San José State University
Department of Computer Science

Ahmad Yazdankhah
ahmad.yazdankhah@sjsu.edu
www.cs.sjsu.edu/~yazdankhah

Turing Machines

(Part 3)

Lecture 17 - 1

Day 20/31

CS 154

Formal Languages and Computability

Spring 2018

Agenda of Day 20

- Collecting Quiz 6 (Take-Home Exam)
- About Teams
- Summary of Lecture 16
- Lecture 17: Teaching ...
  - Turing Machines (Part 3)
  - Other Models of TMs (Part 1)

Summary of Lecture 16: We learned ...

TMs

- A standard Turing machine M is defined by the septuple:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$$

$$\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$$

- $\square \in \Gamma$ is a special symbol called blank.

- $\delta(q_1, a) = (q_2, b, R)$

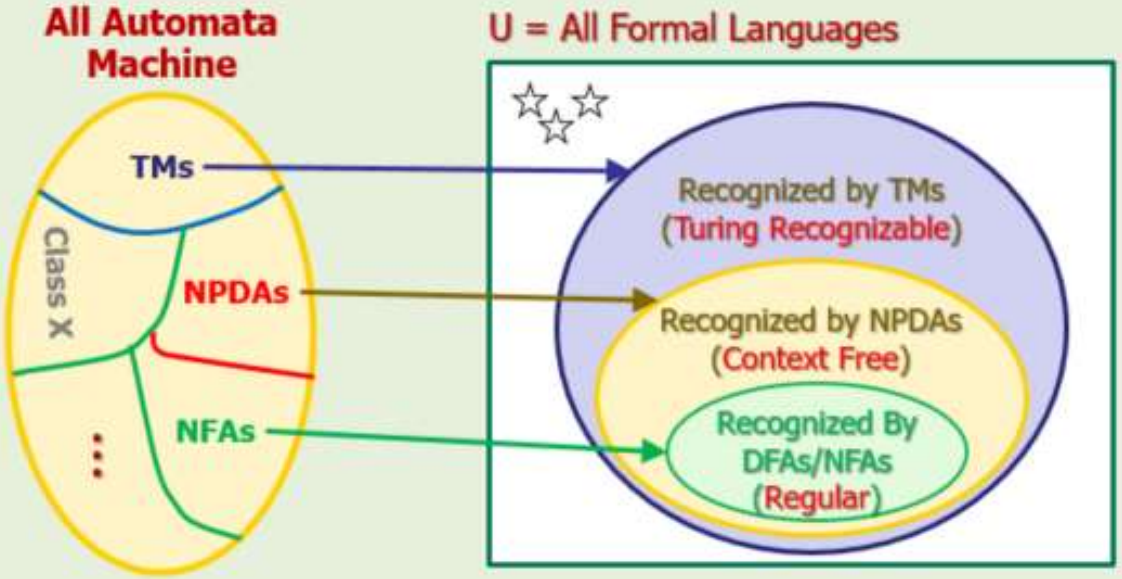
TMs vs NPDAs

- Another way to compare classes of automata is "simulation".
- We can simulate whatever NPDAs do with TMs.
- But not vice versa!
  - At least we know some languages for which we could not construct NPDAs but could construct TMS
  - such as: $a^n b^n c^n$ and ww
- So, TMs are more powerful than NPDAs.

Any Question

Summary of Lecture 16: We learned ...

Machines and Languages Association



- TMs recognize some other non-regular languages called "Turing recognizable".

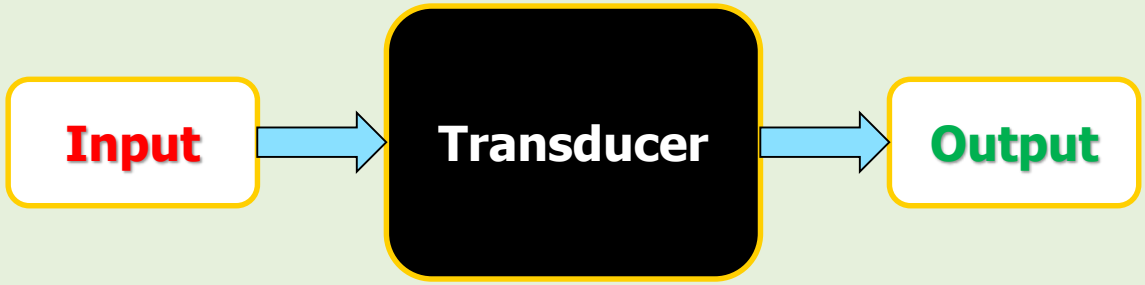
Any Question

TMs as Transducers

What are transducers?

- In physics, transducer is a device that converts the variation in a physical quantity into an electrical signal, or vice versa.
  - The variation could be movement, pressure, brightness, or so forth.

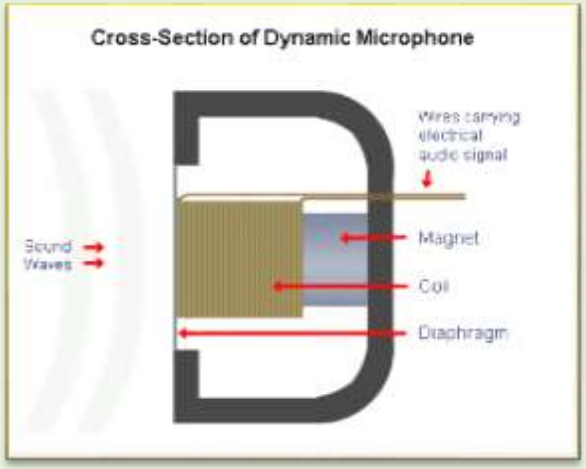
- In a nutshell, transducer is a device that converts an input to an output.



Examples of Transducers

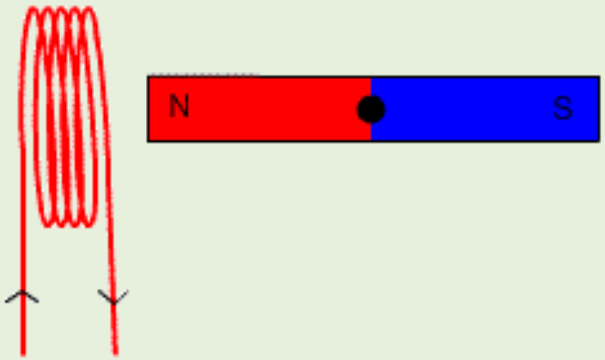
Example 11: Microphone

- A microphone converts your voice to electrical signals.



Example 12: Electric Generator

- An electric generator converts movement to electrical power.



Reference

- Bo Krantz Simonsen at da.wikipedia

Mathematical Model of Transducers

- What is the mathematical model for transducers?
- Functions!

- Recall that a function ...
- ... converts (aka maps) an input (a member of its domain) to an output (a member of its range) based on a rule.

TMs as Transducers

- A TM can act like a transducer. Why?
  - Because it can transform an input to an output.
- What are the input and output of a TM when we run TMs in transducer mode?

Input

- All or part of the nonblank symbols on the tape at the initial time.

Output

- All or part of the tape's content when the machine halts.

- In fact, it's designer's responsibility to define the input and output.

How JFLAP Shows TMs' Outputs

- JFLAP can run TMs in two modes:
  - Regular mode
  - Transducer mode

- In regular mode, JFLAP shows "Accept" or "Reject" for accepting/rejecting strings.
- In transducer mode, it shows the output of the computation too.

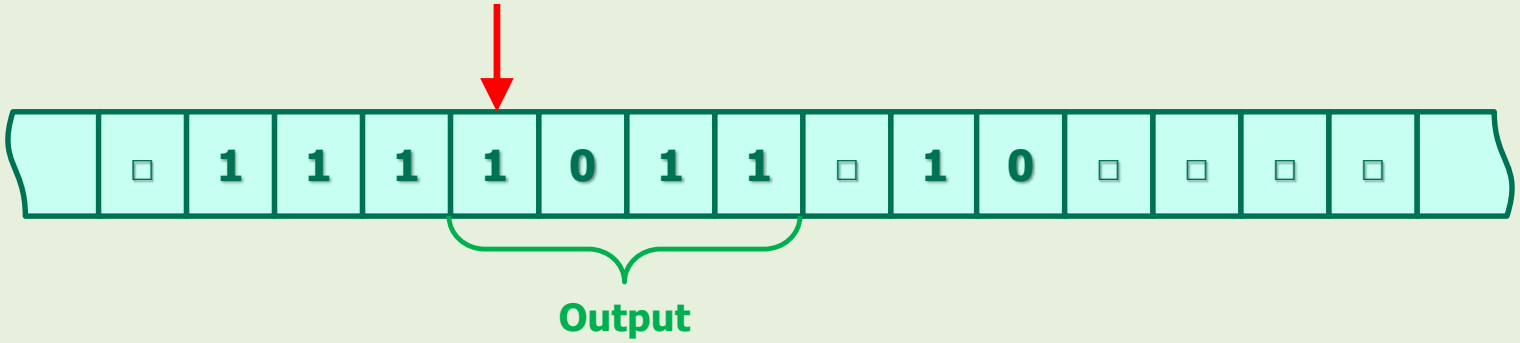
- We'll follow how JFLAP shows the output of TMs when we run them in transducer mode.

- Let's explain it through an example.

How JFLAP Shows TMs Outputs in Transducer Mode

Example 13

- When the TM halts in an accepting state, if the content of the tape is what the following schematic shows , ...



- ... what would be the output?
- Output = 1 0 1 1

- So, the output starts from the symbol at which the head is pointing, to the right, until the first blank.

Turing Computable Functions

Definition

- A function is said to be "Turing-computable" (or just "computable") if there exists a TM that implements it.

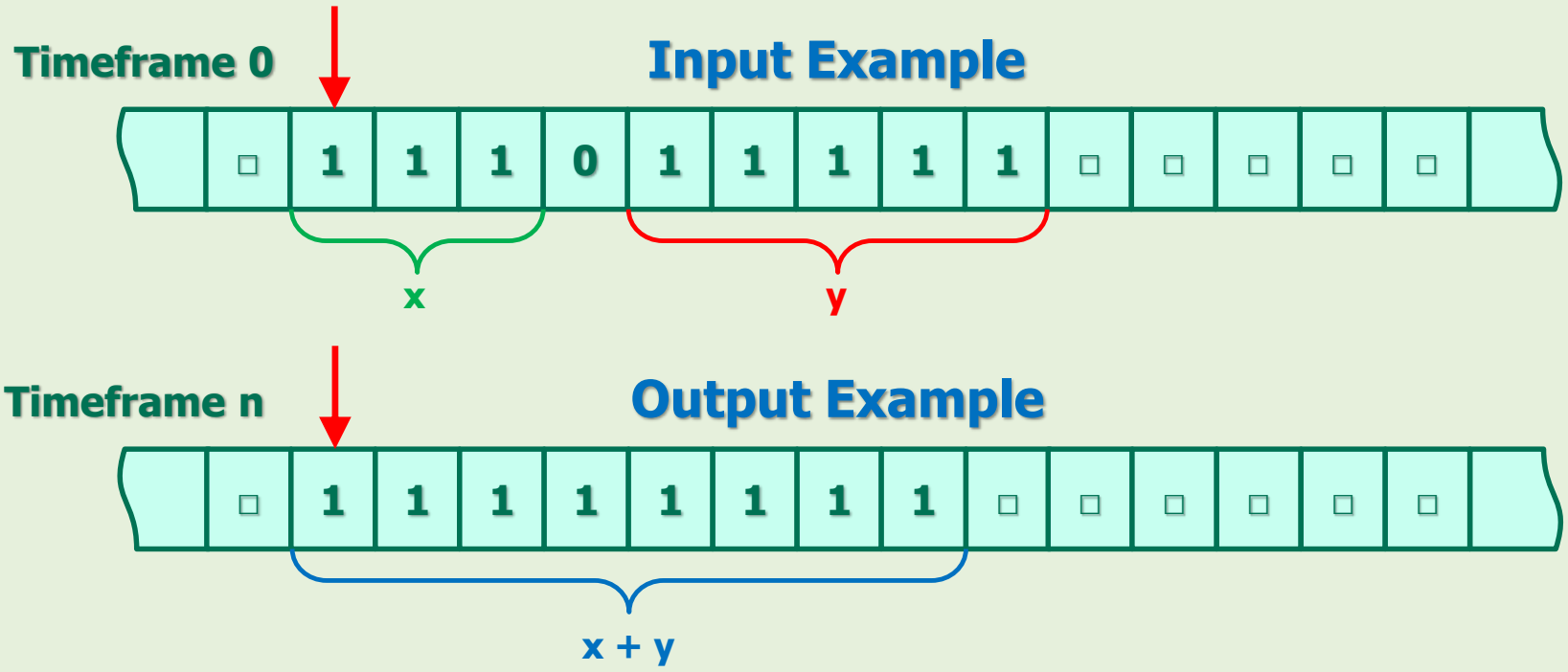
- It has been proved that all common mathematical functions, no matter how complicated, are Turing-computable.

- Let's take some practical examples (basic operations of computers) such as:
  - adding numbers
  - copying strings
  - simple comparisons

TMs as Transducers

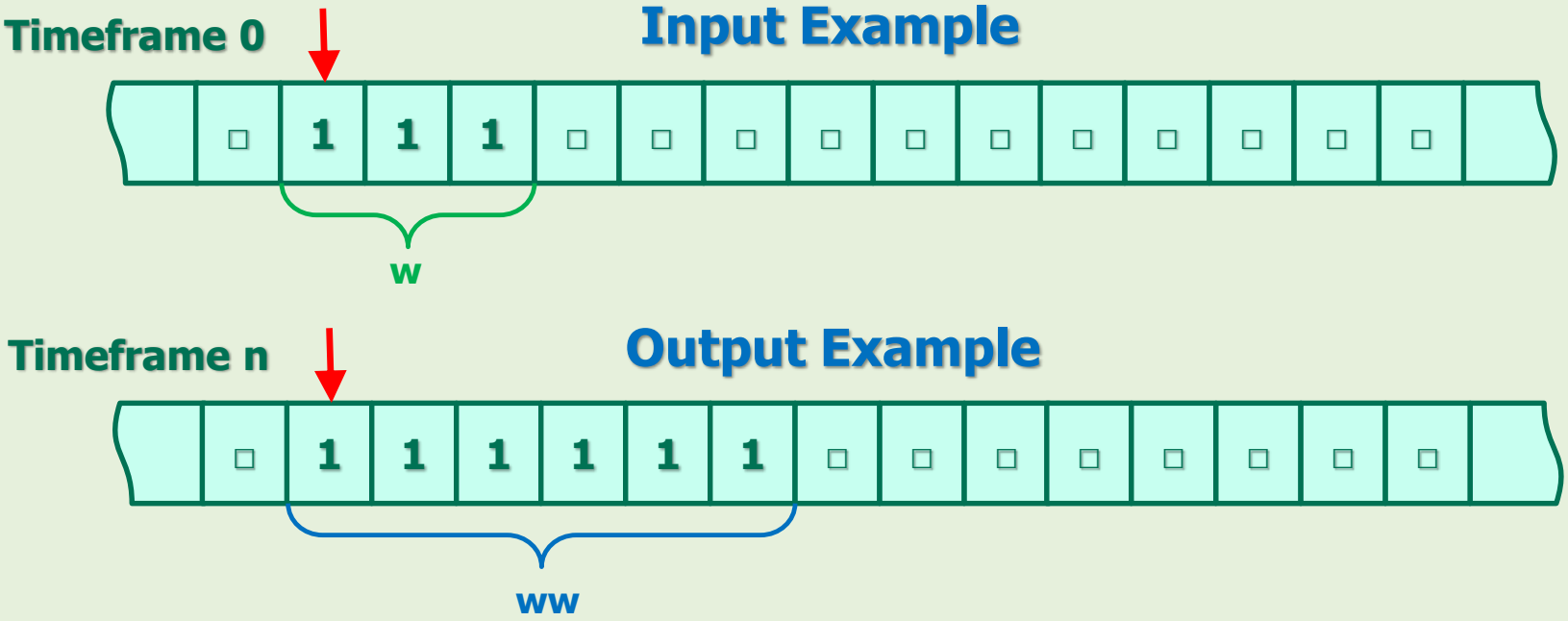
Example 14

- Given two non-zero positive integers x and y, represented in unary notation, separated by a single 0 (zero).
- Design a TM that computes x + y.



Homework

- Given a string $w \in L = \{1\}^+$.
- Design a TM that writes a copy of w at the end of the w.



- At the end of the execution, we'll have "ww" on the tape.

Homework

- Given two non-zero positive integers x and y (x goes first), represented in unary notation separated by a single 0 (zero).

- Design a TM that halts in an accepting state if $x \geq y$, otherwise it halts in a non-accepting state.

Combining TMs

Combining TMs

- How to combine TMs to solve more complex problems?

Example 15

- Given two non-zero positive integers x and y, represented in unary notation, and separated by a zero.
- Design a TM to compute the following function:

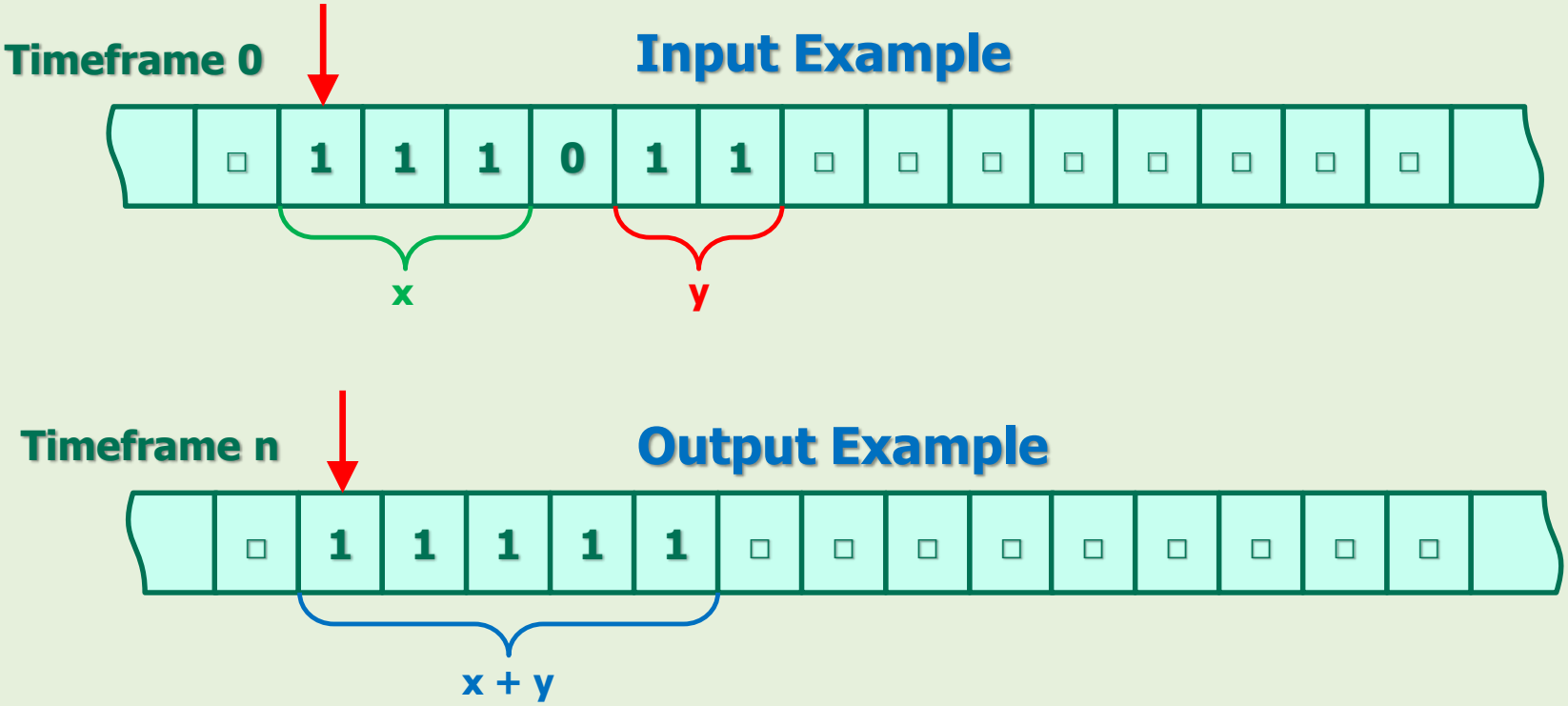
$$f(x, y) = \begin{cases} x + y & \text{if } x \geq y \\ 0 & \text{if } x < y \end{cases}$$

- Let's visualize what would be the output in different situations ...

TM as Transducer

Example 15: When x ≥ y

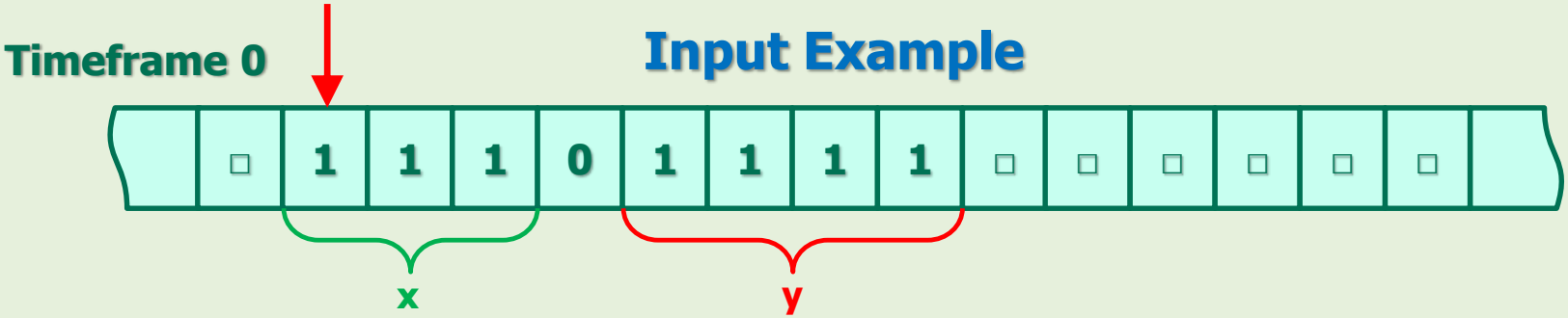
$$f(x, y) = \begin{cases} x + y & \text{if } x \geq y \\ 0 & \text{if } x < y \end{cases}$$

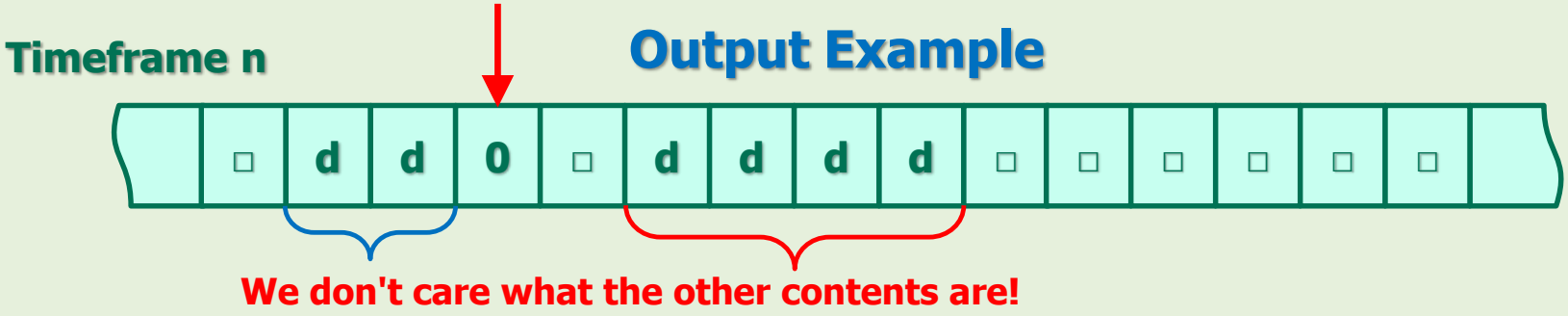


TM as Transducer

Example 15: When x < y

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

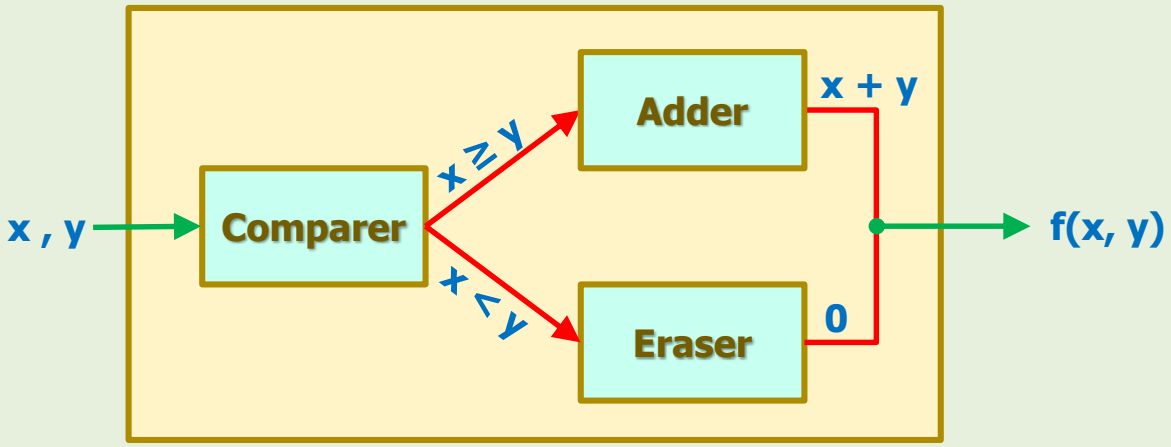




Combining TMs

Example 15 (cont'd)

- To implement this function, we would need the following routines:
  - a comparer
  - an adder
  - an eraser
- A comparer compares x and y ...
  - If $x \geq y$, then it activates the adder to compute x + y.
  - Else (i.e.: $x < y$), it activates the eraser to show zero.



Combining TMs

Example 15 (cont'd)

- This example shows that we can decompose any complex problem into simpler routines and implement each routine with a TM.

- Do the rest of this example as homework!

Homework

- Given two positive integers x and y, represented in unary notation, and separated by a zero.
- Design a TM to compute x * y.

San José State University
Department of Computer Science

Ahmad Yazdankhah
ahmad.yazdankhah@sjsu.edu
www.cs.sjsu.edu/~yazdankhah

Other Models of TMs

Lecture 17 -2
Day 20/31

CS 154
Formal Languages and Computability
Spring 2018

Introduction

- We saw how adding memory could increase the computing power.
- Now the question is ...
- ... to make more powerful machines, can we add more memory, or different kind of memories?
- For example:
  - two or more stacks,
  - a queue and a stack,
  - two or more tapes
  - or ...
- How far can we go?
- What would be the most powerful automata?

Objective of This Lecture

- We are going to check some of those questions.
- We'll add some extra capabilities to the standard TMs.

- By any changes, we introduce a new class of automata.
- Then, the very first question would be:
  - Is this new class more powerful than the standard TMs?

- We've already saw how we compare the classes of automata by simulation.

TMs with Stay-Option

TMs with Stay-Option

- In standard TMs, the head can move "left (L)" or "right (R)".
- Now, we add a new movement to the head, ...
  - Stay (S), or no move

- Everything else would remain exactly the same as standard TMs.

Formal Definition

$$M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$$

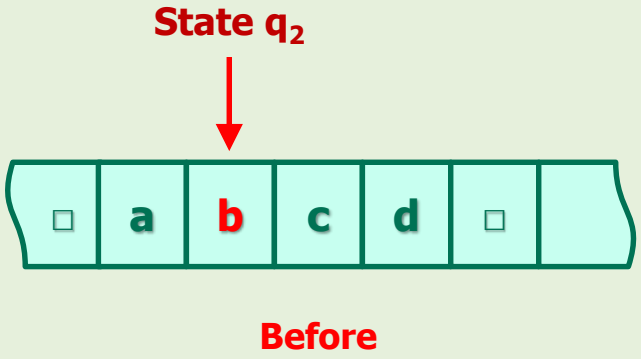
- Where:
  - ... (same as standard TM)
  - $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, S\}$

TMs with Stay-Option

Example 1

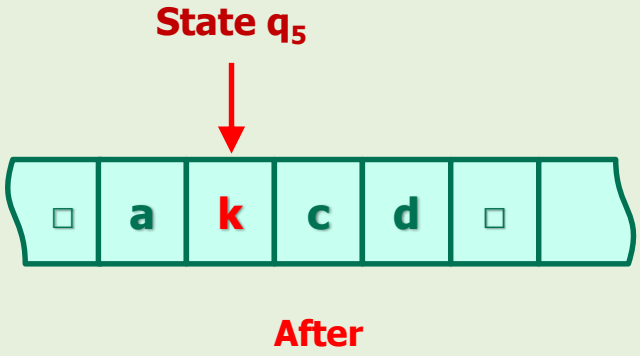
- Consider the following transition:

$\delta(q_2, b) = (q_5, k, S)$



Before

- The figures show the content of the tape and the position of the read-write-head before and after the transition.



After

Is this new class more powerful than standard TMs?

Theorem

- The class of TMs with stay-option is equivalent to the class of standard TMs.
- We need to prove two things:
  1. TMs with stay-option simulate standard TMs.
  2. Standard TMs simulate TMs with stay-option.

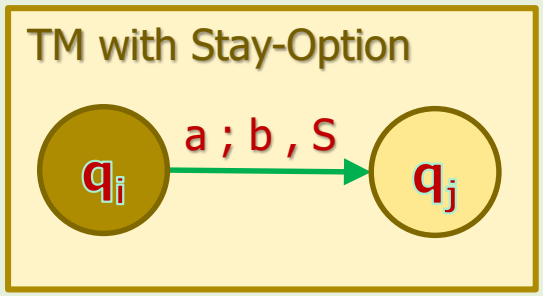
Proof of 1

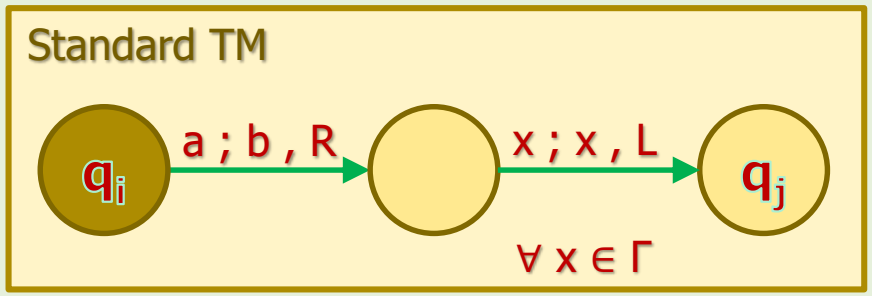
- Let's assume we've constructed a standard TM for an arbitrary language L.
- Can we always construct a TM with stay-option for L? How?
- Yes, just don't use "S" option!

Is this new class more powerful than standard TMs?

Proof of 2

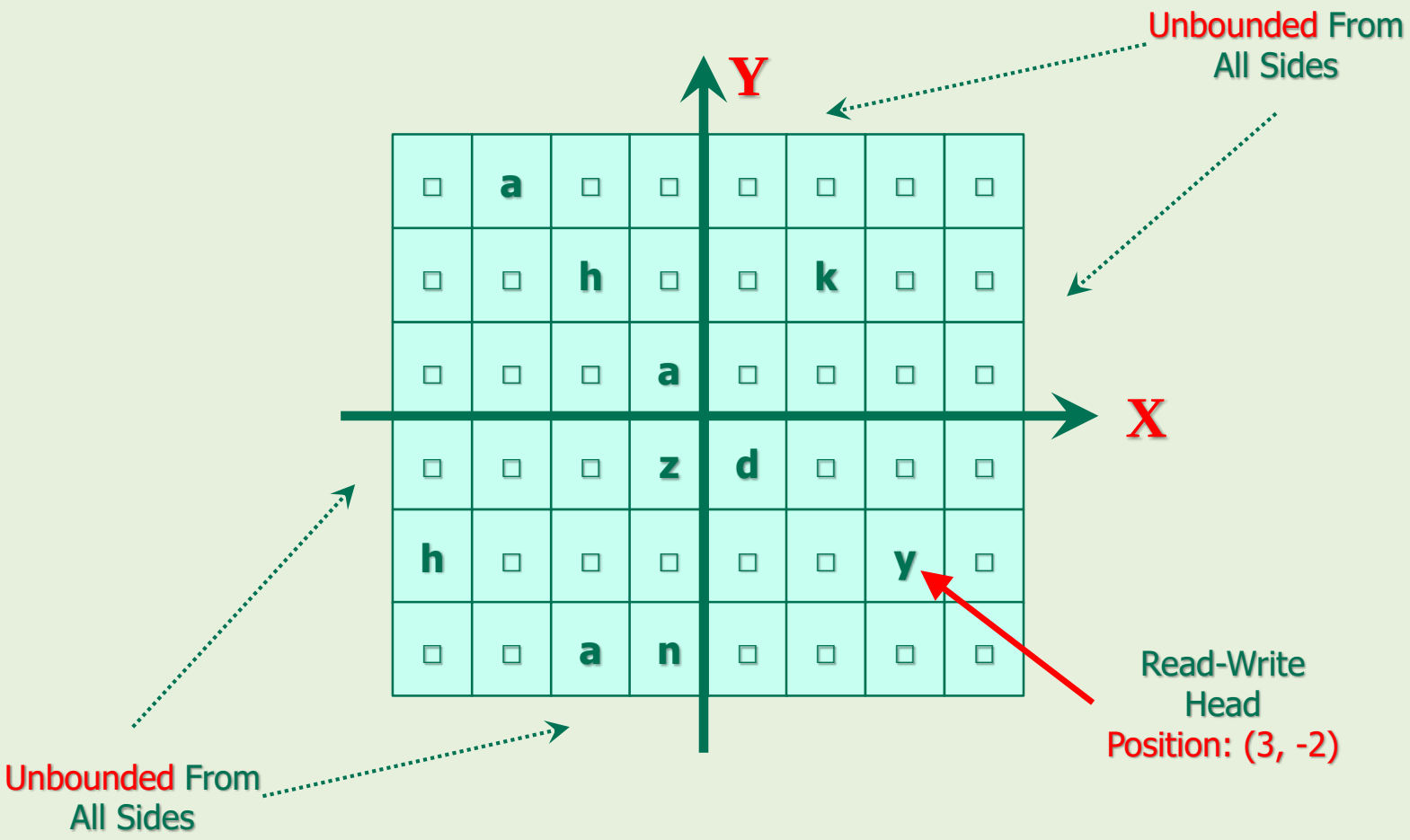
- Let's assume we've constructed a TM with stay-option for an arbitrary language L.
- Can we construct a standard TM for L? How?
- The only difference is those transitions that use "S" option.
- So, we just need to simulate "no move" for those transitions.
- That can be simulated by two moves of the head: "left, then right" (or "right, then left") as the following figures show.





Multidimensional-Tape TMs

Multidimensional-Tape TMs



Multidimensional-Tape TMs

Definition

- A multidimensional-tape TMs M is defined by the septuple:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$$

- Where:
  - ... (same as standard TM elements)

$$\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, U, D\}$$

- The new movements: U=Up and D=Down

Is this new class more powerful than standard TMs?

Theorem

- The class of multidimensional-tape TMs is equivalent to the class of standard TMs.
- We need to prove two things:
  1. Multidimensional-tape TMs simulate standard TMs.
  2. Standard TMs simulate multidimensional-tape TMs.

Proof

1. Multidimensional-tape TMs simulate standard TMs.

- This step is trivial because if we just use one row of the tape, then we have standard TM.

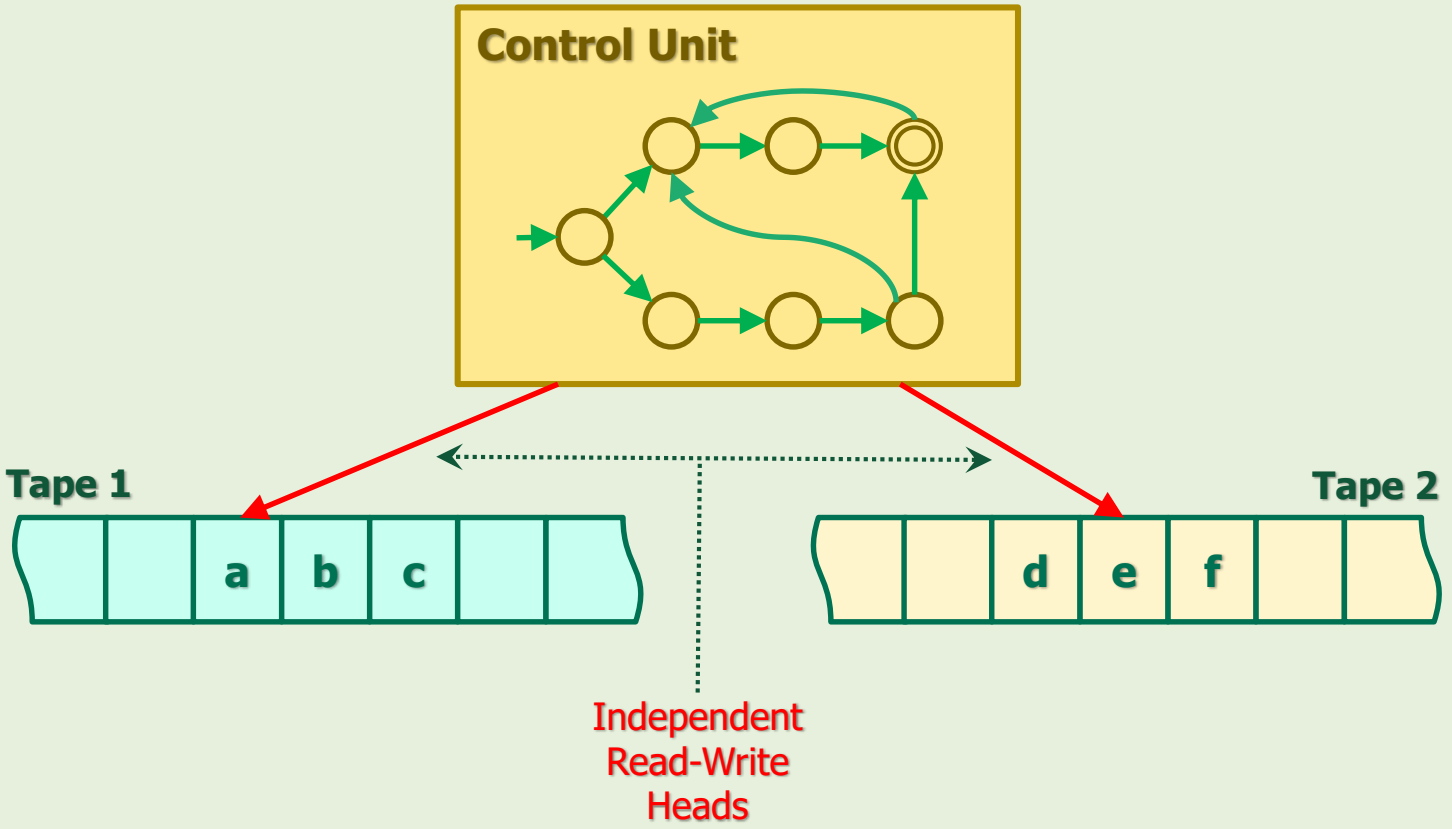
2. Standard TMs simulate multidimensional-tape TMs.

- Prove this as exercise!

Multi-Tape TMs

Multi-Tape TMs

- We can add additional tapes with independent read-write head to the standard TMs.
- For example, a double-tape TM looks like this:



Multi-Tape TMs

Formal Definition

- An n-tape TM M is defined by the septuple:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$$

- Where:
  - ... (same as standard TM elements)

$$\delta: Q \times \Gamma^n \rightarrow Q \times \Gamma^n \times \{L, R\}^n$$

- Note that: $\Gamma^n = \Gamma \times \Gamma \times \ldots \times \Gamma$ (Cartesian product)
- Therefore, the result is an n-tuple.

Formal Definition of Multi-Tape TMs

- **Example 2**

- This is a transition of a double-tape TM.
  - We separate the labels of different tapes with "|".
- The transition condition is both inputs:

  input symbol of tape 1 = 'a'

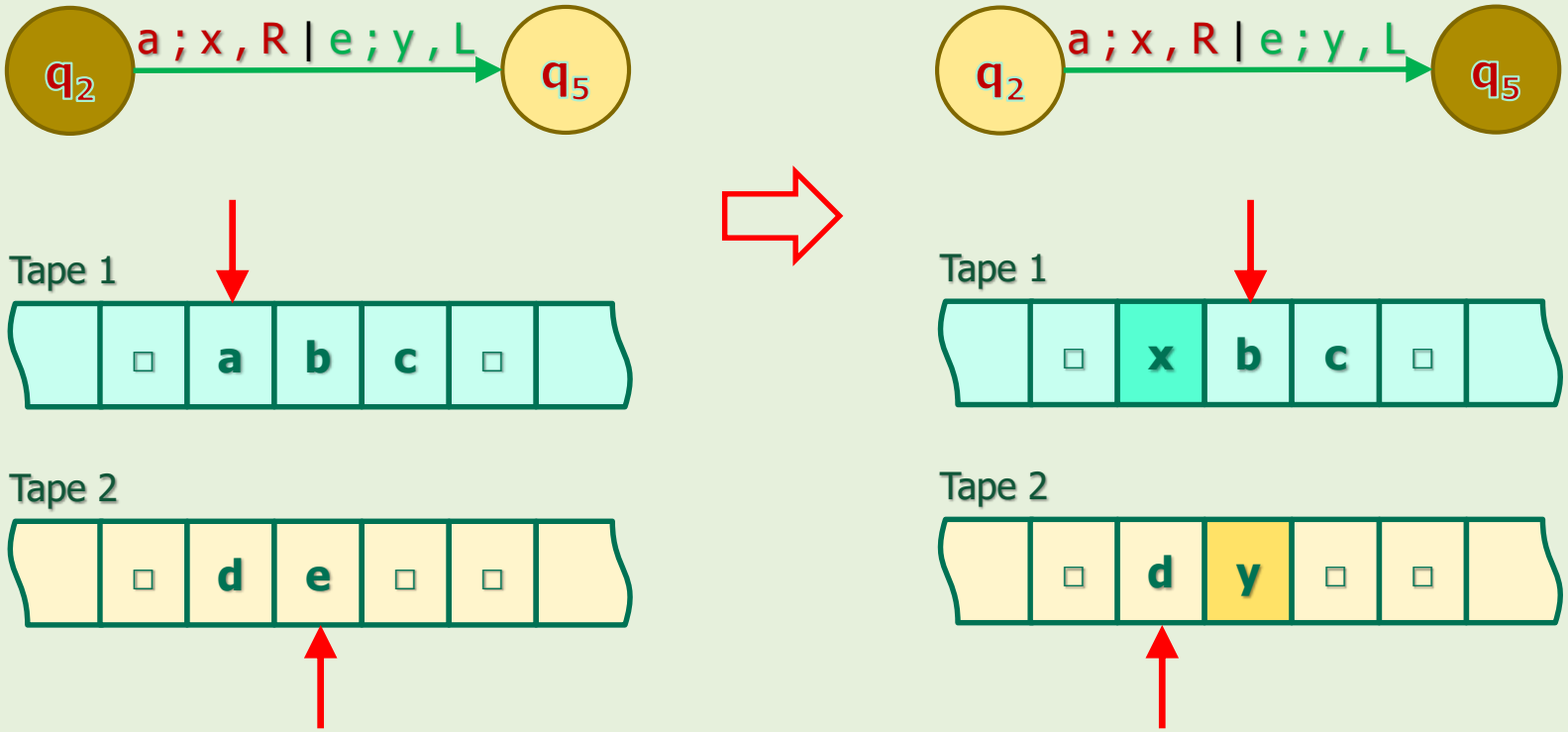
  AND

  input symbol of tape 2 = 'e'.

- The sub-rule looks like this: $\delta(q_2, a, e) = (q_5, x, y, R, L)$

Formal Definition of Multi-Tape TMs

- **Example 2 (cont'd)** $\delta(q_2, a, e) = (q_5, x, y, R, L)$

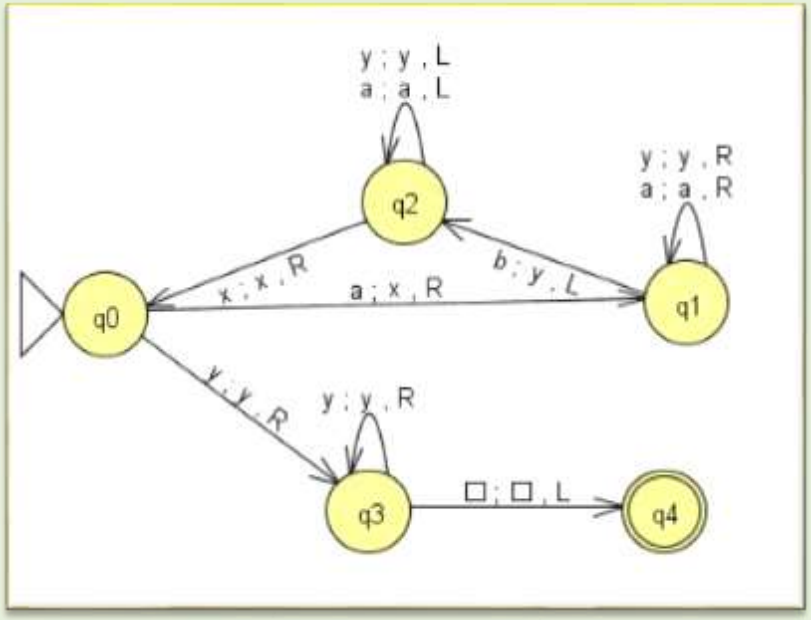


Before **After**

Multi-Tape TMs Example

Example 3

- Design a double-tape TM for accepting the language $L = \{a^n b^n : n \geq 1\}$ over $\Sigma = \{a, b\}$.
- Assume that the input is written on the tape 1 and for simplicity, we can use "stay-option".
- Recall that we had designed a standard TM for L before as the figure shows.



Multi-Tape TMs Example

Example 3 (cont'd)

Strategy

- Read a's from tape 1 and write them on tape 2.
- When sensed the first 'b' on tape 1, match b's with the a's on tape 2.
- If all match, then accept,
- otherwise, reject.

- Do double-tape TMs facilitate our programming?

Homework

- Design a double-tape TM for accepting the following language:

  $L = \{a^n b^n c^n : n \geq 1\}$ over $\Sigma = \{a, b, c\}$

- Assume that the input is written on the tape 1 and for simplicity, you can use "stay-option".

Is this new class more powerful than standard TMs?

Theorem

- The class of multi-tape TMs is equivalent to the class of standard TMs.
- We need to prove two things:
  1. Multi-tape TMs simulate standard TMs.
  2. Standard TMs simulate multi-tape TMs.

Proof

1. Multi-tape TMs simulate standard TMs.
- This step is trivial because if we just use one tape, then we have standard TM.

2. Standard TMs simulate multi-tape TMs.

Nice Videos

1. Turing machines explained visually
   https://www.youtube.com/watch?v=-ZS_zFg4w5k
2. A Turing machine – Overview
   https://www.youtube.com/watch?v=E3keLeMwfHY

References

1. Linz, Peter, "An Introduction to Formal Languages and Automata, 5th ed.," Jones & Bartlett Learning, LLC, Canada, 2012
2. Kenneth H. Rosen, "Discrete Mathematics and Its Applications, 7th ed.," McGraw Hill, New York, United States, 2012
3. Michael Sipser, "Introduction to the Theory of Computation, 3rd ed.," CENGAGE Learning, United States, 2013
   ISBN-13: 978-1133187790
4. Wikimedia Commons, https://commons.wikimedia.org/wiki/Category:Animations_of_machinery
5. https://en.wikipedia.org/wiki/Turing_Award
6. https://en.wikipedia.org/wiki/Alan_Turing