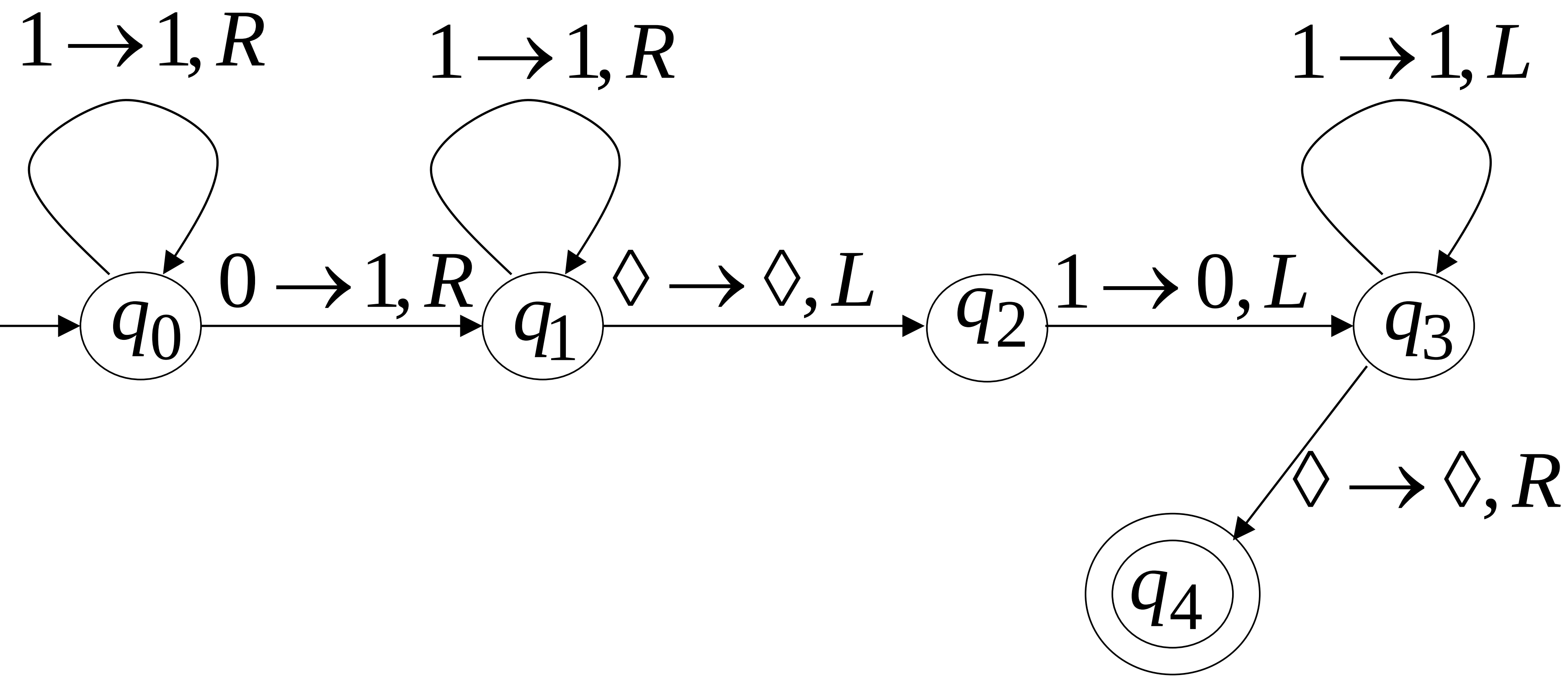
Start



initial state

The 0 is the delimiter that separates the two numbers

Turing machine for function $f(x, y) = x + y$

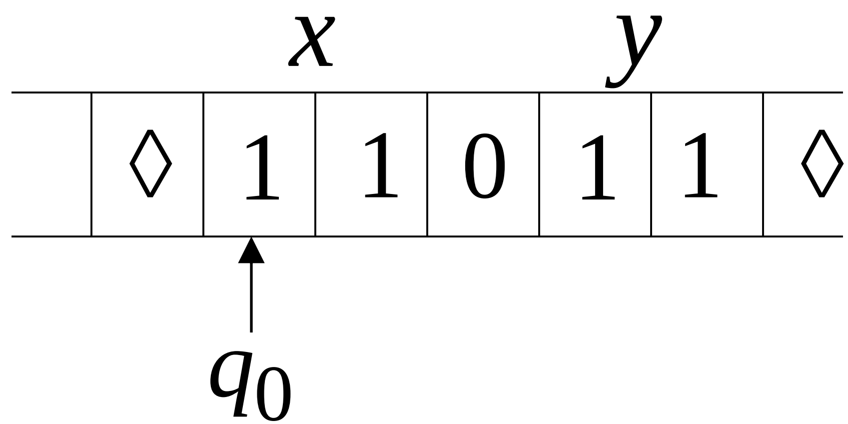


Execution Example:

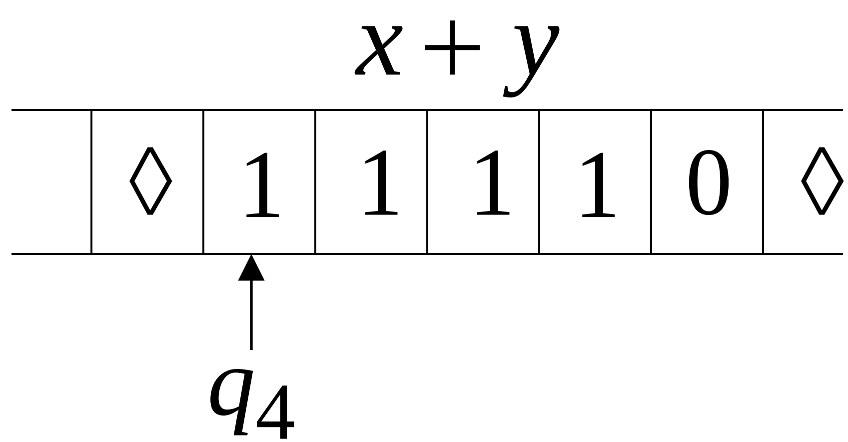
$$x = 11 \quad (2)$$

$$y = 11 \quad (2)$$

Time 0



Final Result



Another Example

The function $f(x) = 2x$ is computable

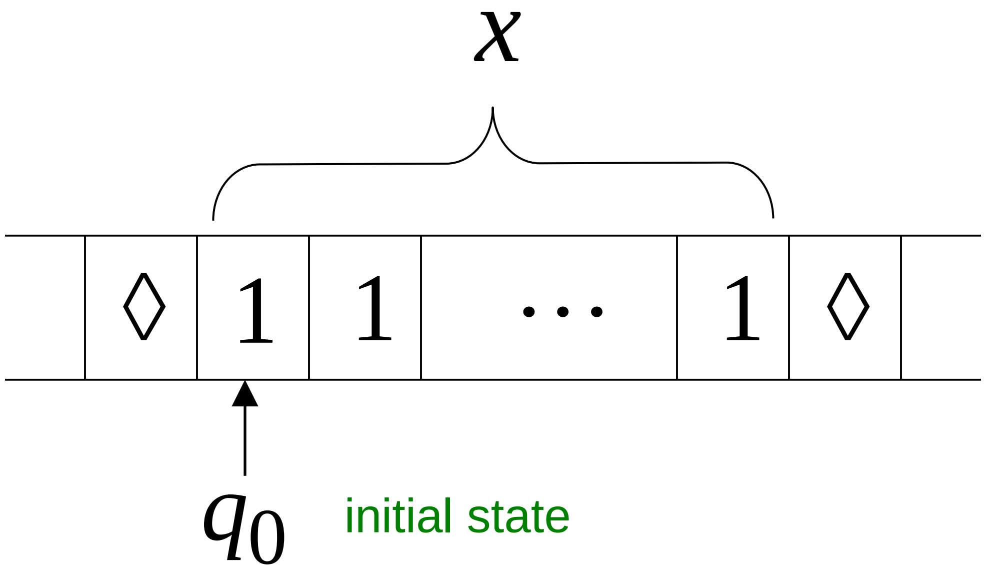
x is integer

Turing Machine:

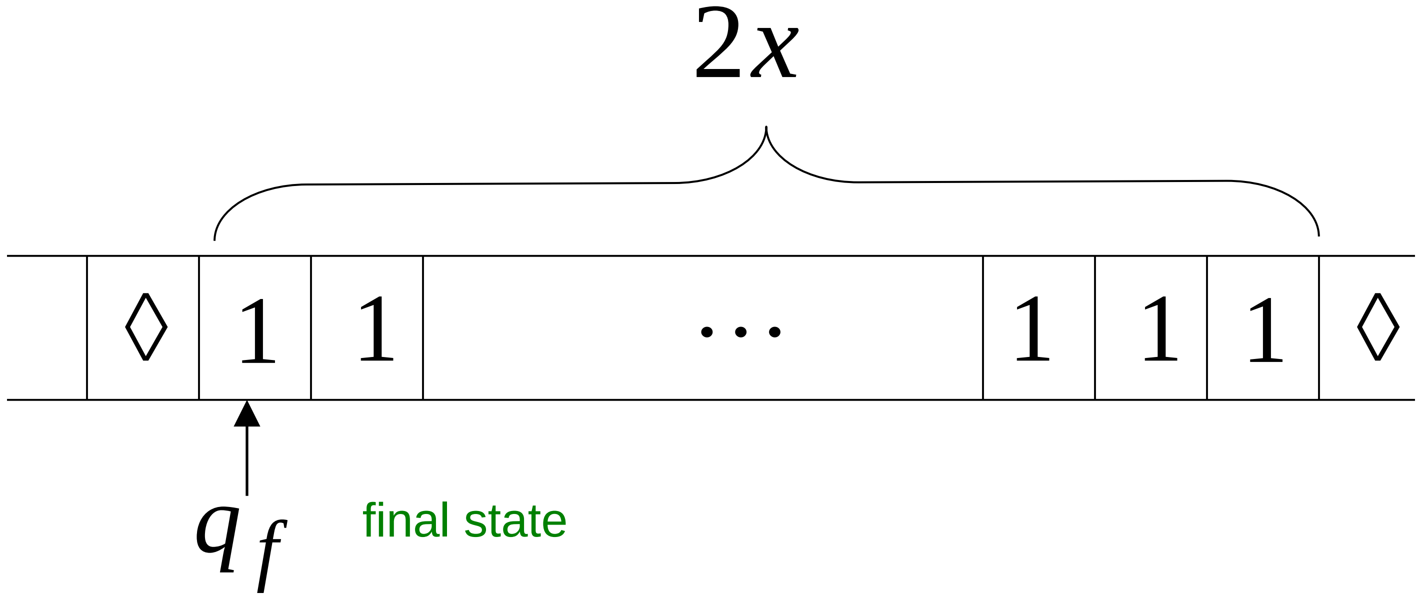
Input string: x unary

Output string: xx unary

Start



Finish



Turing Machine Pseudocode for $f(x) = 2x$

- Replace every 1 with $

- Repeat:

    - Find rightmost $, replace it with 1

    - Go to right end, insert 1

Until no more $ remain

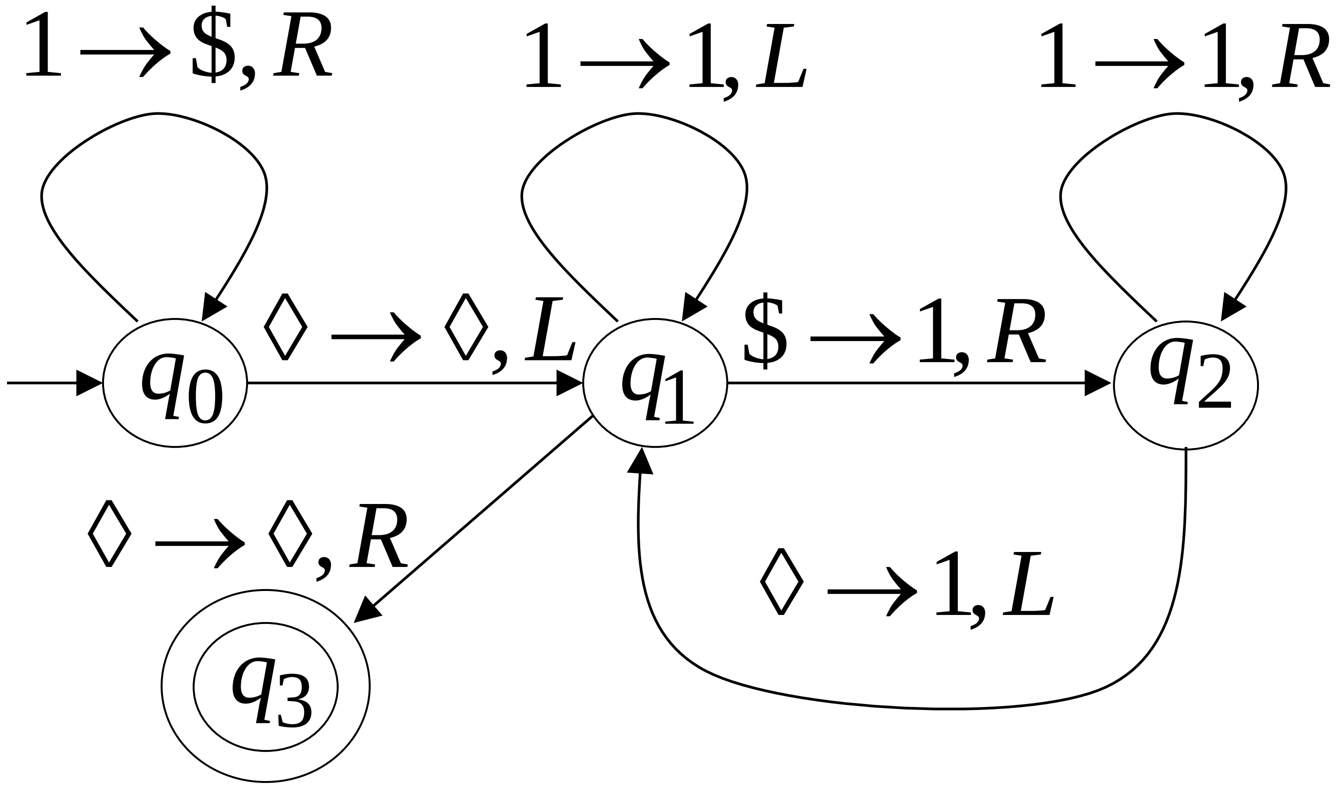
Computer Science Law:

A computation is mechanical iff it can be performed by a Turing Machine

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

Turing Machine for

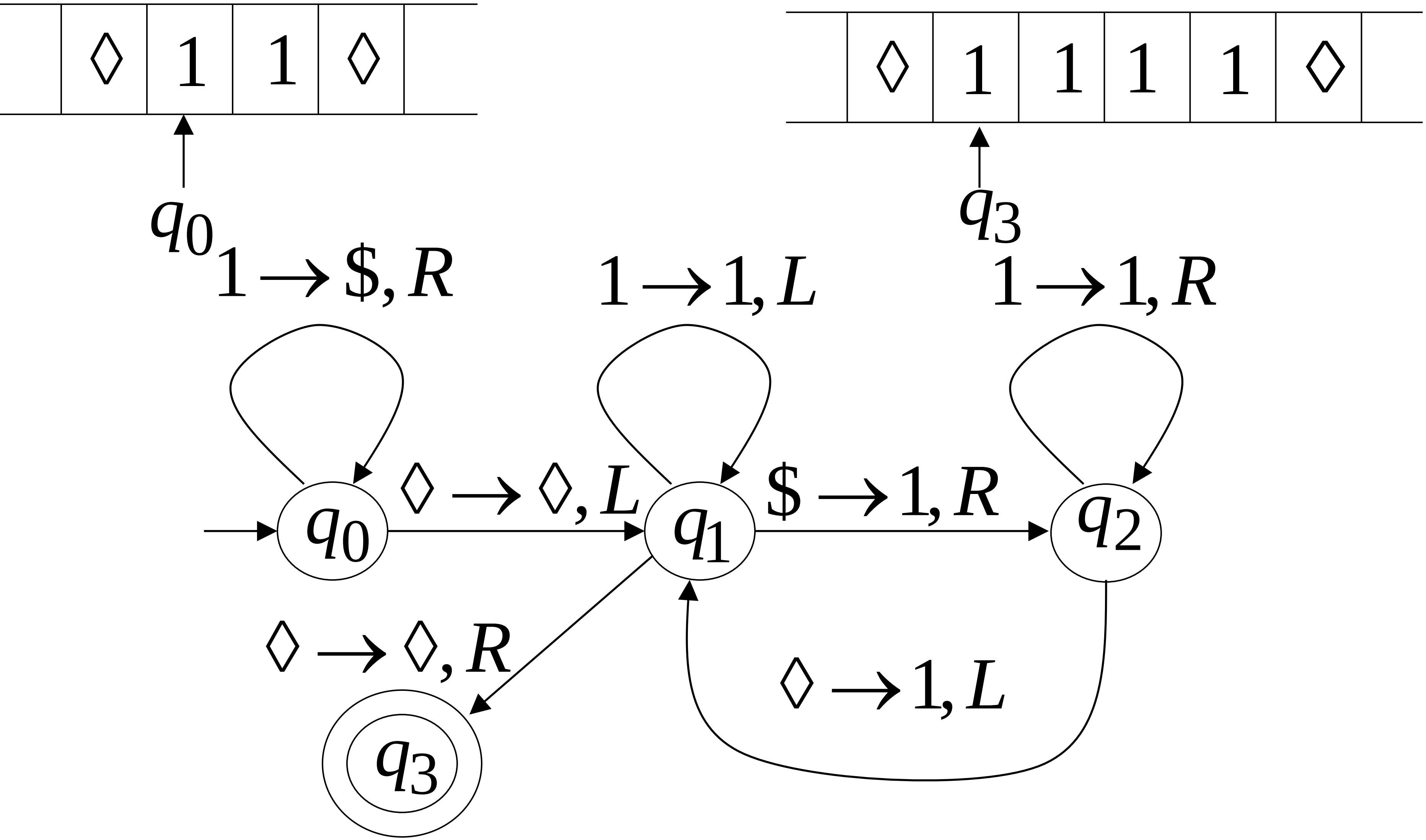
$$f(x) = 2x$$



Start

Example

Finish



Another Example

The function

$$f(x, y) = \begin{cases} 1 & \text{if} \quad x > y \\ 0 & \text{if} \quad x \leq y \end{cases}$$

is computable

Turing Machine for

$$f(x,y) = \begin{cases} 1 & \text{if} \quad x > y \\ 0 & \text{if} \quad x \le y \end{cases}$$

Input: $x0y$

Output: 1 or 0

Turing Machine Pseudocode:

- Repeat

Match a 1 from x with a 1 from y

Until all of x or y is matched

- If a 1 from x is not matched
  erase tape, write 1 $\quad (x > y)$

else
  erase tape, write 0 $\quad (x \leq y)$

Alonzo Church’s Thesis (1936)

“It is believed that there is no functions that can be defined by humans, whose calculation can be described by any well-defined mathematical algorithm that people can be taught to perform, that cannot be computed by TM”

- Unfortunately the Church thesis is not a theorem because the terms (in red) are not part of any math branch and mathematical axioms deals with “people”.

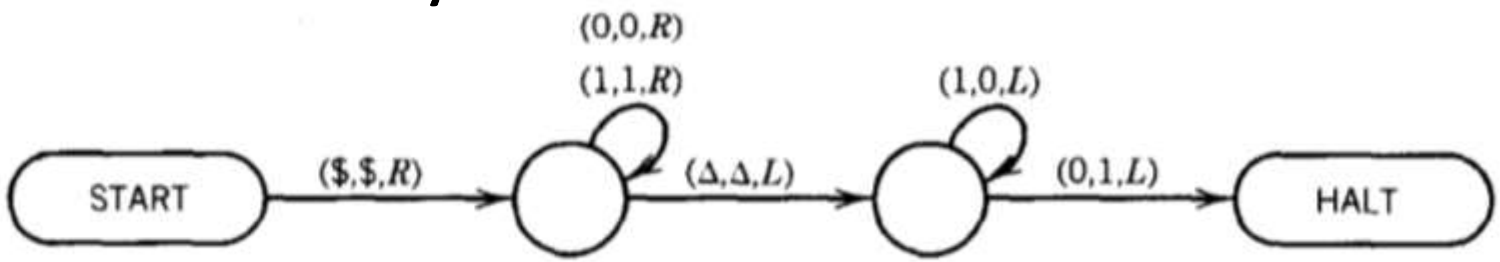
Examples

- ADDER (two binary numbers)
  - Input $1111$1111111Δ
  - Output $$11111111111Δ
- SIMPLE SUBTRACTION (two unary numbers)
  - Input aaaabaaΔ
  - Output aabΔ
- MULTIPLICATION (two unary numbers)
  - Input baaabaa#
  - Output bΔΔΔΔΔΔaaaaaa
- Square Root (two unary numbers)

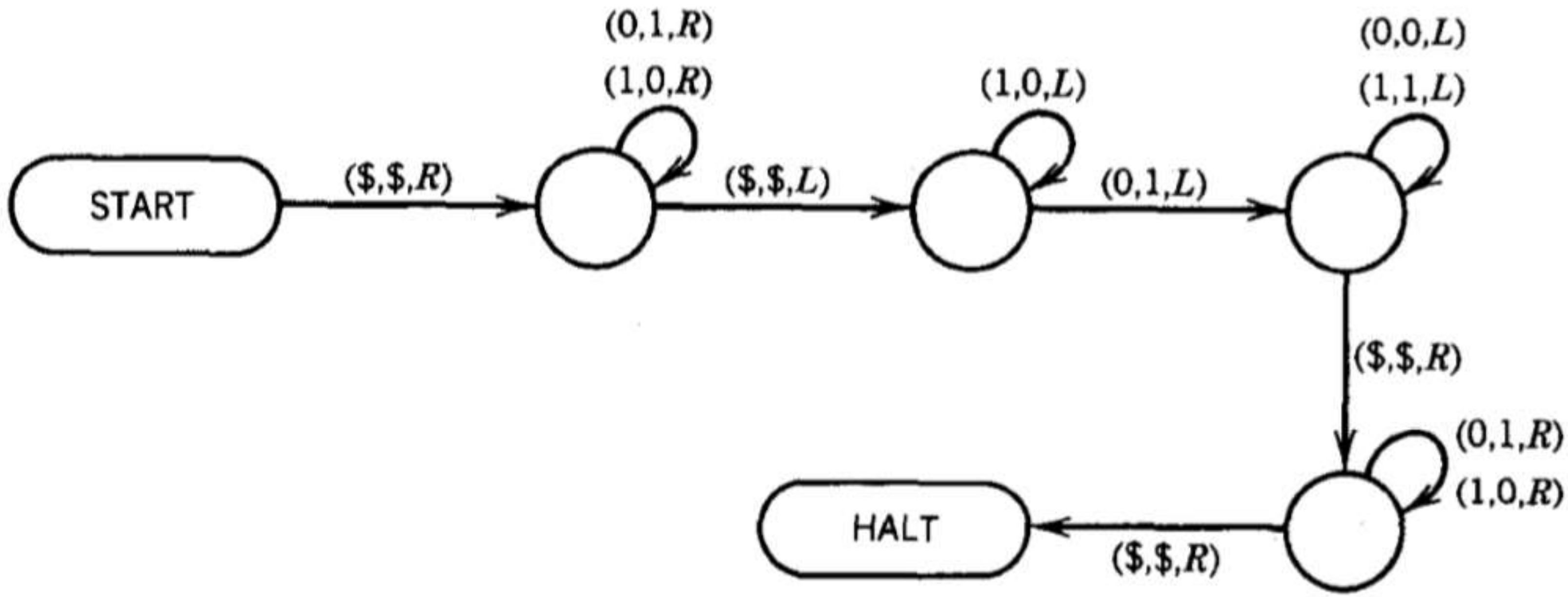
ADDER (adds two binary numbers

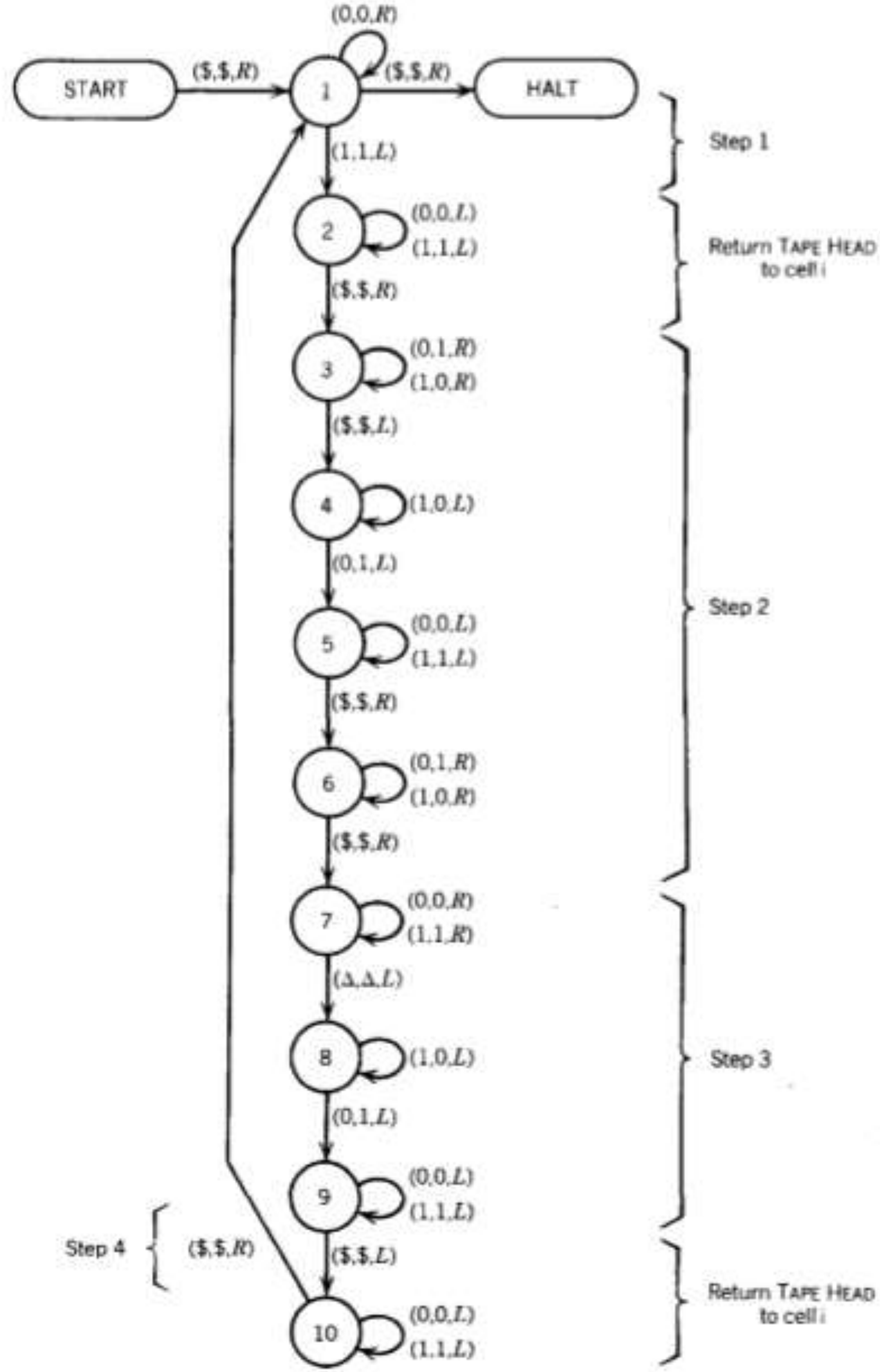
	$ 4 $ 7
becomes	$ 3 $ 8
becomes	$ 2 $ 9
becomes	$ 1 $ 10
becomes	$ 0 $ 11

- Increments by 1



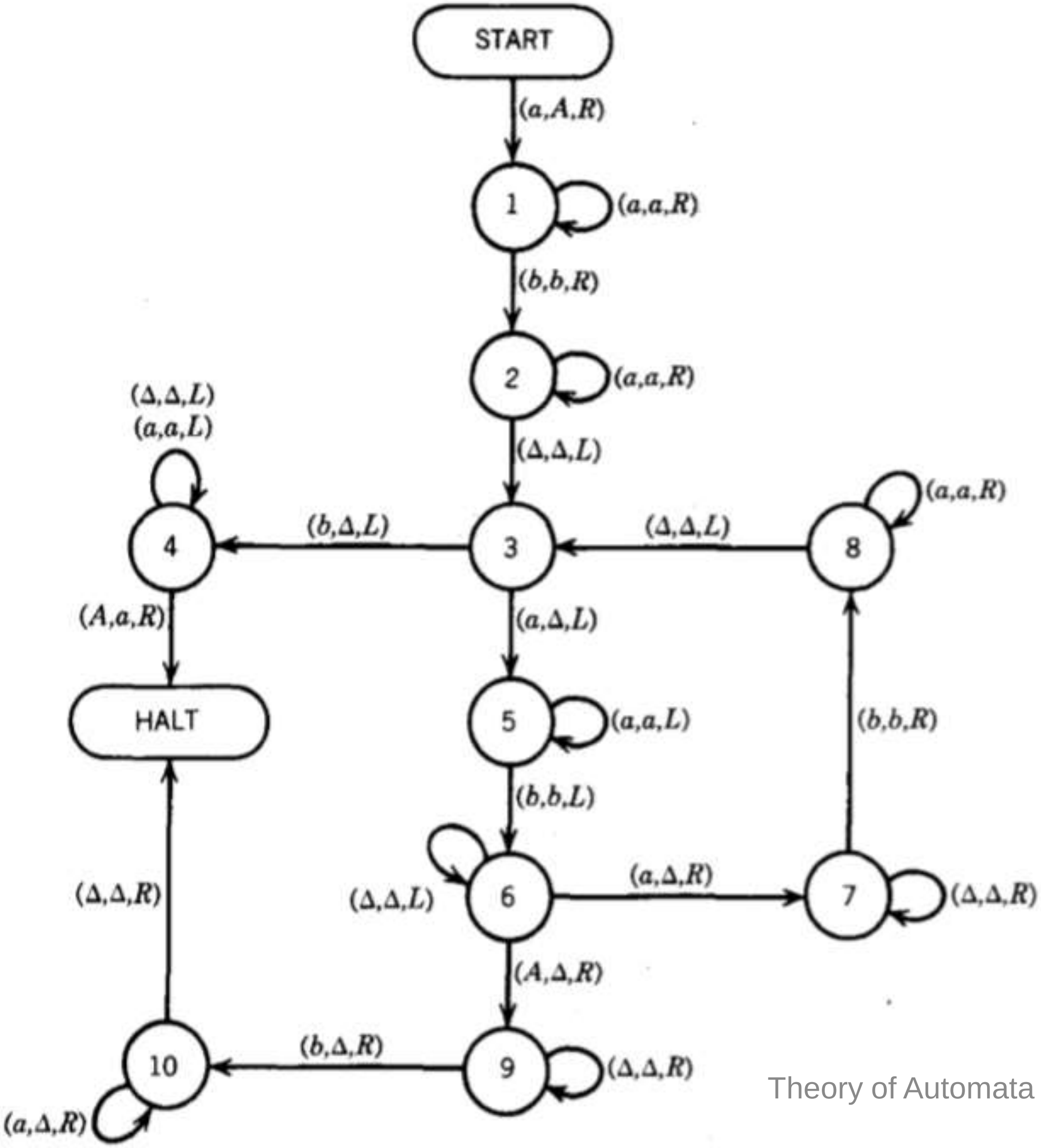
-




START
HALT
1
2
3
4
5
6
7
8
9
10
(0,0,R)
($,$,R)
($,$,R)
(1,1,L)
(0,0,L)
(1,1,L)
($,$,R)
(0,1,R)
(1,0,R)
($,$,L)
(1,0,L)
(0,1,L)
(0,0,L)
(1,1,L)
($,$,R)
(0,1,R)
(1,0,R)
($,$,R)
(0,0,R)
(1,1,R)
(Δ,Δ,L)
(1,0,L)
(0,1,L)
(0,0,L)
(1,1,L)
($,$,L)
(0,0,L)
(1,1,L)
Step 4
($,$,R)
Step 1
Return TAPE HEAD to cell i
Step 2
Step 3
Return TAPE HEAD to cell i

Theory o

SIMPLE SUBTRACTION



$$m \dot{-} n = \begin{cases} m - n & \text{if } m \geqslant n \\ 0 & \text{if } m \leqslant n \end{cases}$$

- SQURE ROOT
  - Input: 10
  - Output:3

  - Input: 20
  - Output: 4