San José State University
Department of Computer Science

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

Other Models of TMs

(Part 2)

Lecture 18

Day 19/31

CS 154

Formal Languages and Computability

Fall 2019

Agenda of Day 19

- Demo: One of the Previous Semesters Term Project
- About Midterm 2
- Summary of Lecture 17
- Quiz 7
- Lecture 18: Teaching ...
  - Other Models of TMs (Part 2)

Demo

One of the Previous Semesters Term Project

About Midterm 2

Reminder 1

- Midterm #2 (aka Quiz++)
  - Date: Thursday 10/31
  - Value: 15%
  - Topics: Everything covered from the beginning of the semester
  - Type: Closed y ∈ Material

    Material = {Book, Notes, Electronic Devices, Chat, ... }

- The cutoff for this midterm is the end of lecture 18.

Study Guide

- I'll announce the type and number of questions via Canvas.

Summary of Lecture 17: We learned ...

TMs as Transducer

- Transducer is a device that converts an "input" to an "output".
- We model a transducer by a ...
  - ... function.
- TMs can work in transducers mode.
- **Input** is all or part of the nonblank symbols on the tape at the initial time.
- **Output** is all or part of the tape's content when the machine halts.
- It's designer's responsibility to define the input and output.

- We learned how JFLAP shows the output.
- Note that if the TM does not halt in an accepting-state, JFLAP does not show the output.
- A function is called Turing-computable if ...
  - ... there exists a TM that implements it.
- We learned how to break a complex problem into smaller ones and how to combine TMs to make a bigger one.

Any Question

Summary of Lecture 17: We learned ...

Other Models of TMs

- We tried to figure out whether we can get more power by adding some capabilities to standard TMs.
- With any changes in standard TMs, we created a new class of automata.
- The changes we made:
  - TMs with stay-option ...
  - TMs with multidimensional-tape ...
- To use this option in JFLAP, you'd need to activate it in the JFLAP preferences.

- Were the new classes more powerful than the standard TM?
- We mentioned two theorems stating that the new classes were equivalent to standard TMs.

Any Question?

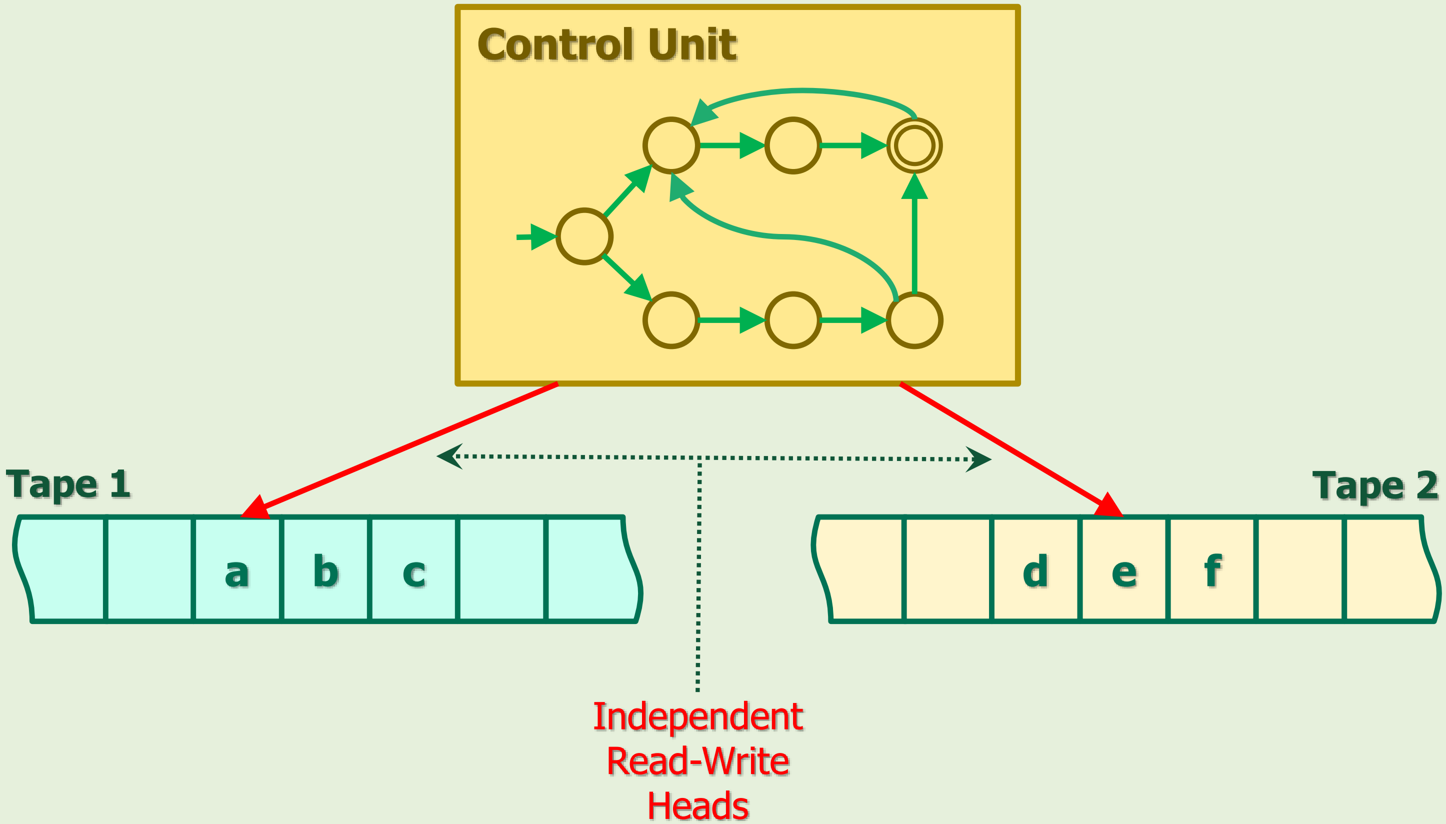
Quiz 7

No Scantron

Multi-Tape TMs

Multi-Tape TMs: Building Block

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



Multi-Tape TMs: Transitions

- **Example 5**

- 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'

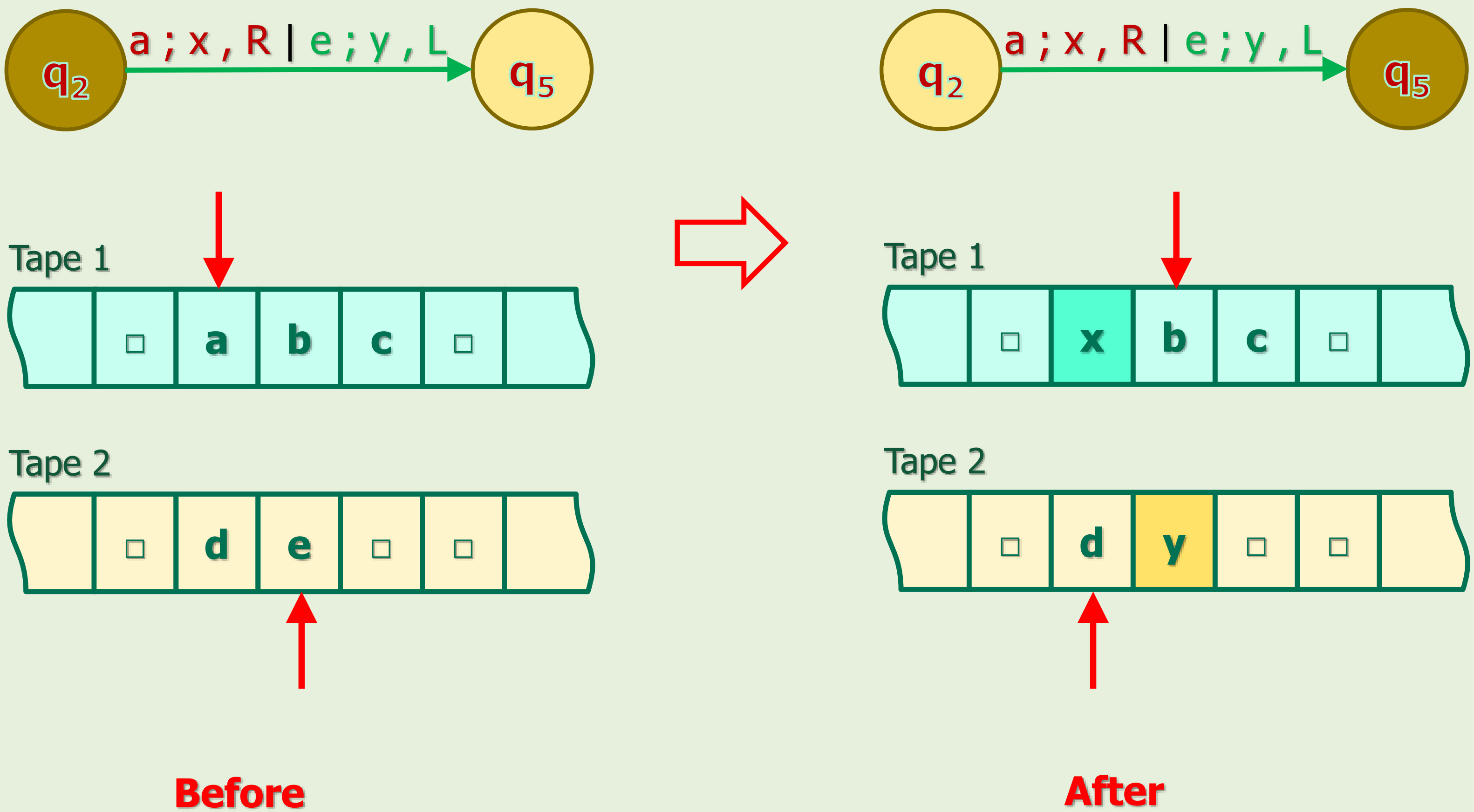
AND

input symbol of tape 2 = 'e'.

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

Multi-Tape TMs: Transitions

- **Example 5 (cont'd)** $\delta(q_2, a, e) = (q_5, x, y, R, L)$



Multi-Tape TMs: Example

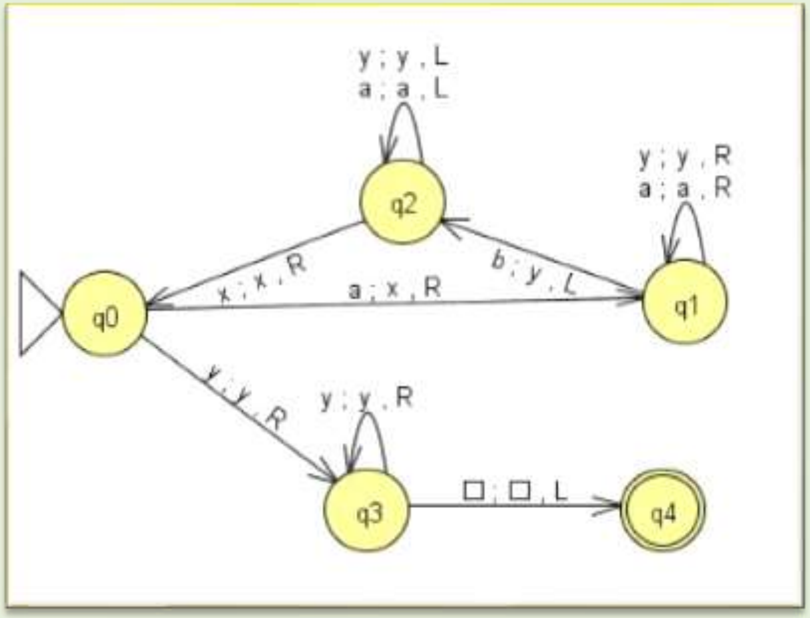
Example 6

- Design a double-tape TM for accepting the language $L = \{a^n b^n : n \geq 1\}$ over $\Sigma = \{a, b\}$.

Requirements:

1. The input is written on the tape 1.
2. You can use stay-option.

- Recall that we designed a standard TM for L before as the figure shows.



Multi-Tape TMs: Example

Example 6 (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 $\Sigma = \{a, b, c\}$

Requirements:

1. The input is written on the tape 1.
2. You can use stay-option.

Multi-Tape TMs: Formal Definition

- A TM with n-tape M is defined by the septuple:

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

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

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

Where $\Gamma^n = \Gamma \times \Gamma \times \ldots \times \Gamma$ (Cartesian product)

$$\{L, R\}^n = \{L, R\} \times \{L, R\} \times \ldots \times \{L, R\}$$

- Recall that we defined δ as the program of TMS.
- Therefore, the syntax of the code would be the format of the sub-rules.
- For example, one line code of this class looks like this:

$$\delta\,(q_2, a, e) = (q_5, x, y, R, L)$$

Is this new class more powerful than standard TMs?

Theorem

- The TMs with multi-tape class is equivalent to the standard TMs class.
- 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.

Nondeterministic TMs (NTMs)

Nondeterministic TMs (NTMs)

Determinism:
During any timeframe, there is no more than one transition.

Any violation of this makes a machine nondeterministic.

Recap

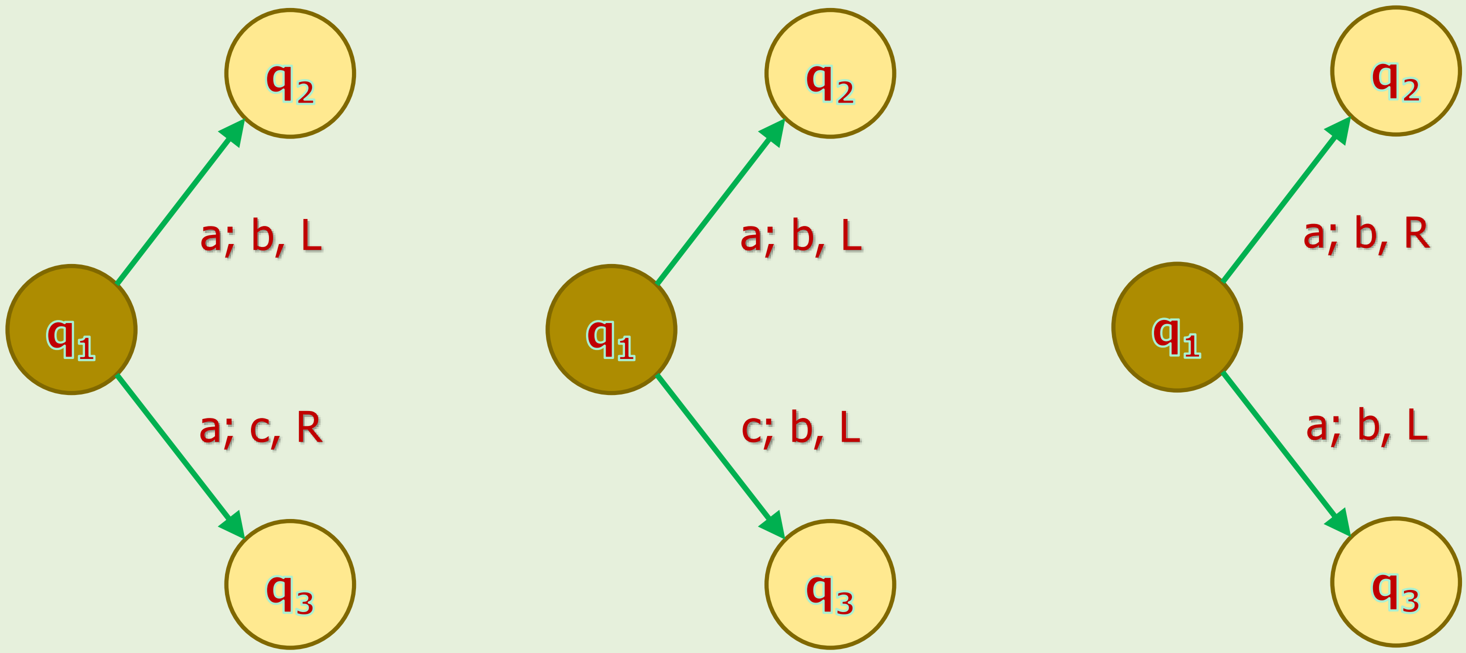
- What could be those violations in standard TMs?
  - ~~λ-transition~~
  - When δ is multifunction

- Theoretically, we can define λ-transition as usual.
- But historically it was not defined for TMs!

NTMs: Multifunction Examples

Example 7

- Are the following transitions violations for determinism?



Yes No Yes

NTMs: Formal Definition

- A nondeterministic TM M is defined by the septuple:

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

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

$$\delta: Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L, R\}}$$

δ is total function.

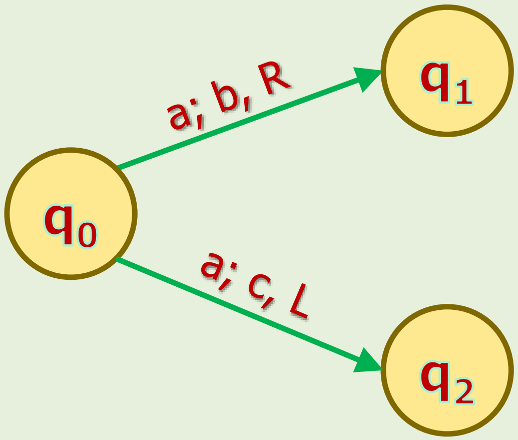
NTMs: Sub-Rules of Transition Function

Example 8

- Draw the transition graph of the following sub-rule:

$$\delta\ (q_0, a) = \{(q_1, b, R), (q_2, c, L)\}$$

Solution



How NTMs Behave If They Have Multiple Choices

- We already know that:

All types of nondeterministic machines start
parallel processing when they have multiple choices.

- In other words, for every possible choice,
they create a new process and every process independently continues processing the string.

- The procedure of initiating a new process is exactly the same as NFAs.

How NTMs Behave If They Have Multiple Choices

Procedure of Initiating New Processes

1. It replicates its entire structure (transition graph + tape)
2. It initializes the new process with the current configuration.
3. The new process independently continues processing the rest of the input string.

- The only thing we need to know is:

  What info do we need for the configuration?

NTMs' Configuration

1. Current state of the transition graph
2. Tape content + Position of the cursor

NTMs vs Standard TMs

NTMs vs Standard TMs

Theorem

- Nondeterministic TMs class is equivalent to standard TMs class.
- We need to prove two things:
  1. Nondeterministic TMs simulate standard TMs.
  2. Standard TMs simulate nondeterministic TMs.

Proof of 1

- Let's assume we've constructed a standard TM for an arbitrary language L.
- Can we always construct a NTM for L? How?
- Yes, just convert TMs definition to the NTMs',
  the same way we did for converting DFAs' to NFAs'.

NTMs vs Standard TMs

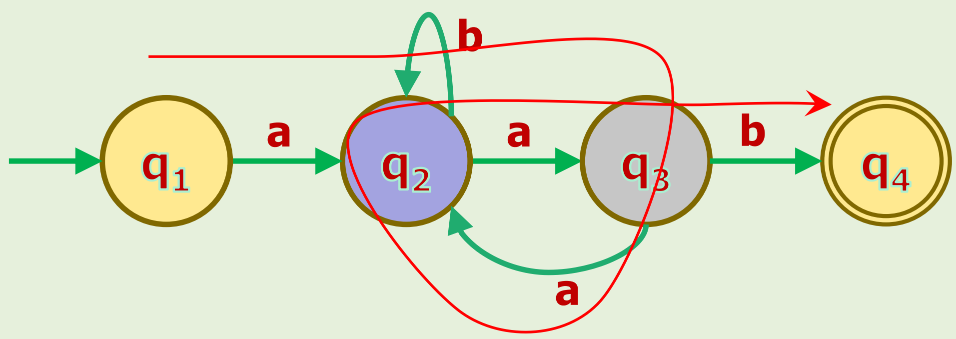
Proof of 2

- Mathematical proof of this part is not so easy but we can understand it intuitively.
- We'll explain it through an example.
- But first, we need to refresh our mind about one-dimensional projection.

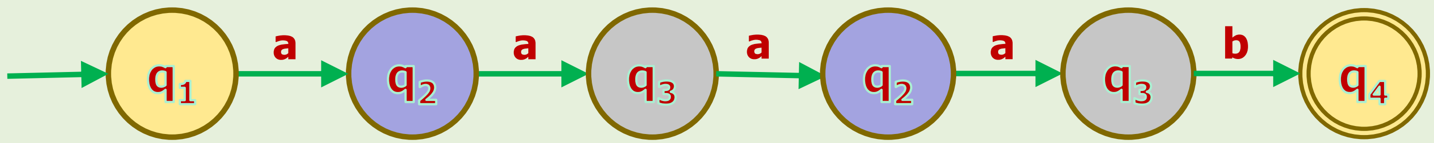
One-Dimensional Projection of a Walk

Recap

- As we learned before, we can represent a walk by one-dimensional projection.
- As an example, look at the string (walk) w = aaaab in the following NFA:

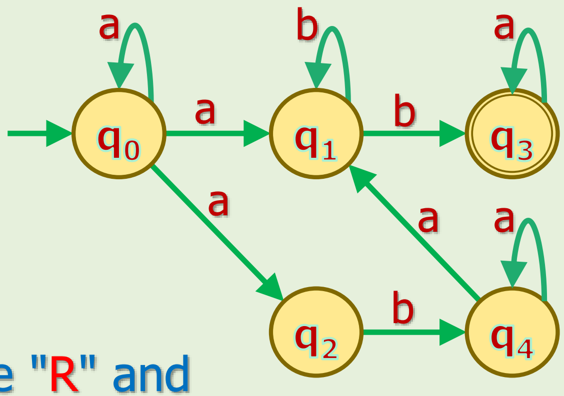


- This walk can be shown as:



NTMs vs Standard TMs

- **Proof of 2 (cont'd)**
- The following transition graph is an example of an NTM.

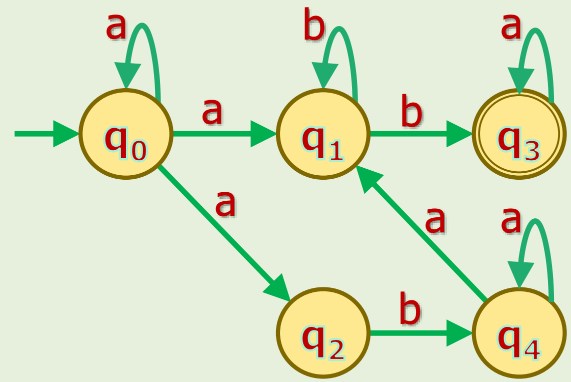


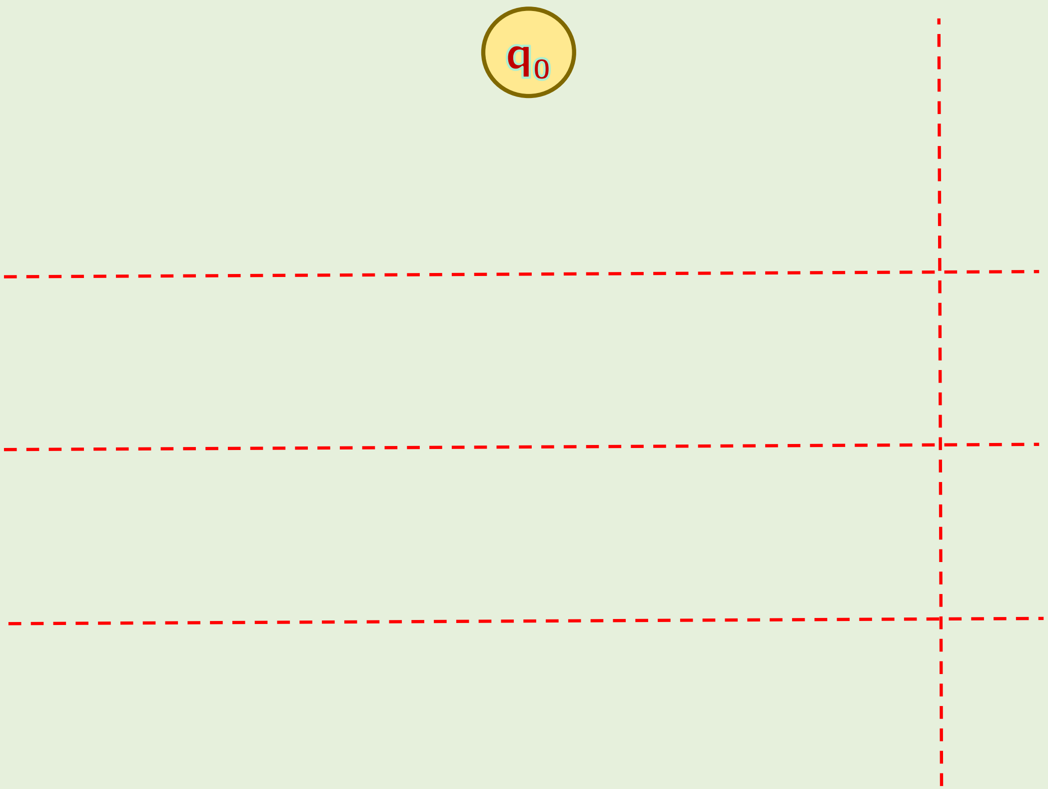
  - For simplicity, assume that all move-symbols are "R" and we don't change the symbols of the tape.
  - Therefore, 'a' in the transition graph is equivalent "a; a, R".
  - Note that we won't lose the generality of the point.

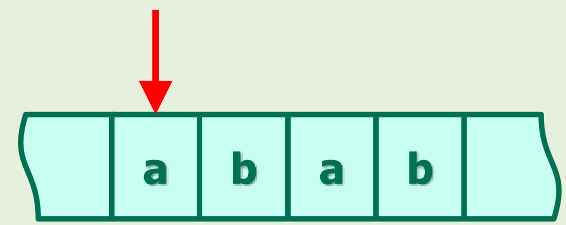
- If we input w = abab into this NTM, overall 6 processes will be initiated.
- We usually prefer to organize them as a tree (aka process-tree).

Nondeterministic TMs vs Standard TMs

- **Proof of 2 (cont'd)**
- All processes for the string w = abab are organized in the following processes tree:

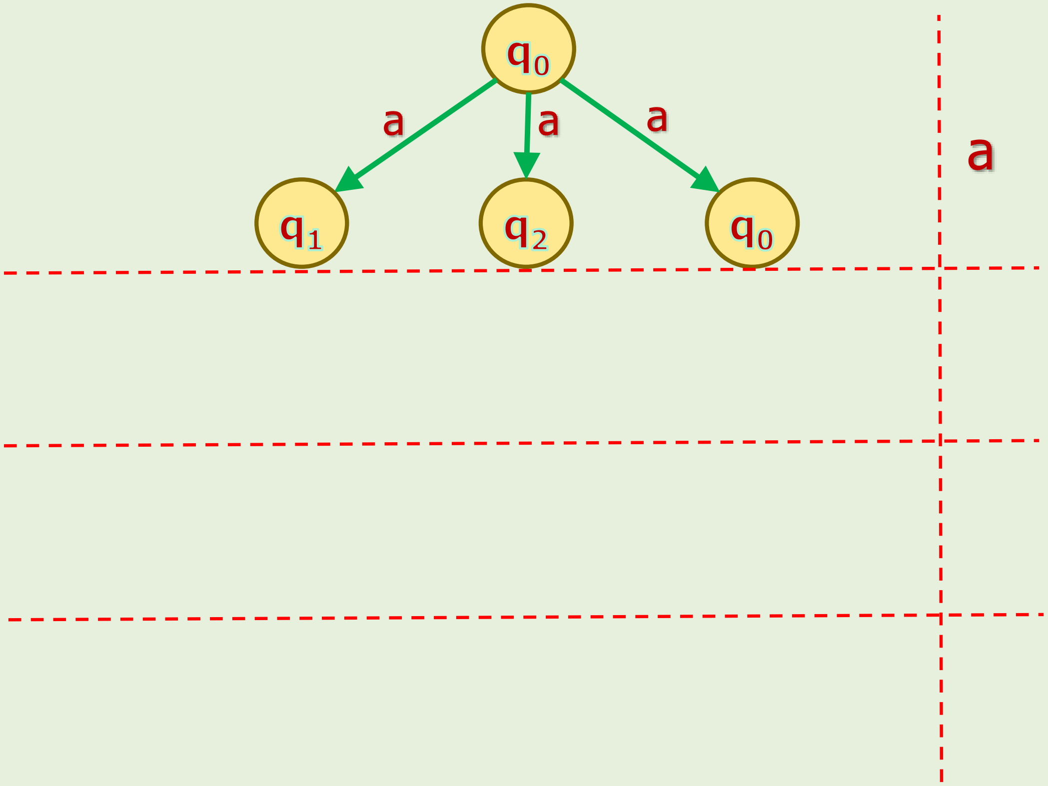


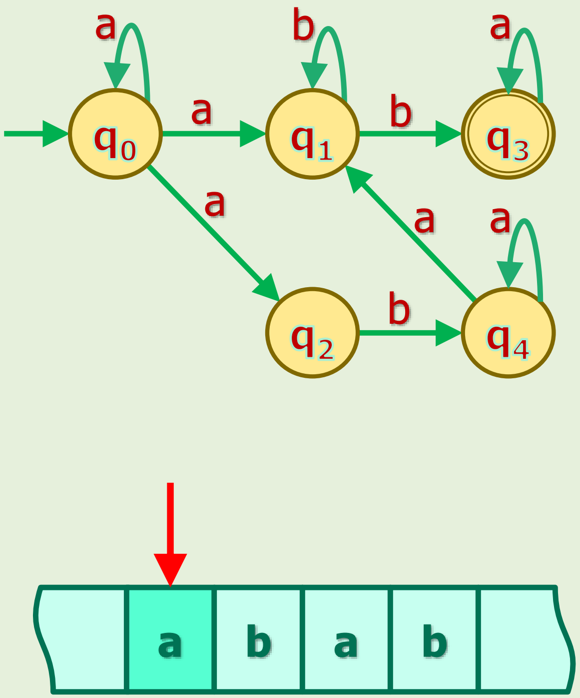




Nondeterministic TMs vs Standard TMs

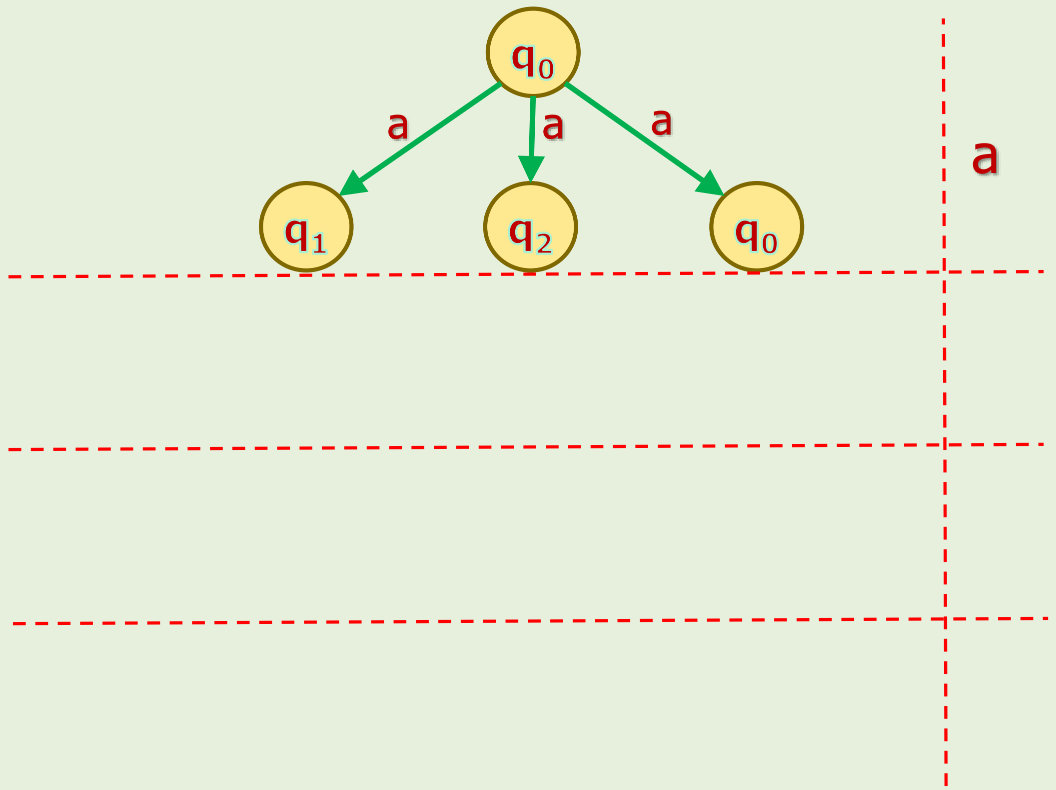
- **Proof of 2 (cont'd)**
- All processes for the string w = abab are organized in the following processes tree:

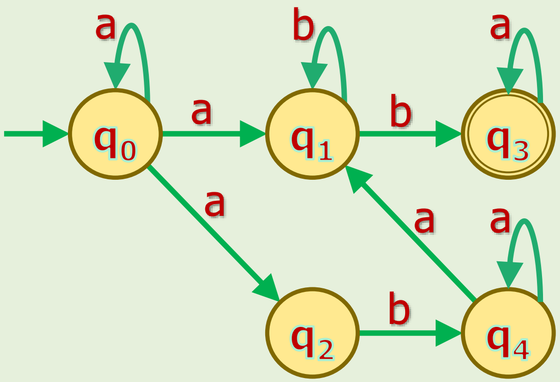


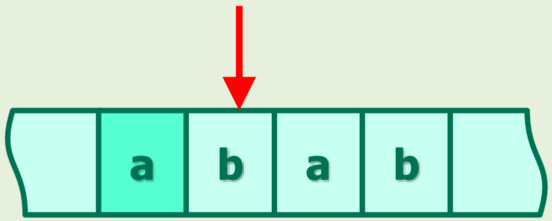


Nondeterministic TMs vs Standard TMs

- **Proof of 2 (cont'd)**
- All processes for the string w = abab are organized in the following processes tree:

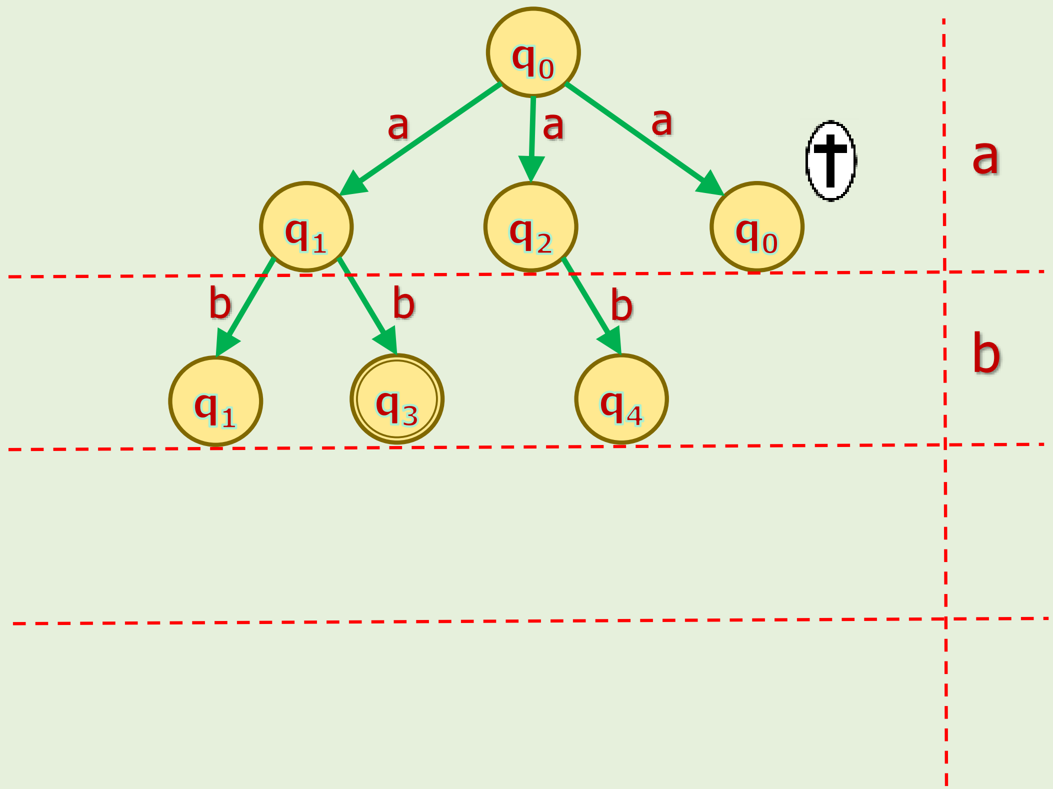


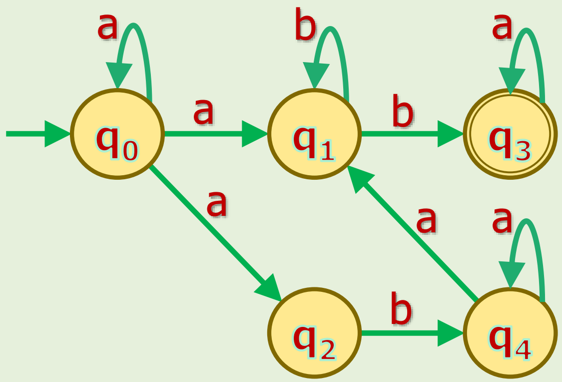


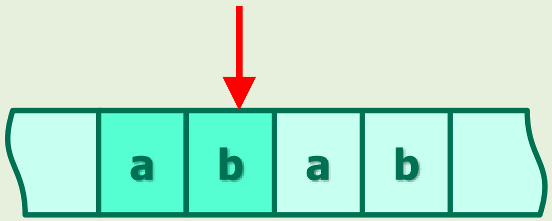


Nondeterministic TMs vs Standard TMs

- **Proof of 2 (cont'd)**
- All processes for the string w = abab are organized in the following processes tree:

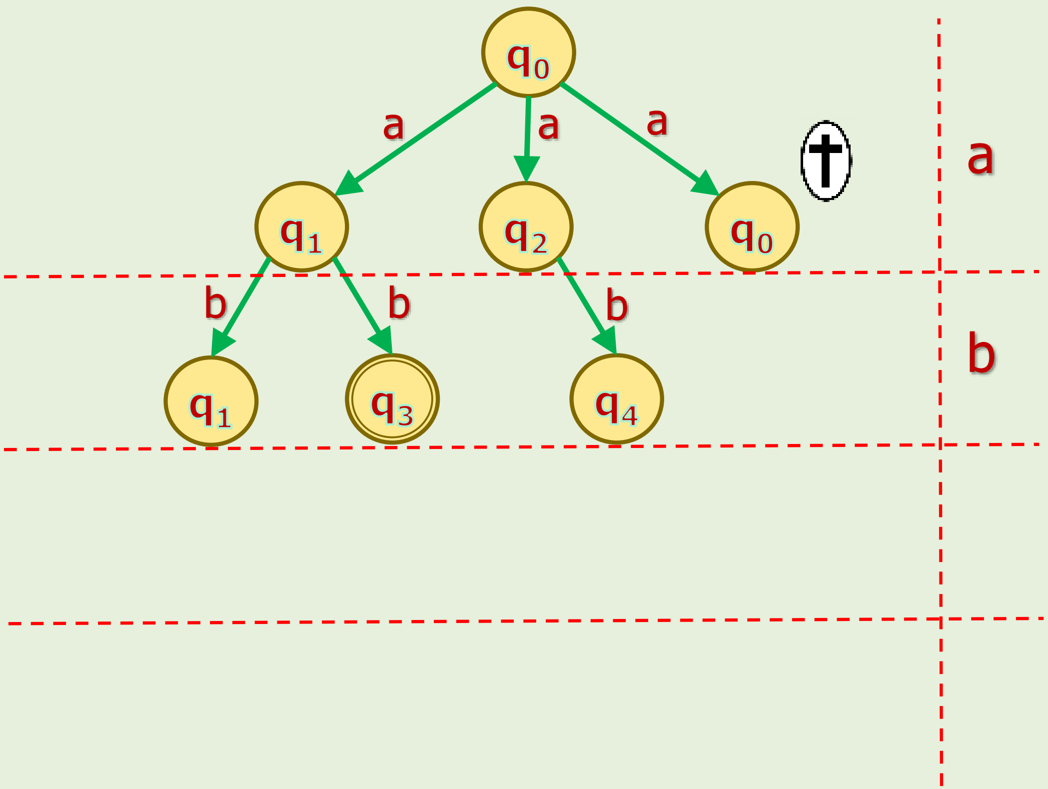


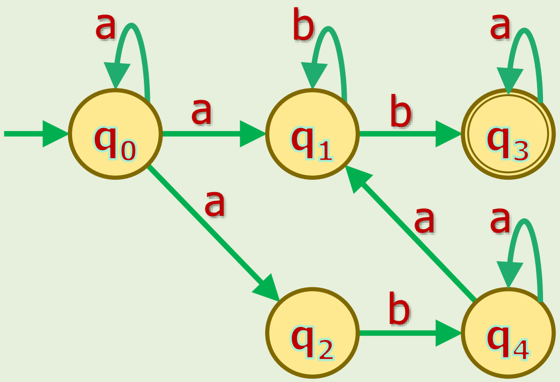


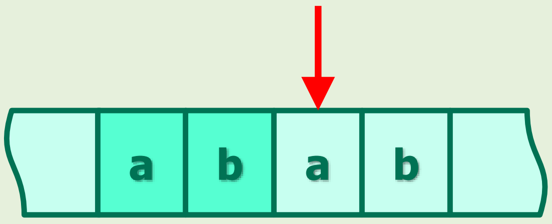


Nondeterministic TMs vs Standard TMs

- **Proof of 2 (cont'd)**
- All processes for the string w = abab are organized in the following processes tree:

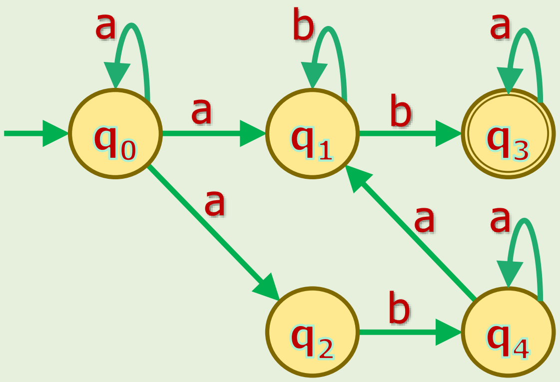


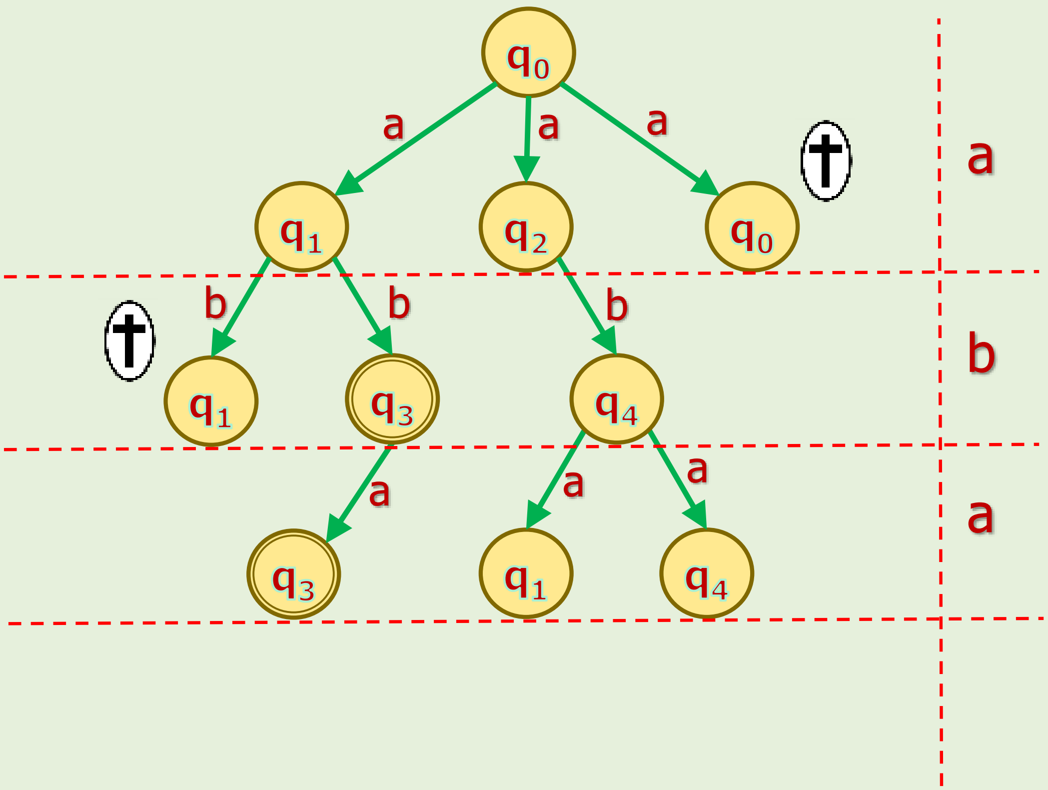


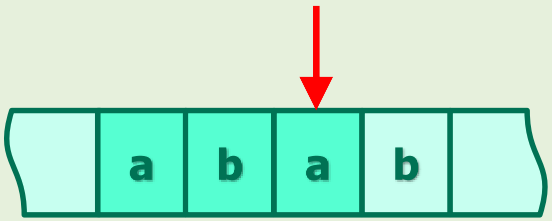


Nondeterministic TMs vs Standard TMs

- **Proof of 2 (cont'd)**
- All processes for the string w = abab are organized in the following processes tree:

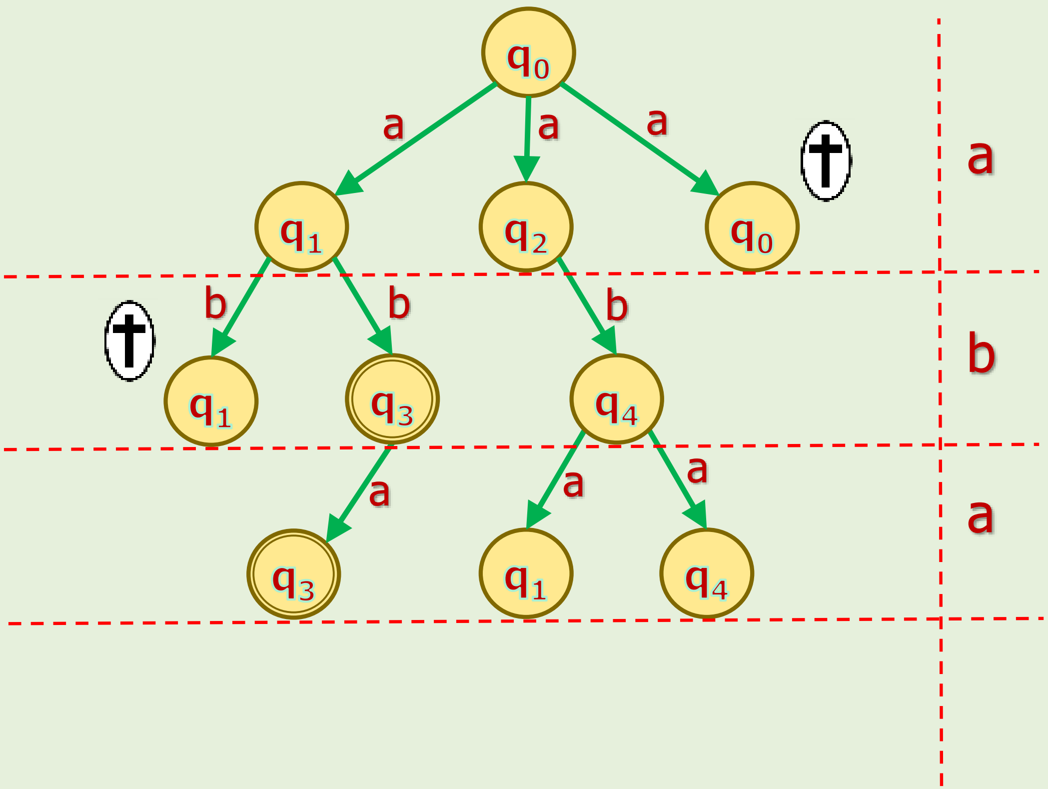


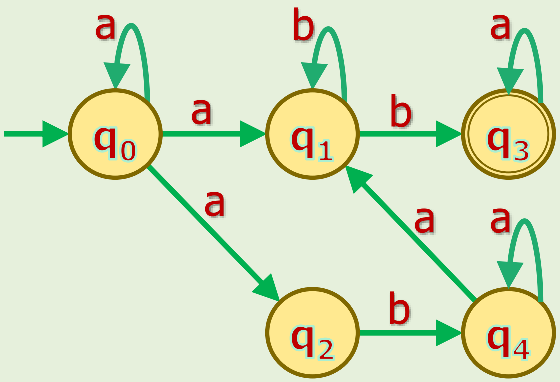


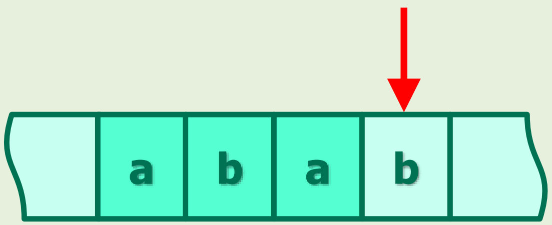


Nondeterministic TMs vs Standard TMs

- **Proof of 2 (cont'd)**
- All processes for the string w = abab are organized in the following processes tree:

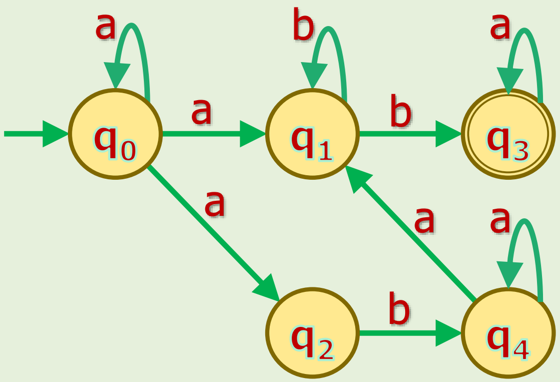


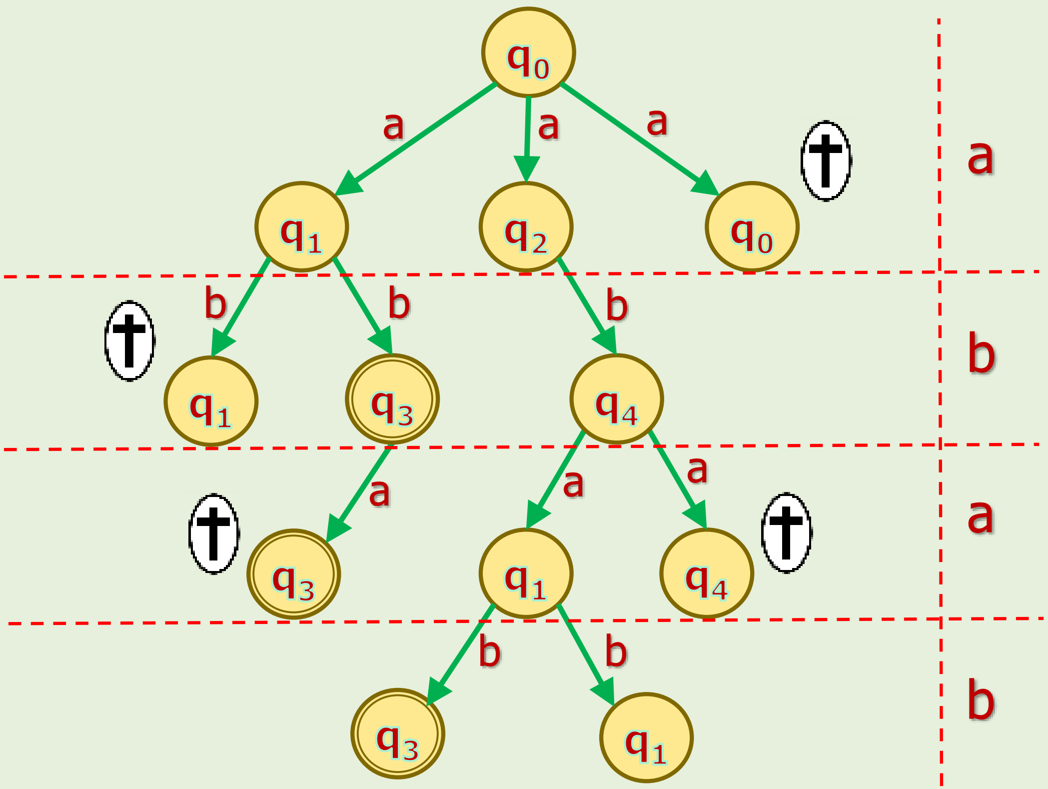


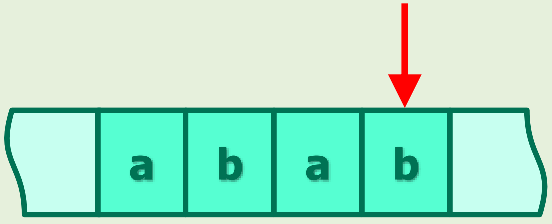


Nondeterministic TMs vs Standard TMs

- **Proof of 2 (cont'd)**
- All processes for the string w = abab are organized in the following processes tree:

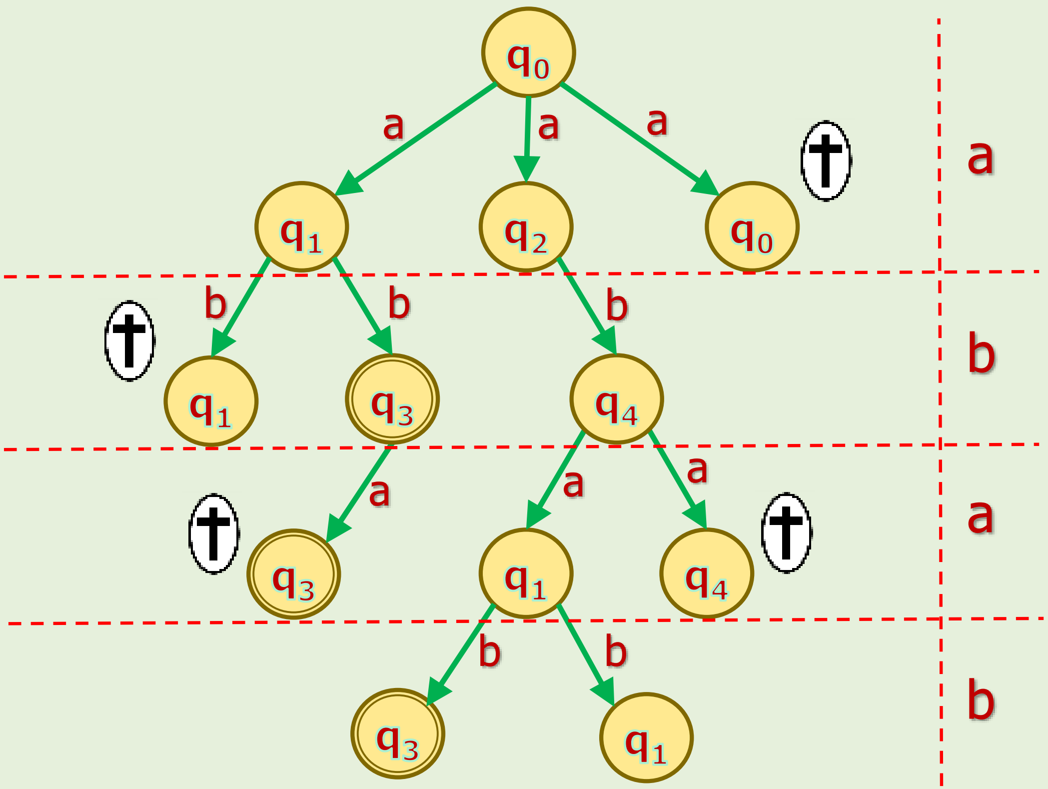


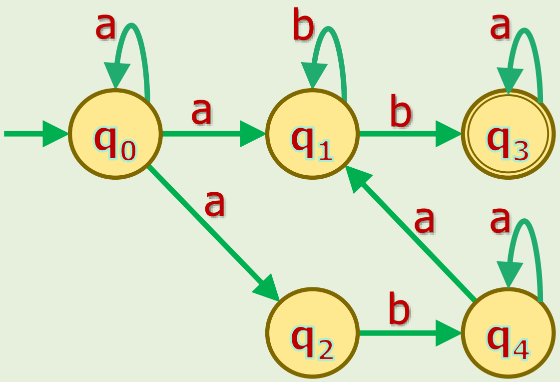


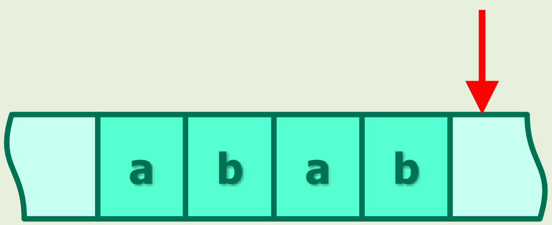


Nondeterministic TMs vs Standard TMs

- **Proof of 2 (cont'd)**
- All processes for the string w = abab are organized in the following processes tree:



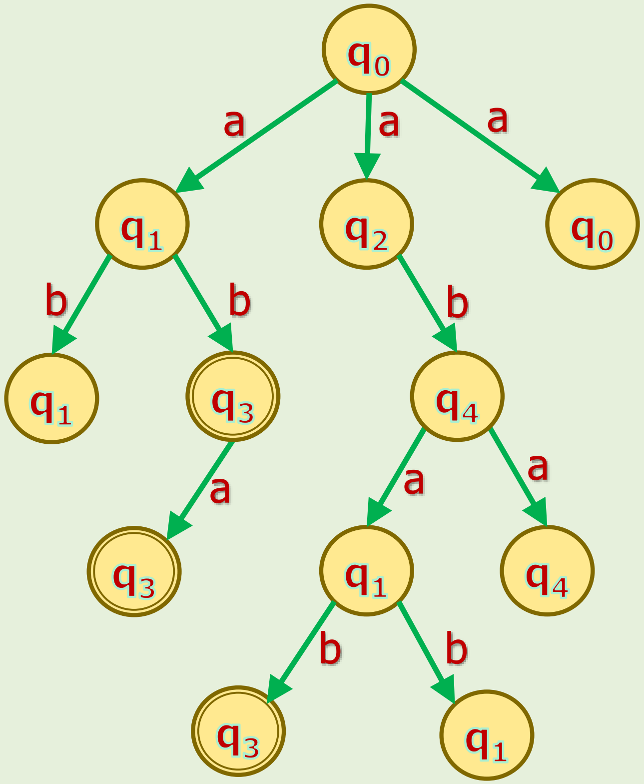




Nondeterministic TMs vs Standard TMs

- **Proof of 2 (cont'd)**
- Every walk from q_0 to a leaf is a process.
- Is every process a standard TM?
- Yes!

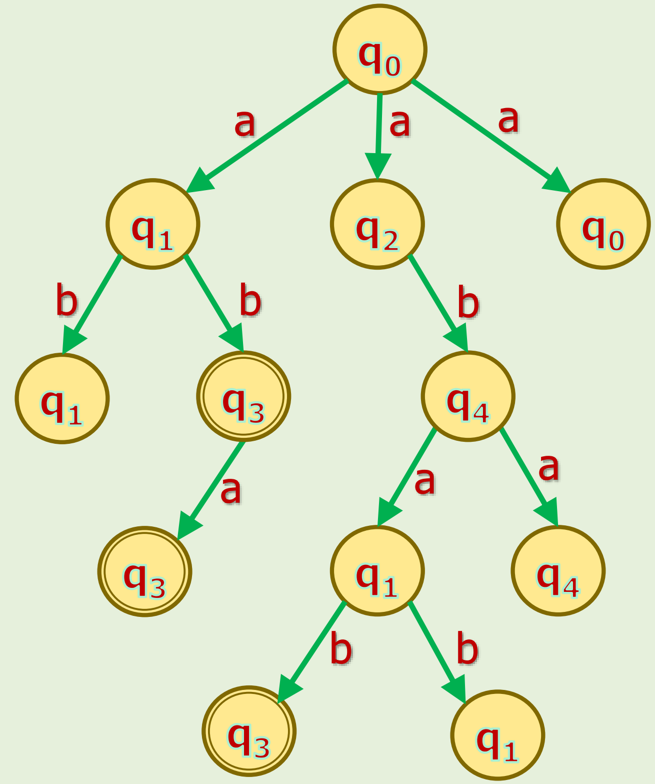
Therefore, an NTM
is a collection of standard TMs.



Nondeterministic TMs vs Standard TMs

Proof of 2 (cont'd)

- Can standard TM simulate NTMs?
- If it can handle the bookkeeping of the processes, then YES!
- Your term project and previous semesters term projects show that standard TMs can do this!



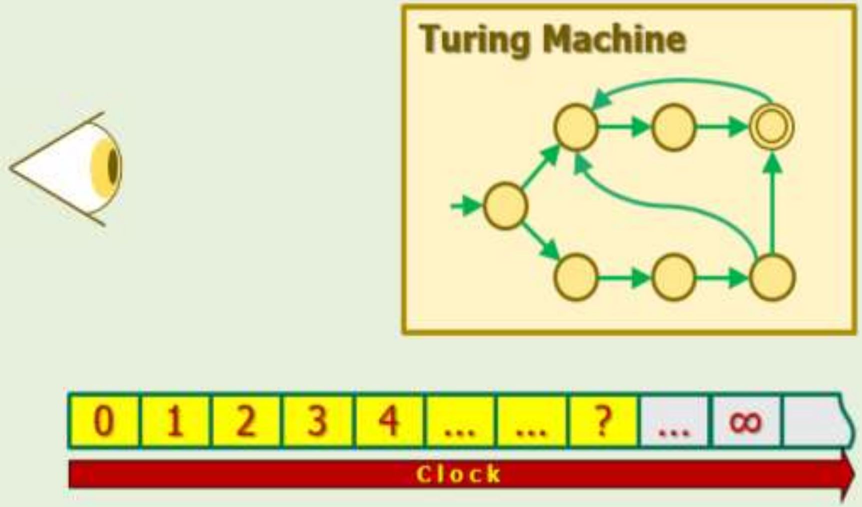
Nondeterministic TMs: Notes

1. Nondeterminism does NOT ADD any POWER to the automata theory.
   - It just speeds up the computation.

2. We are always looking for more power and speed is NOT our concern yet.
   - Speed will be a matter of concern when we will be talking about "complexity theory".

3. Quantum computing tries to implement nondeterminism!
   - So, it does NOT add any power to computing too!

Another $1,000,000 Question



- An observer is looking at a TM that is working for a long time!

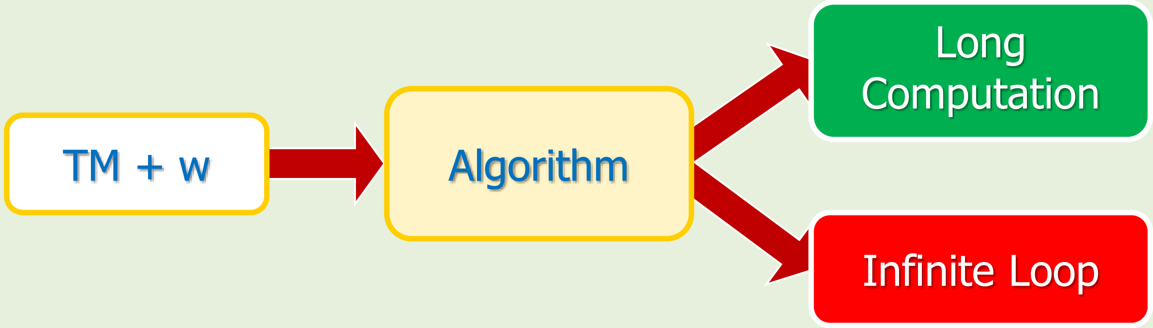
- How can the observer figure out whether ...
  1. it is in the middle of a very long computation?

     OR

  2. it got stuck in an infinite loop?

Another $1,000,000 Question

- Let's formulate the question in computer science terminology!
- We are looking for the following algorithm:



- Note that the algorithm must be able to solve the problem for any arbitrary TM against any arbitrary string $w \in \Sigma^*$.
- Do you think this is a solvable problem?

- As we'll see later, this question was asked and responded by Alan Turing in 1936!

References

1. Linz, Peter, "An Introduction to Formal Languages and Automata, 5th ed.," Jones & Bartlett Learning, LLC, Canada, 2012
   ISBN: 978-1-4496-1552-9
2. Michael Sipser, "Introduction to the Theory of Computation, 3rd ed.," CENGAGE Learning, United States, 2013
   ISBN-13: 978-1133187790