San José State University
Department of Computer Science

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

Turing Machines

(Part 2)

Lecture 16

Day 16/31

CS 154

Formal Languages and Computability

Spring 2019

Agenda of Day 16

- Announcement
- Solution and Feedback of Quiz 5
- Summary of Lecture 15
- Lecture 15: Teaching ...
  - Turing Machines (Part 1)

Announcement

- Your term project has been posted!
  - We are couple of days ahead of the schedule!

- A new assignment about the term project is posted too.

- What our real developers do?
- What the lazy cheaters and Googlers do?!

Solution and Feedback of Quiz 5 (Out of 22)

Section	Average	High Score	Low Score
01 (TR 3:00 PM)	19.47	22	15
02 (TR 4:30 PM)	17.83	22	4
03 (TR 6:00 PM)	18.35	21	13.5

Summary of Lecture 15: We learned ...

PDAs vs DFAs/NFAs

- PDAs are more powerful because:
  1. They can recognize all languages recognized by DFAs/NFAs.
     - This was proved by converting NFAs to PDAs.
  2. There are some languages that NFAs cannot recognize but PDAs can, such as: a^nb^n and ww^R.
- The portion of languages that PDAs can recognize is called context free (will be covered when talking about grammars).
- There are still some non-regular languages that cannot be recognized by NPDAs, such as:

$$ww \text{ and } a^nb^nc^n$$

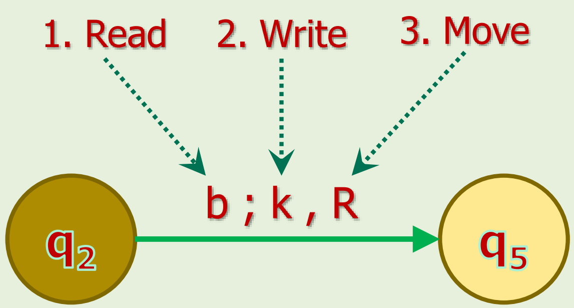
Any question?

Summary of Lecture 15: We learned ...

Standard Turing Machines (TMs)

- NPDAs are unable to accept some languages like $a^nb^nc^n$ and ww.
- The limitation of NPDAs is ...
  - ... stack is not so flexible in storing and retrieving data.
  - ... we lose some data when we access the older data.
- We replaced stack with RAM and ...
- ... introduced Turing machines (TM) to overcome this limitation.
- TMs have both deterministic and nondeterministic TMs (NTM) versions.

- The main difference between TMs and NPDAs is ...
  - ... we have the ability to move the read/write head to the left or right.
- We talked about the structure of TMs.



Any Question

4. How TMs Work

4. How TMs Work

Repeated

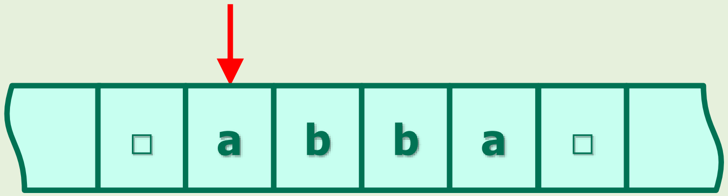
- To understand how TMS work, we should clearly respond to the following questions:

1. What is the "starting configuration"?
2. What would happen during a timeframe?
3. When would the machine halt (stop)?
4. How would a string be Accepted/Rejected?

4.1. TMs Starting Configuration

Clock

- The clock is set at timeframe 0.

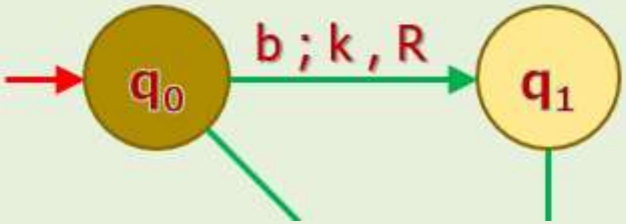


Input / Output Tape

- The tape has already been initialized with blank symbols '□'.
- The input string has already been written somewhere on the tape.
- The read-write head is pointing to the left-most symbol.

Control Unit

- The control unit is set to initial state.



4.2. What Happens During a Timeframe

- During a timeframe,
  the machine "transits" (aka "moves")
  from one configuration to another.

- Several tasks happen during a timeframe.
- The combination of these tasks is called a "transition".

- Let's first visualize these tasks through some examples.
- Then, we'll summarize them in one slide.

4.2. What Happens During a Timeframe

Transition Examples

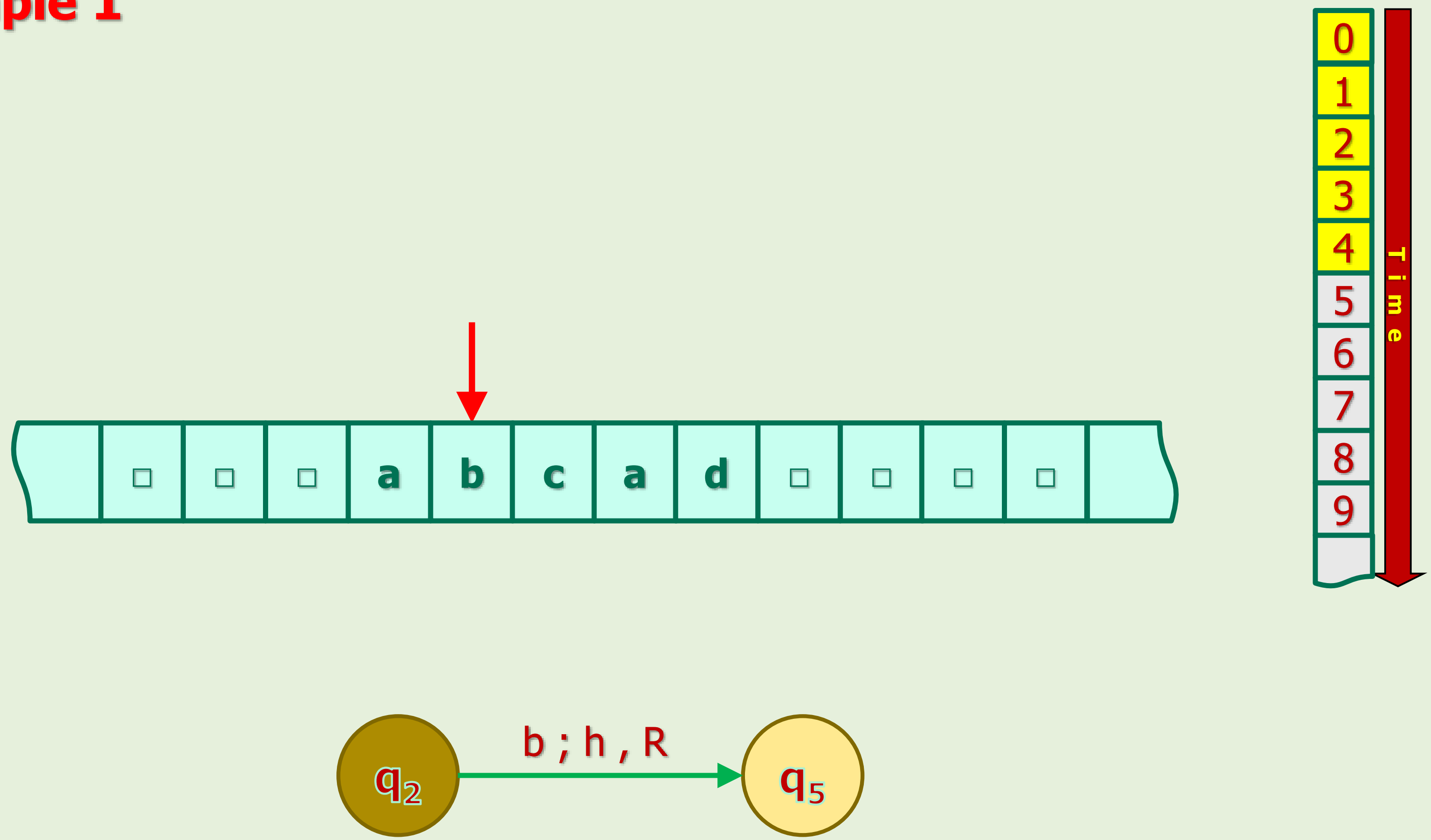
Transition Examples

- The next examples will show:
  - a partial transition graph
  - an input / output tape
  - a clock

- We assume that the machine is in the middle of its operation at timeframe n.
- The question is: in what configuration would the machine be at timeframe n+1?

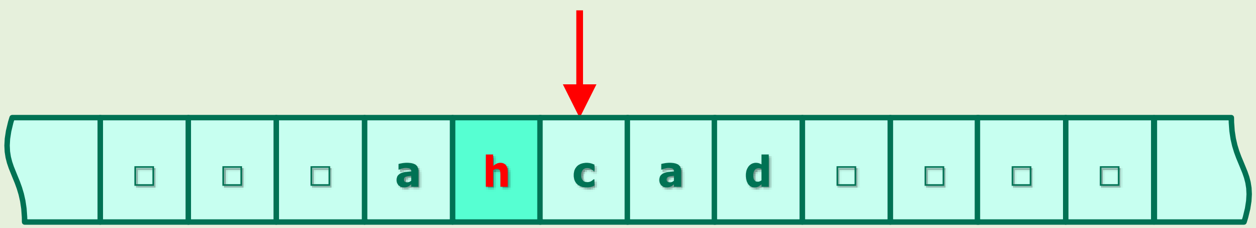
Transition Example

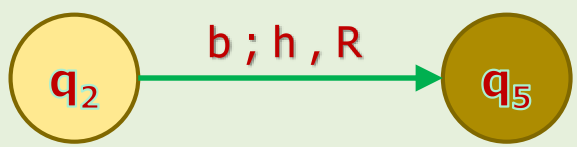
Example 1

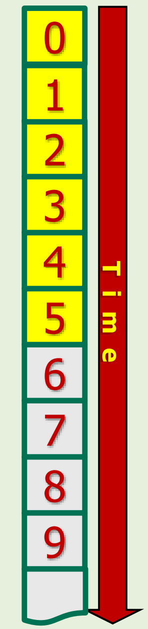


Transition Example

Example 1 (cont'd)







4.2. What Happens During a Timeframe

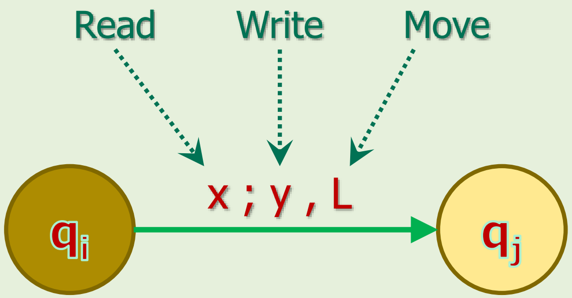
Transition

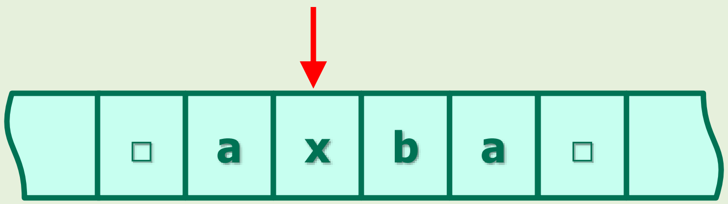
- The following tasks happen during a timeframe:

  1. A symbol at which the read-write head is pointing, is read.
  2. A symbol is written into the same cell.
  3. The read-write head is moved one cell to the left or right.
  4. The control unit makes its move based on the "logic of the transition".

- What is the "logic of the transition" of TMs?

TMs' Logic of Transitions





If (Condition)

in q_i

AND

the input symbol is 'x'

Then (Operation)

replace 'x' with 'y'

AND

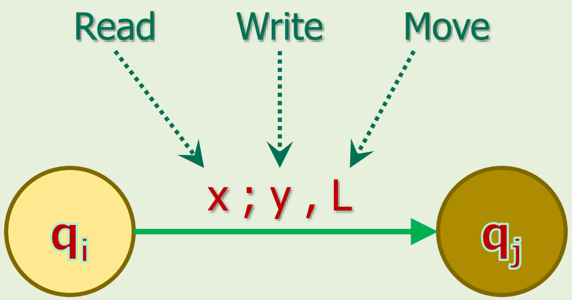
move the head to the Left

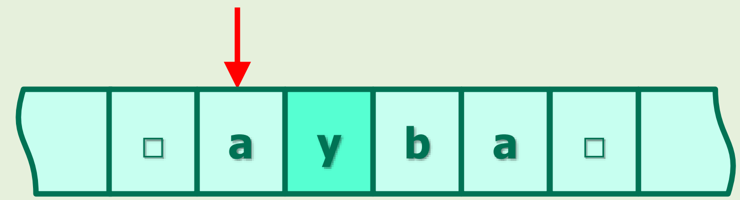
AND

transit to q_j

How does the machine look like after this transition?

TMs' Logic of Transitions





After the previous transition ...

- You might ask: what if the input is not 'x'?
- Good questions! We'll get back to this question later.

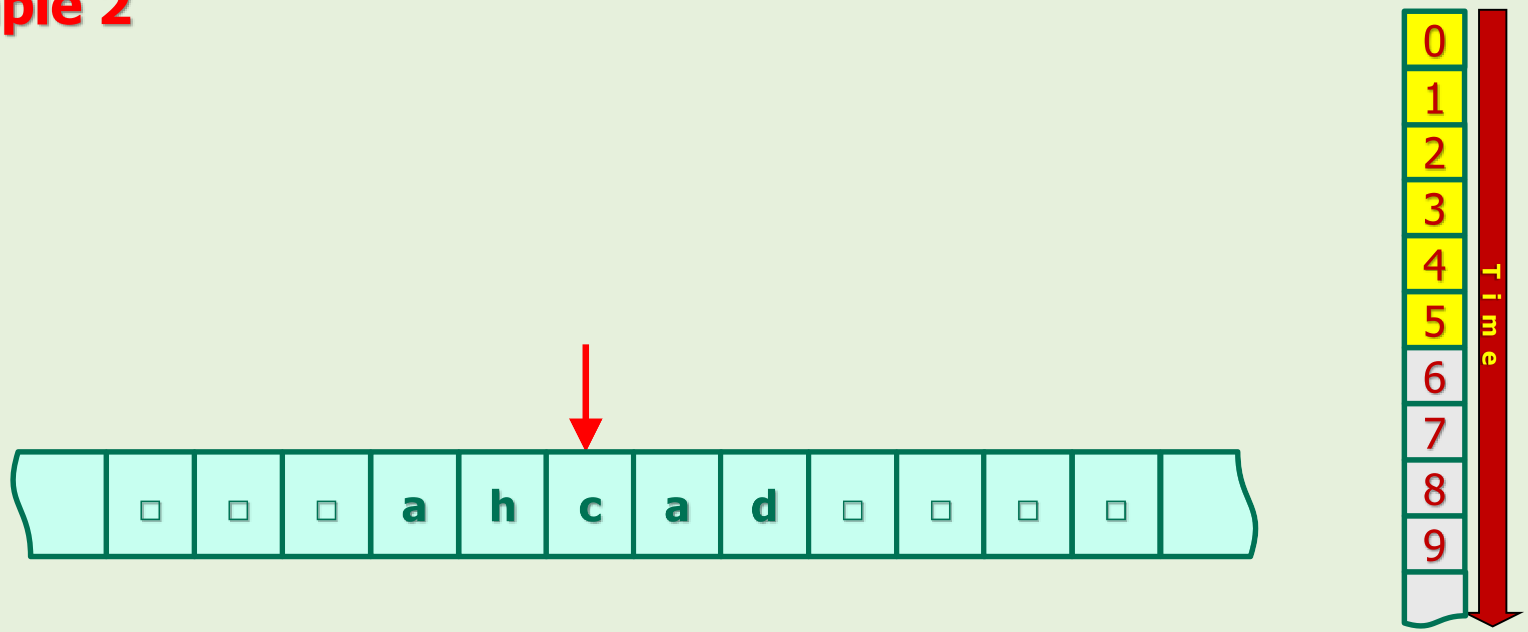
Multiple Labels

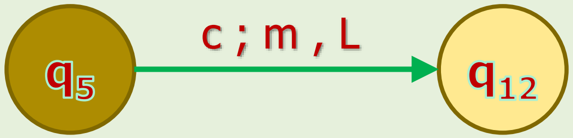
- A transition might have multiple labels.
- In that case, we stack them over the edges.

- Note that there is an OR between them.
- It means, in either condition, the machine transits and follows the label's operations.

Transition Examples

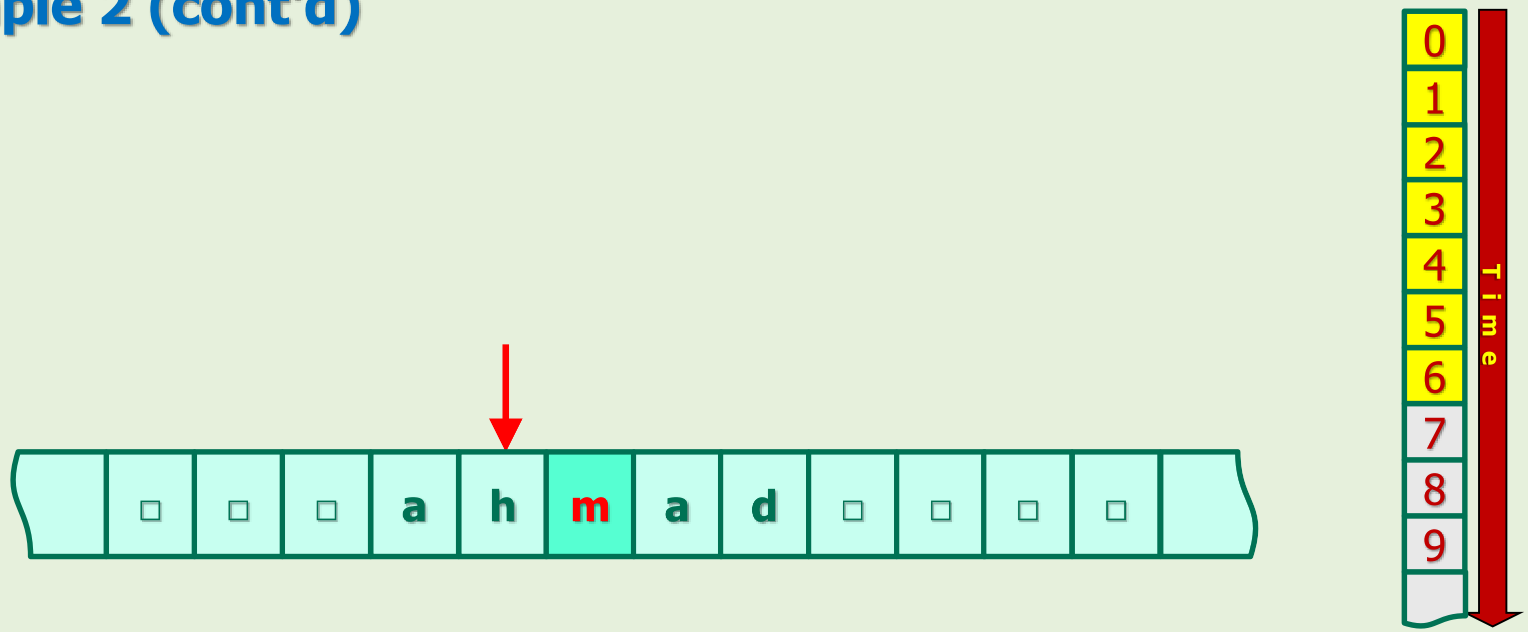
Example 2

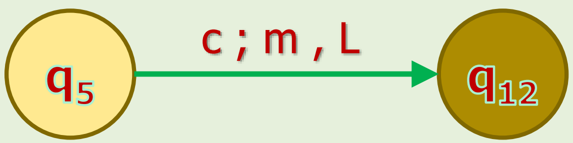




Transition Examples

Example 2 (cont'd)

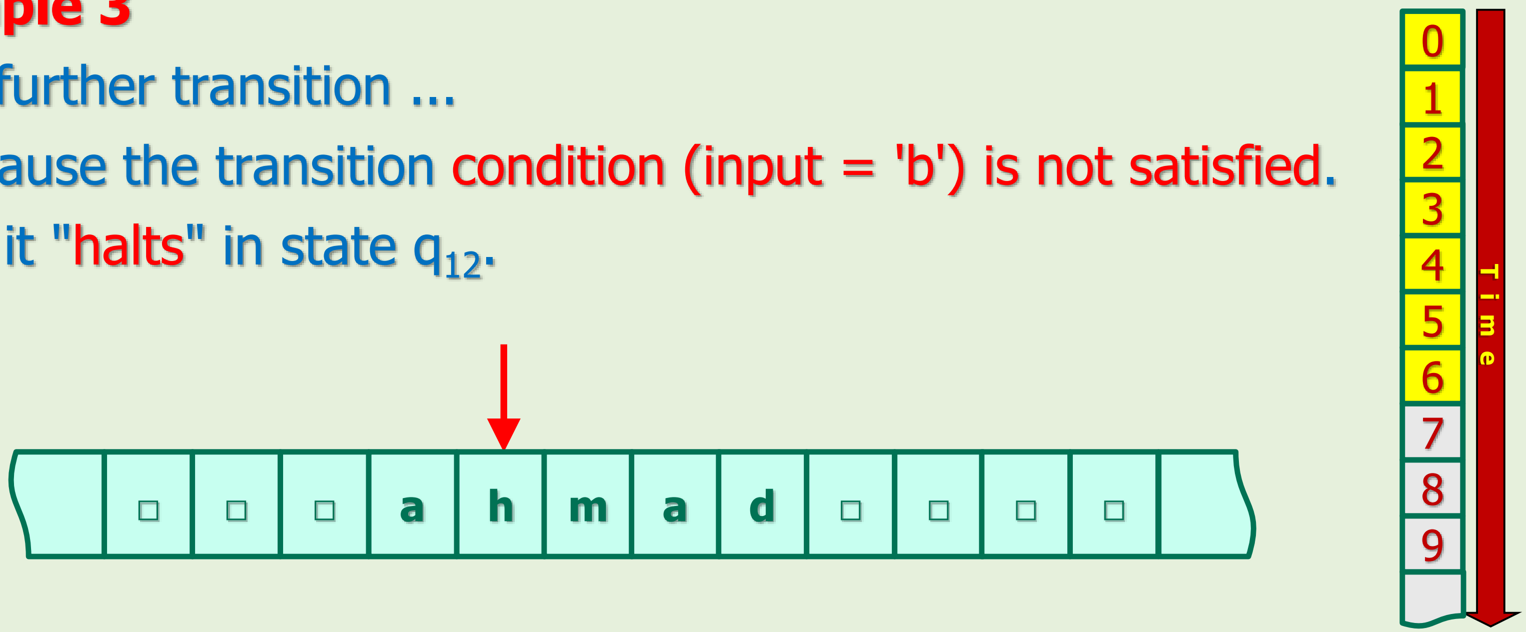




Transition Examples

Example 3

- No further transition ...
- Because the transition condition (input = 'b') is not satisfied.
- So, it "halts" in state q_{12}.



4.3. When TMs Halt

- From the previous example, we found out that:

  TMs halt when the next transition condition is NOT satisfied.

Halt Logical Representation

$$\left.\begin{array}{l}\text{TMs halt.} \equiv \mathbf{h} \\ \text{IFF} \\ \text{They have zero transition.} \equiv \mathbf{z}\end{array}\right\} \quad \mathbf{z} \leftrightarrow \mathbf{h}$$

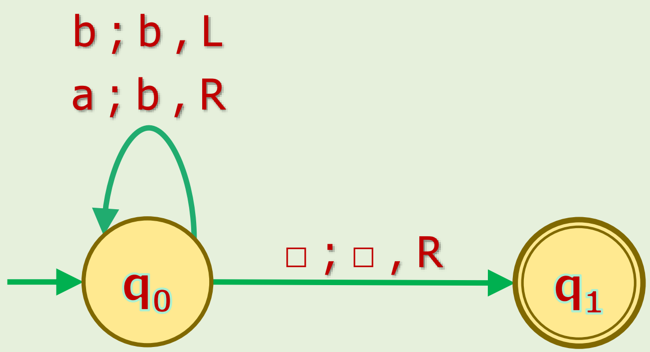
5. TMs in Action

Analysis Examples

5. TMs in Action

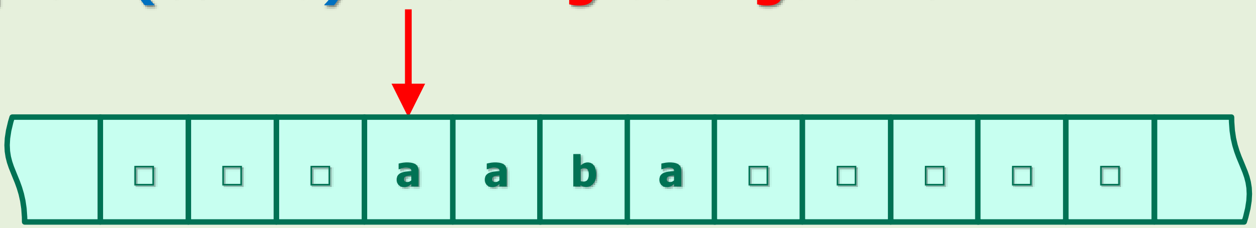
Example 4

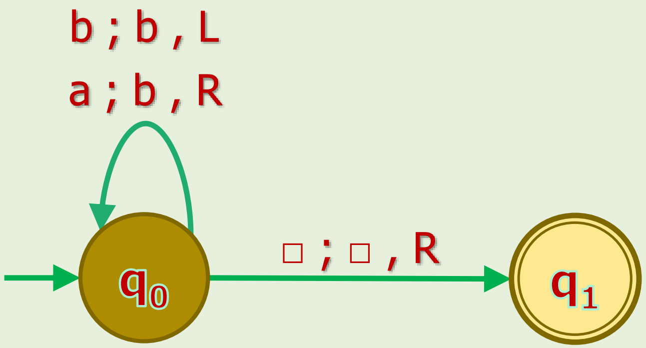
- Consider the following TM over $\Sigma = \{a, b\}$.
- Trace the machine's operations for the input "aaba".



5. TMs in Action

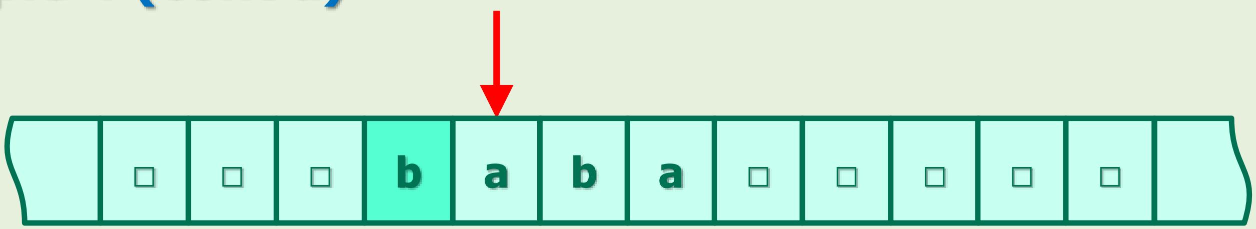
Example 4 (cont'd): Starting Configuration

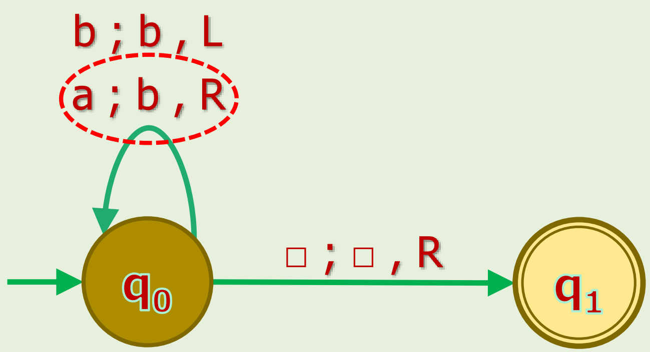




5. TMs in Action

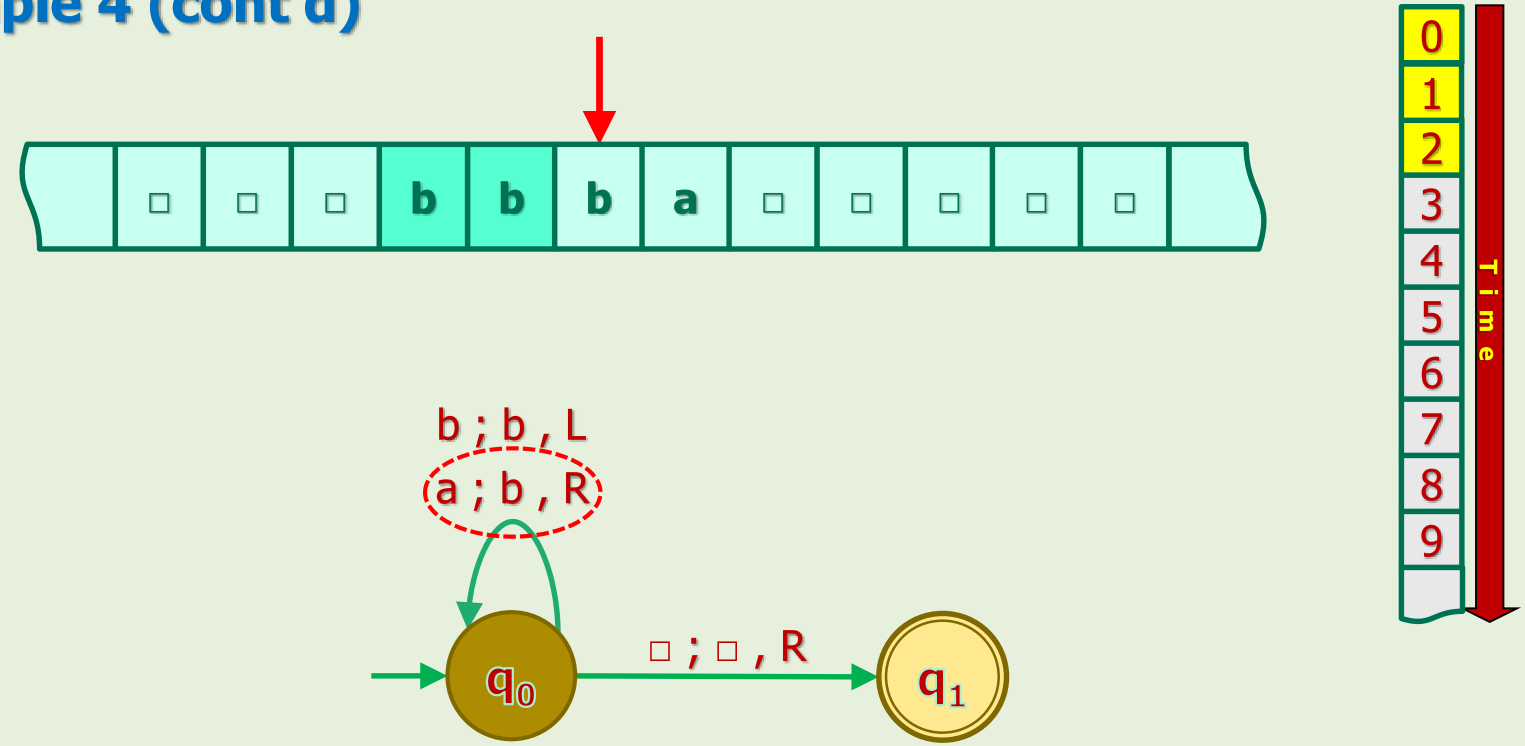
Example 4 (cont'd)





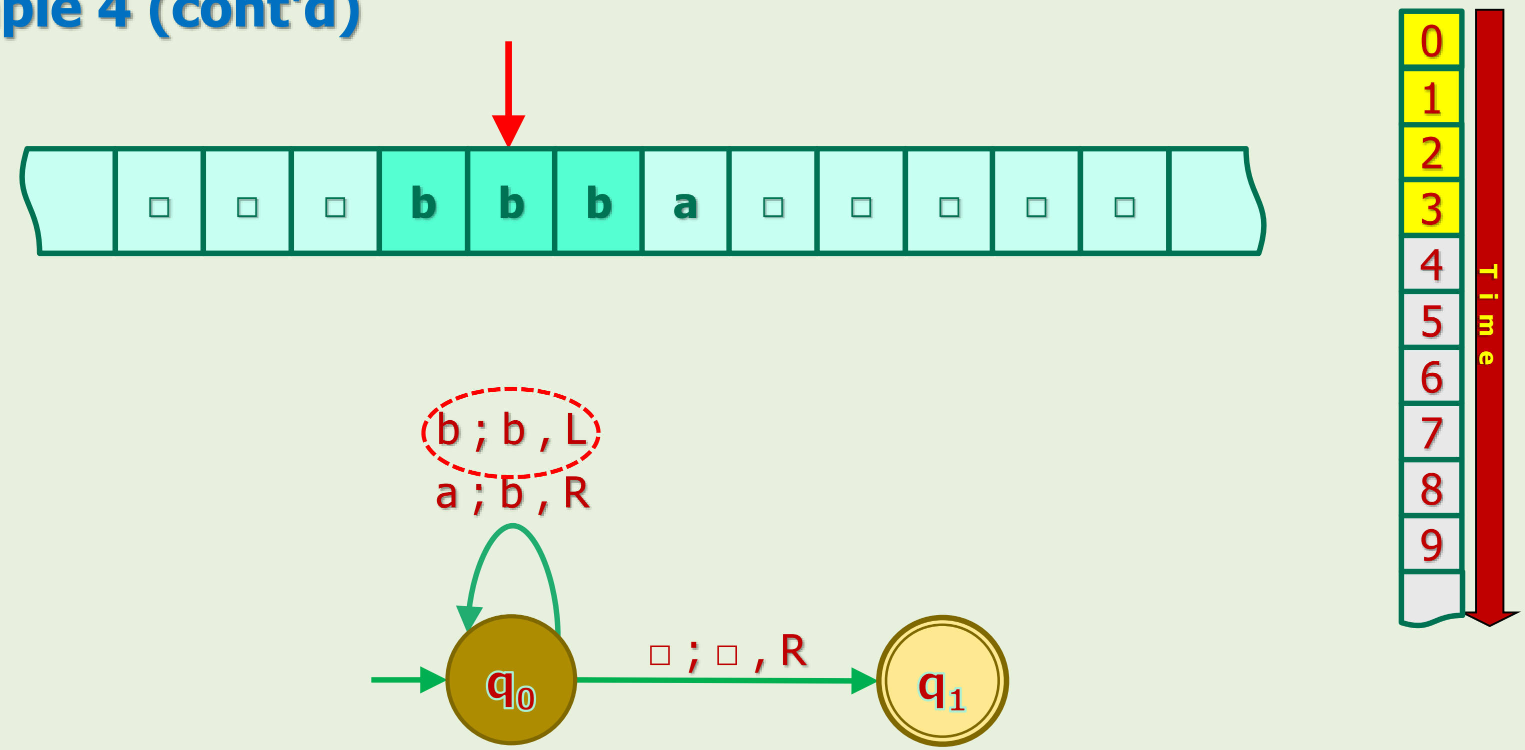
5. TMs in Action

Example 4 (cont'd)



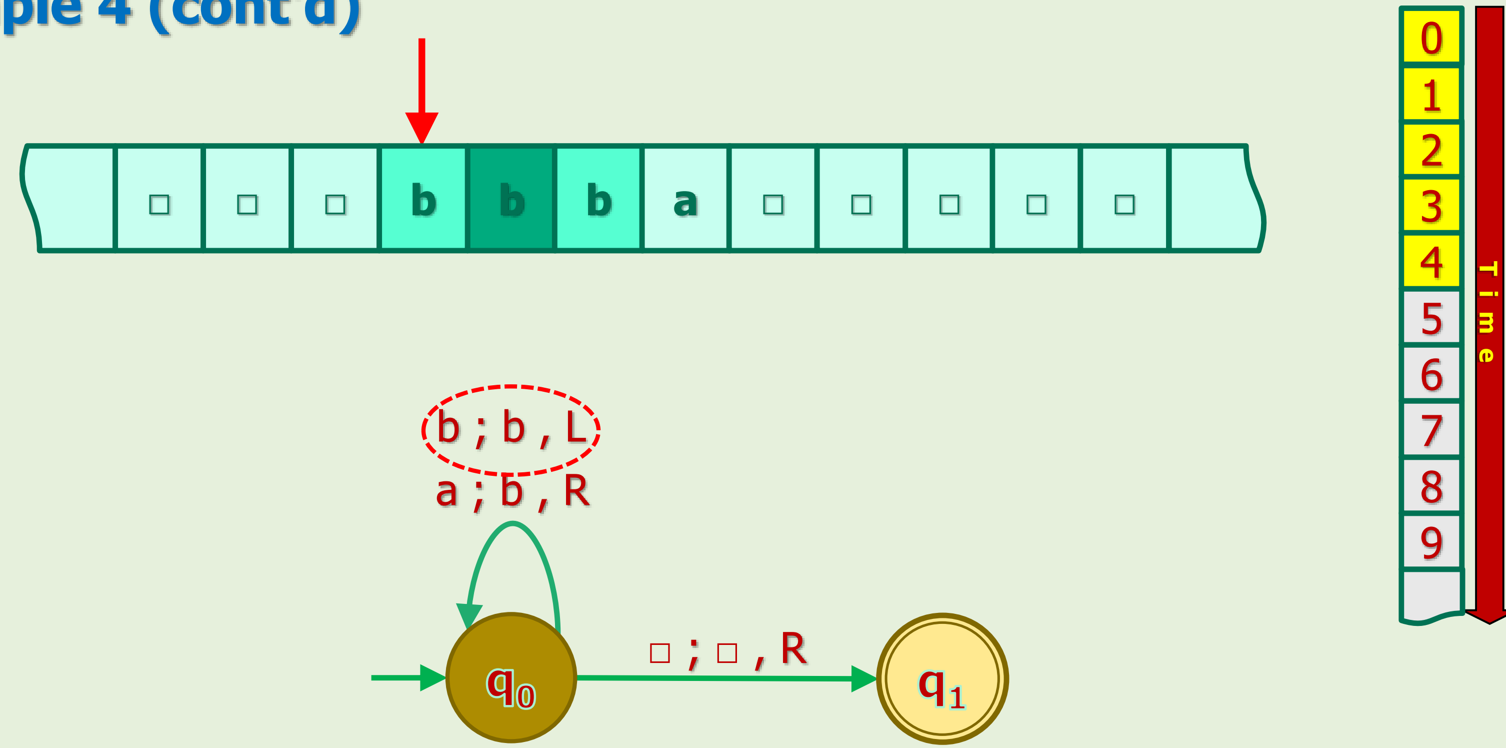
5. TMs in Action

Example 4 (cont'd)



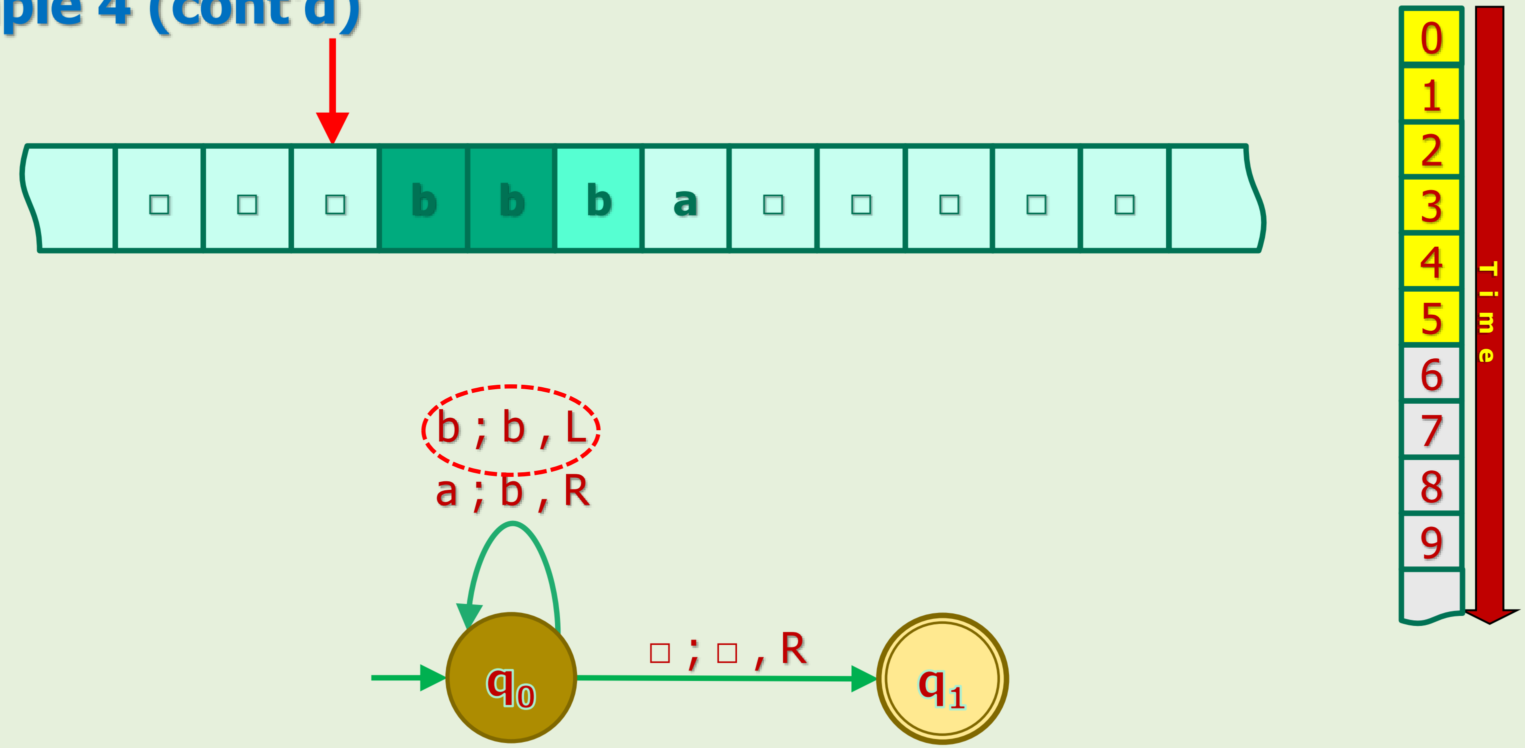
5. TMs in Action

Example 4 (cont'd)



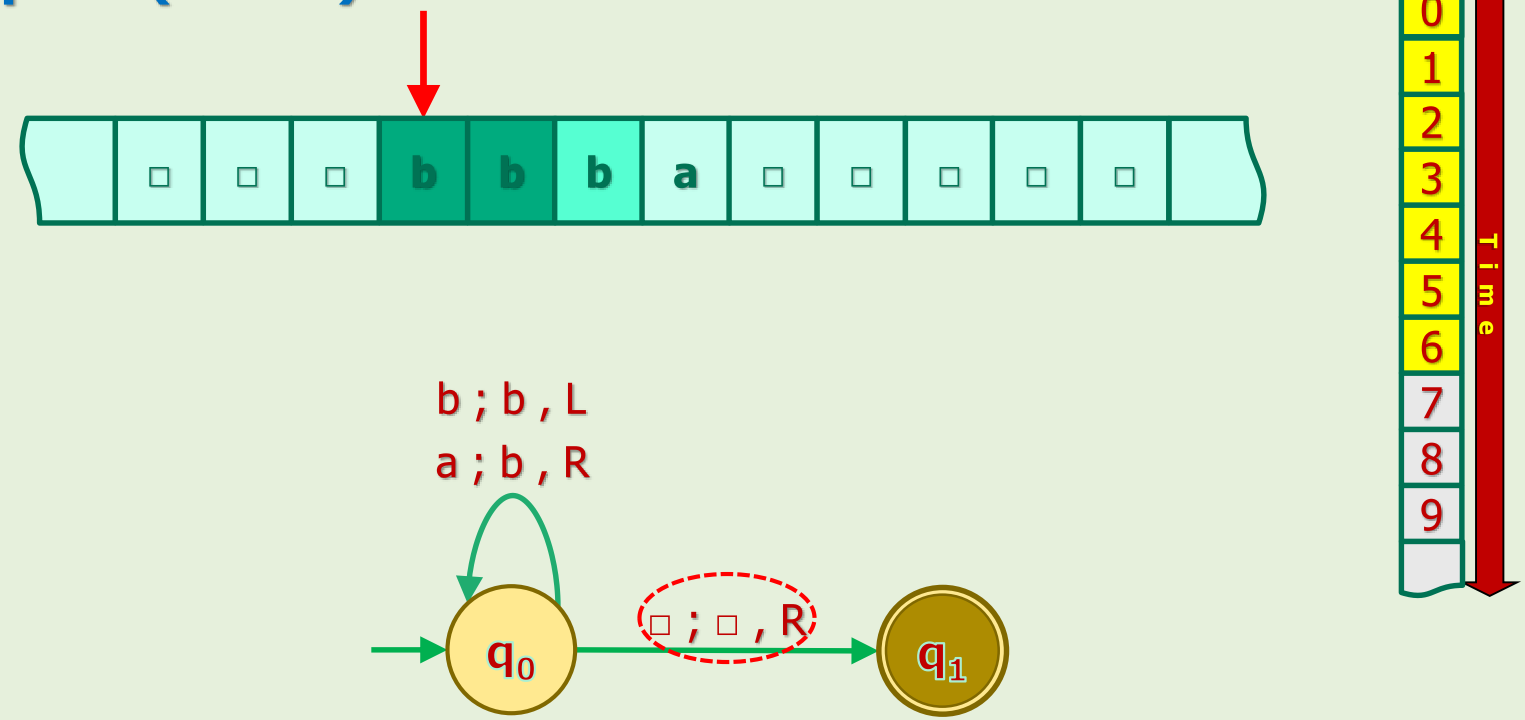
5. TMs in Action

Example 4 (cont'd)



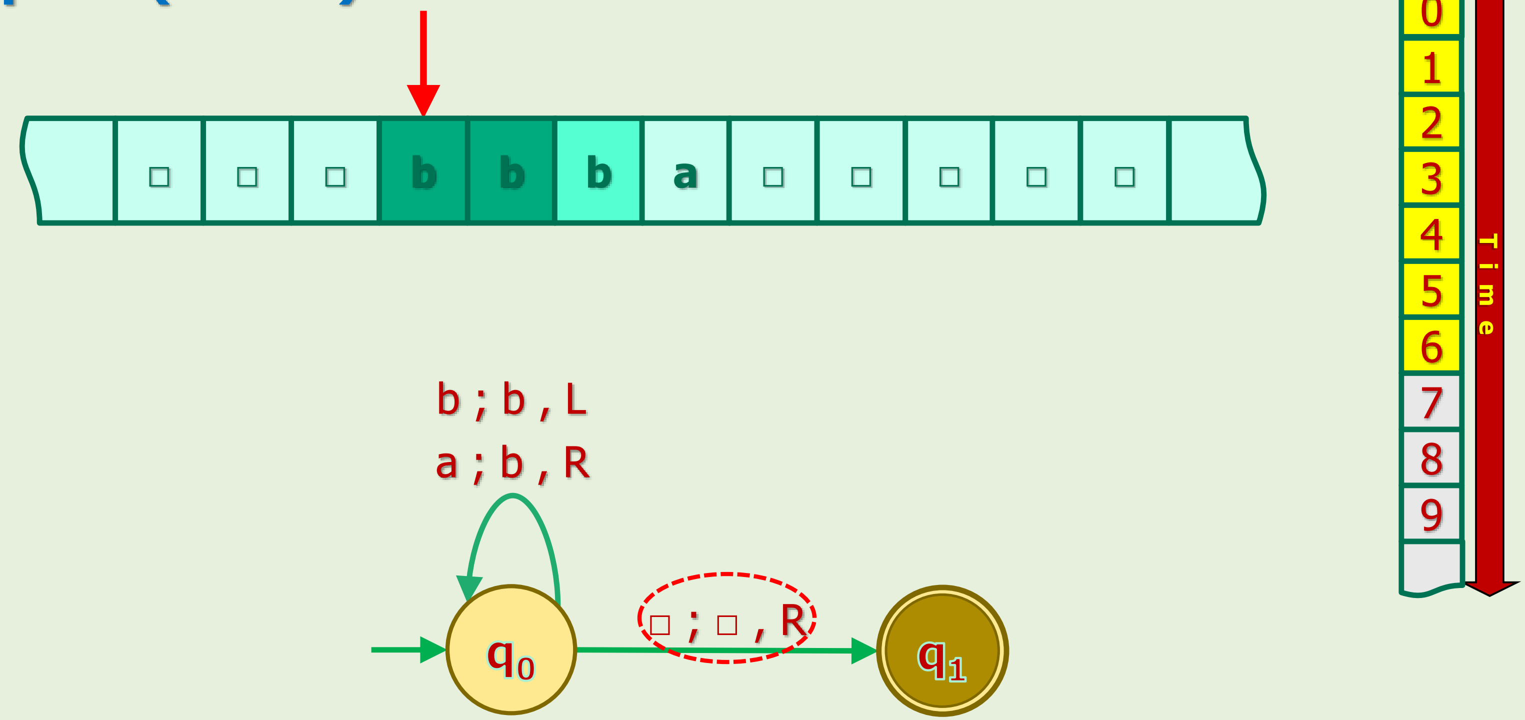
5. TMs in Action

Example 4 (cont'd)



5. TMs in Action

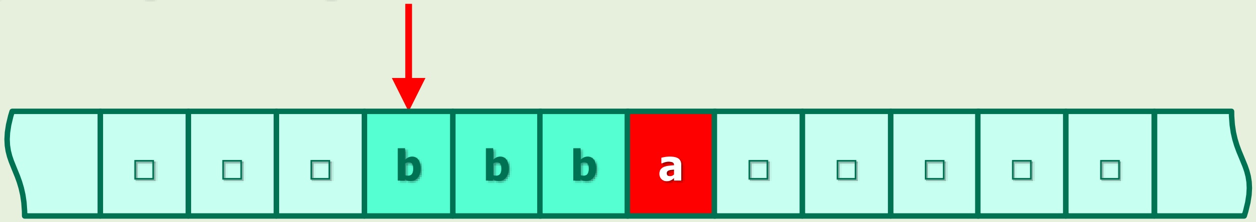
Example 4 (cont'd)



- The machine has no more transition.
- So, it halts.

Was The String Accepted?

Example 4 (cont'd)



- The machine halted in an accepting state.
- But the last symbol of the string (i.e. 'a') was never reached.

Question

- Was the string "aaba" accepted?

Answer

- It depends on how we define the string acceptance in TMs.

Was The String Accepted?

- If we judge based on the criteria of previous machines that were:

$$(\mathbf{h} \wedge \mathbf{c} \wedge \mathbf{f}) \leftrightarrow \mathbf{a}$$

- Then the answer would be "NO" because ...

all symbols were not consumed.

- But consuming the input symbols is meaningless for TMs. Why?
  - Because the head can move left or right.
  - So, some symbols might be visited several times while some other never reached.

- In practice, that is the TMs' designers responsibility to make sure that the machine halts in an accepting state when all symbols are visited.

4.4. How TMs Accept/Reject Strings

Logical Representation of Accepting Strings

- If we remove **c** from the conditions, then theoretically, the logical representation of accepting strings is ...

TMs accept a string w. ≡ **a**

IFF

They halt. ≡ **h**

AND

They are in an accepting (final) state. ≡ **f**

$$(h \wedge f) \leftrightarrow a$$

- Shorter version:
  The string w is accepted iff the TM halts in an accepting state.

4.4. How TMs Accept/Reject Strings

Logical Representation of Rejecting Strings

$$\sim(h \wedge f) \leftrightarrow \sim a$$

$$(\sim h \vee \sim f) \leftrightarrow \sim a$$

Translation

TMs reject a string w. ≡ $\sim$**a**

IFF

They do NOT halt. ≡ $\sim$**h**

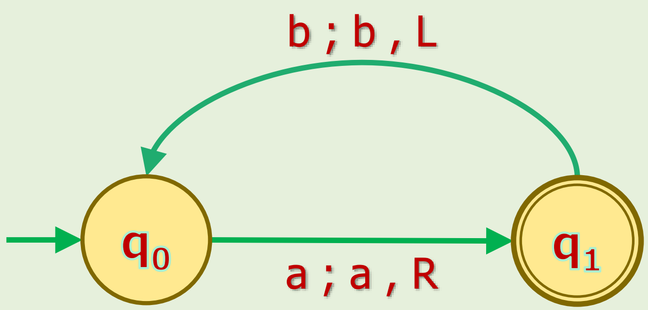
OR

They are NOT in an accepting (final) state. ≡ $\sim$**f**

A Special Phenomenon in TMs

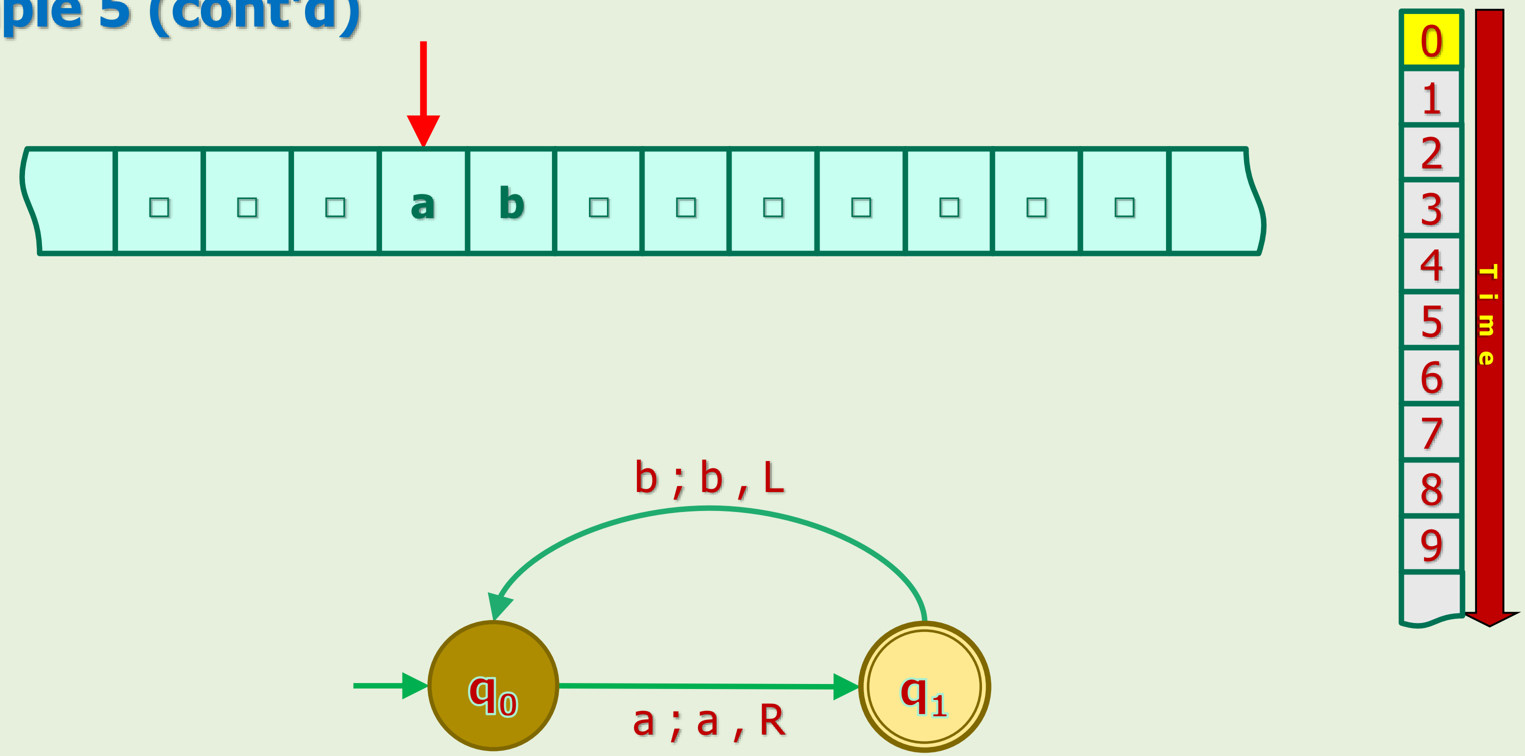
Example 5

- Trace the following TM for the input "ab".



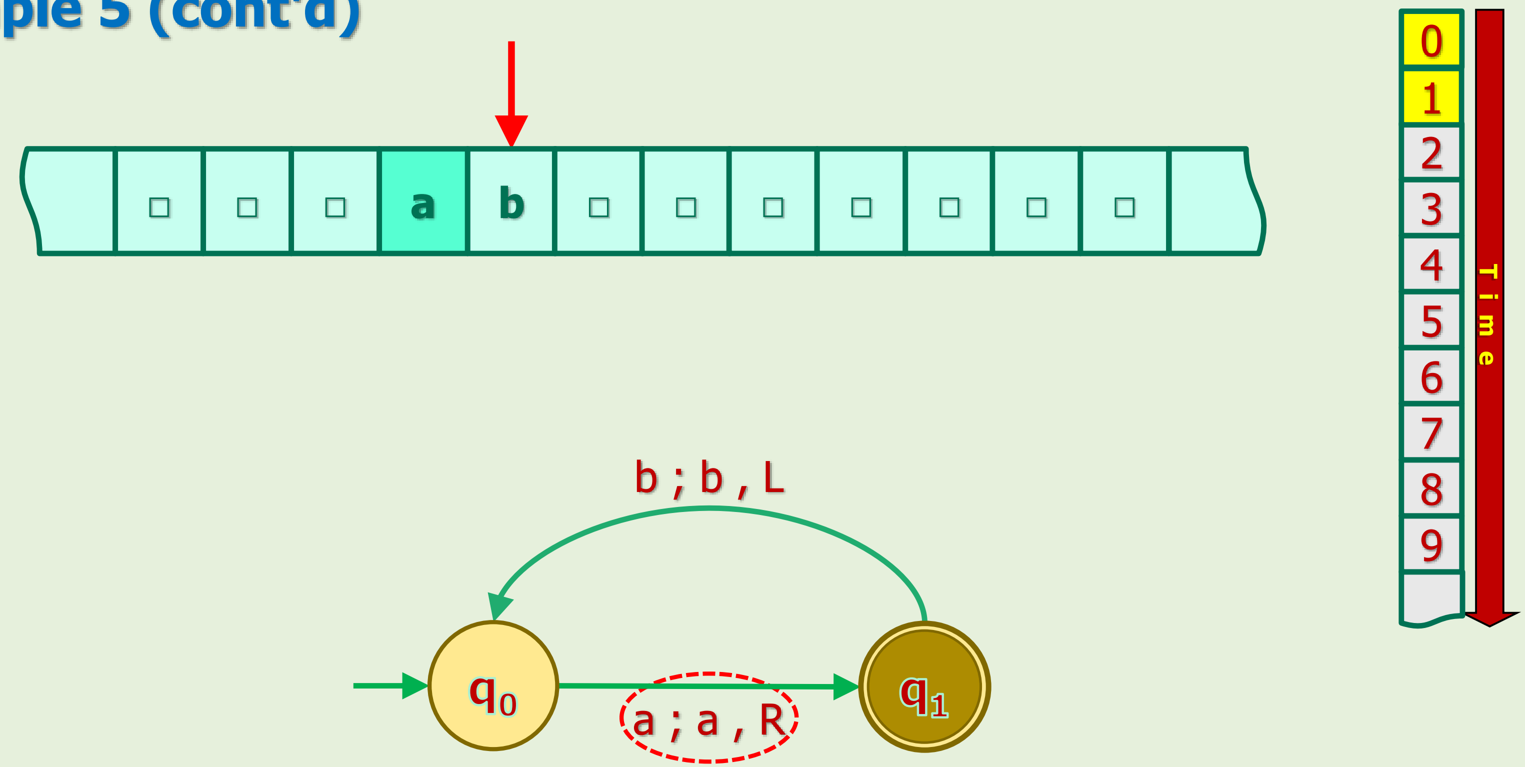
A Special Phenomenon in TMs

Example 5 (cont'd)



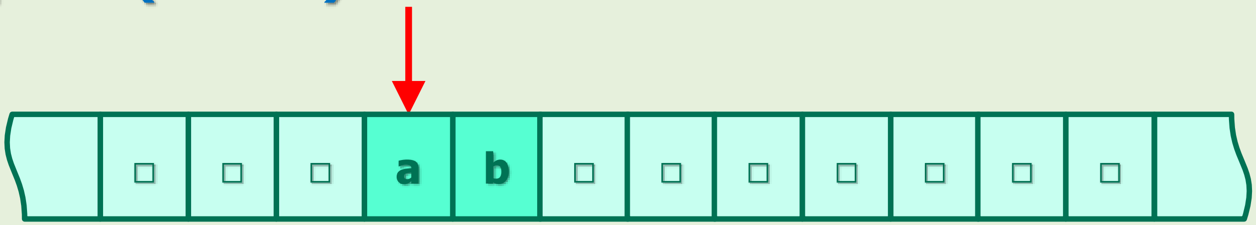
A Special Phenomenon in TMs

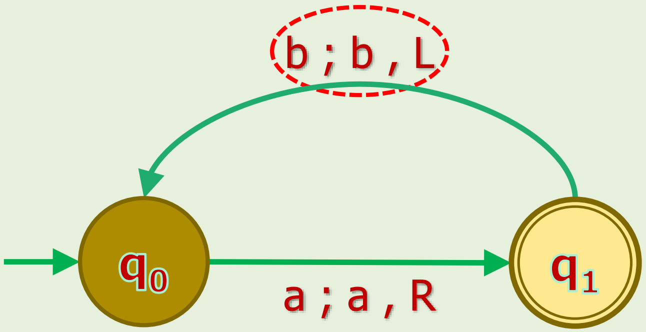
Example 5 (cont'd)



A Special Phenomenon in TMs

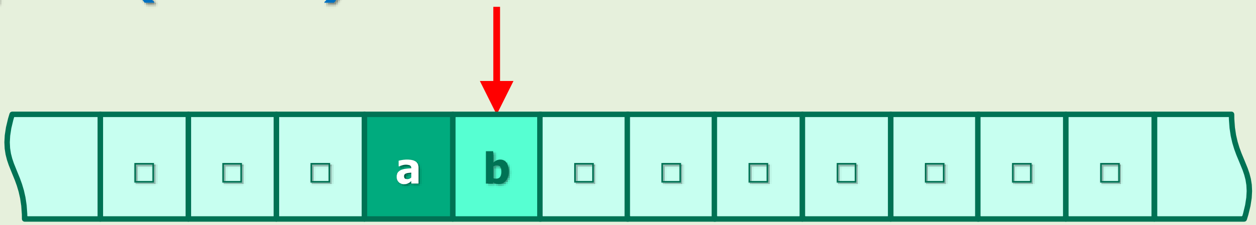
Example 5 (cont'd)

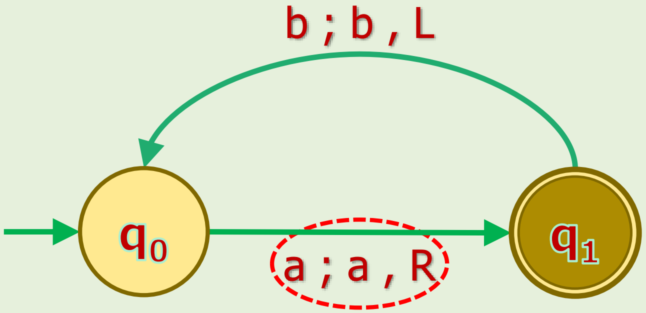




A Special Phenomenon in TMs

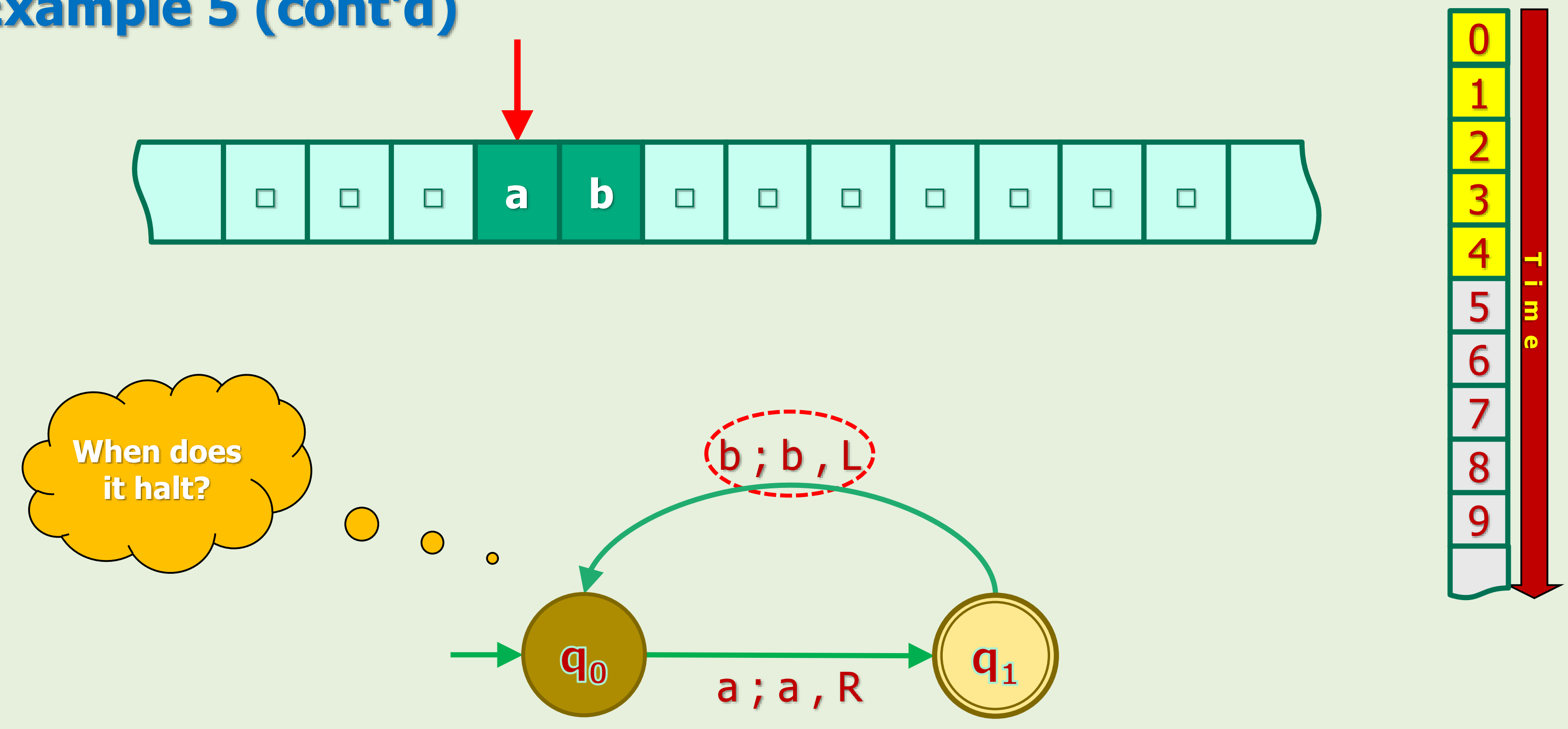
Example 5 (cont'd)





A Special Phenomenon in TMs

Example 5 (cont'd)



What was that phenomenon?

- The TM never halts.
- In other words, in some situations,

  A TM can fall into an "infinite loop".

- This phenomenon ...
- ... never happened in the previous DETERMINISTIC machines.

- What do you think is the reason?
  - This is the consequence of ...

    ... having freedom of moving the read-write head to the left or right.

A Side Note About Rejecting String

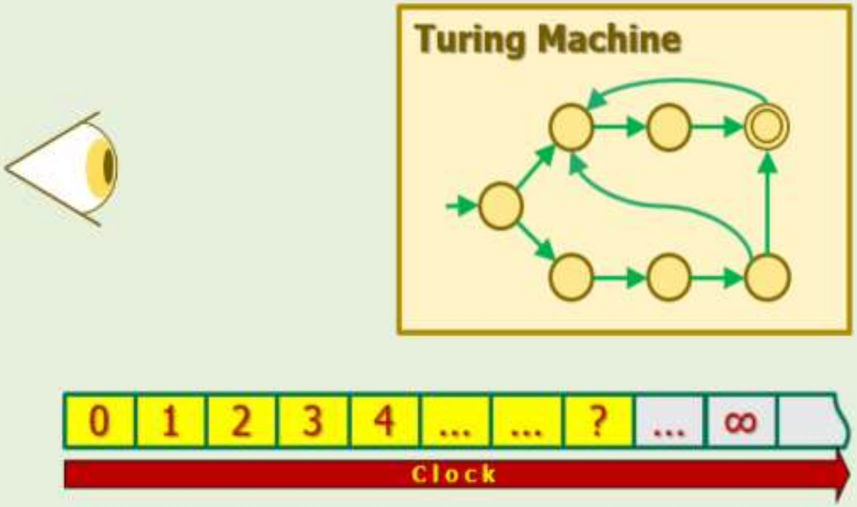
- Note that based on the rejection logic:

$$(\sim h \lor \sim f) \leftrightarrow \sim a$$

- If we can prove somehow that
  the machine falls into an infinite loop, then ...

- ... the string, that is being processed, is considered as rejected.
- ... because $\sim h \equiv$ True.

Another $1,000,000 Question

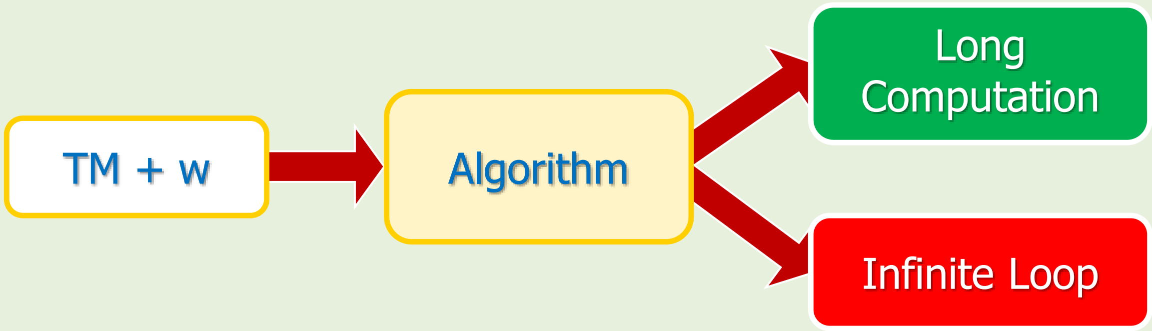


- An observer is looking at a TM that is working for a long time!

- How can the observer figure out whether ...
  ... it is in the middle of a very long computation?
  OR
  ... it is in an infinite loop,

Another $1,000,000 Question

- Let's formulate the question in computer science terminology!
- In fact, 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!

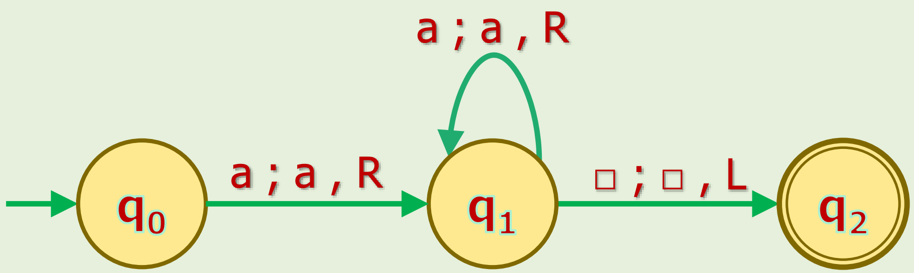
5. TMs in Action

Design Examples

TMs Design Examples

Example 6

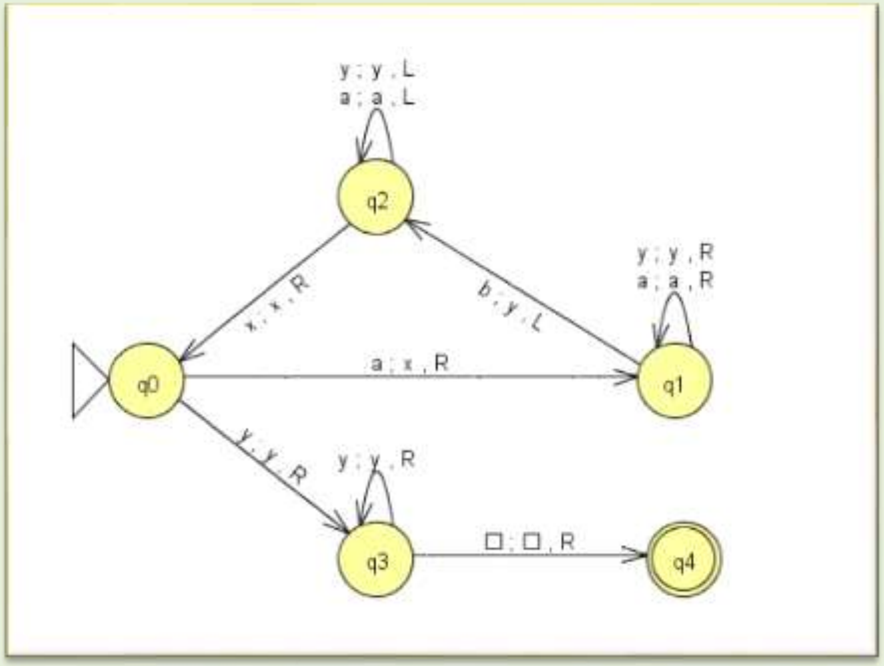
- Design a TM to accept $L = \{a^n : n \geq 1\}$ over $\Sigma = \{a, b\}$.
- Note that TMs usually don't like λ!



TMs Design Examples

Example 7

- Design a TM to accept our famous language $L = \{a^n b^n : n \geq 1\}$ over $\Sigma = \{a, b\}$.



Solution

- **Strategy**: For every a's, you should find one 'b'. So, we read the first 'a' and mark it as read by replacing it with 'x'. Then we go right to find a corresponding 'b' and mark it as 'y'.
  We continue this process until we don't have any a's.
  The string is accepted if there is no 'b' either.

Homework: TM Design

- Design a TM for the following languages:
  1. $L = \{w \in \{a, b\}^+\}$
  2. $L = \{w \in \{a, b\}^+ : |w| = 2k, K \geq 0\}$
  3. $L = \{w \in \{a, b\}^+ : |w| = 2k+1, K \geq 0\}$
  4. $L = \{1^{2k} : k \geq 1\}$ over $\Sigma = \{1\}$
  5. $L = \{w \in \{a, b\}^* : n_a(w) = n_b(w)\}$ //number of a's = number of b's
  6. $L = \{w \in \{a, b\}^+ : n_a(w) = n_b(w)\}$ //number of a's = number of b's
  7. $L = \{a^n b^n c^n : n \geq 1\}$
  8. $L = \{a^n b^m c^{nm} : n \geq 1, m \geq 1\}$
  9. $L = \{w\#w : w \in \{a, b\}^+\}$
  10. $L = \{w \in \{a, b\}^+ : |w| = 2k+1, K \geq 0, w$ contains at least one a$\}$
  11. $L = \{ww : w \in \{a, b\}^+\}$

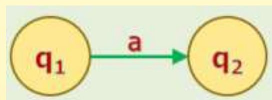
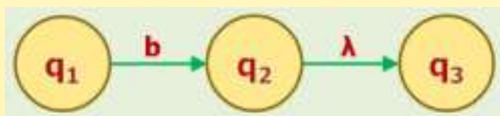
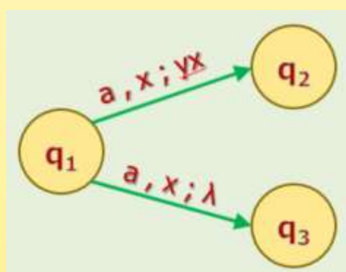
6. Definitions

Transition Function of TMs

- In this section, we are going to formally (mathematically) define the TMs.
- The important part of this definition, as usual, is the transition function.
- Because we are familiar with most other items of the definition.
- So, let's take some examples on transition functions.

  And try to figure out what the transition functions look like.

Transition Function: DFAs, NFAs, NPDAs, TMs

Class	Transition	Sub-Rule Example Transition Function
DFAs		$\delta\ (q_1, a) = q_2$ $\delta : Q \times \Sigma \rightarrow Q$
NFAs		$\delta\ (q_1, b) = \{q_2, q_3\}$ $\delta\ (q_2, a) = \{\ \}$ $\delta : Q \times \Sigma \rightarrow 2^Q$
NPDAs		$\delta\ (q_1\ , a\ , x) = \{(q_2\ , yx), (q_3\ , \lambda)\}$ $\delta: Q \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \rightarrow 2^{Q \times \Gamma^*}$
TMs		$\delta\ (q_1\ , a) = ???$ $\delta: ???$

TMs Transition Function Examples

Example 9

- Write the sub-rule of the following transition.

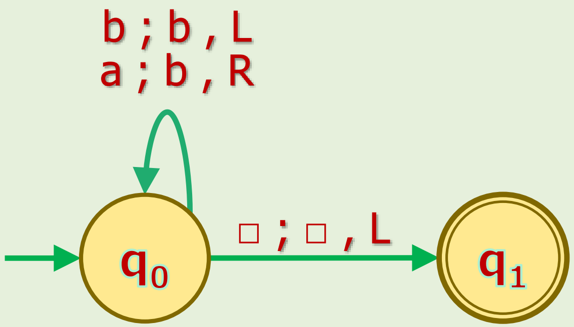
Solution

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

TMs Transition Function Examples

Example 10

- Write the δ of the following transition graph.

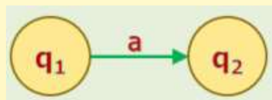
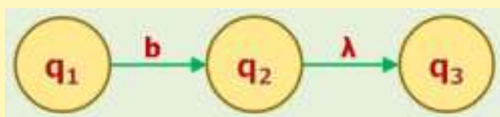
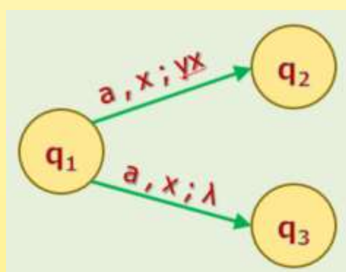


Solution

$$\delta: \begin{cases} \delta(q_0, a) = (q_0, b, R) \\ \delta(q_0, b) = (q_0, b, L) \\ \delta(q_0, \square) = (q_1, \square, L) \end{cases}$$

- Is the function total or partial?

Transition Function: DFAs, NFAs, NPDAs, TMs

Class	Transition	Sub-Rule Example Transition Function
DFAs		$\delta\,(q_1, a) = q_2$ $\delta : Q \times \Sigma \rightarrow Q$
NFAs		$\delta\,(q_1, b) = \{q_2, q_3\}$ $\delta\,(q_2, a) = \{\ \}$ $\delta : Q \times \Sigma \rightarrow 2^Q$
NPDAs		$\delta\,(q_1 , a , x) = \{(q_2 , yx), (q_3 , \lambda)\}$ $\delta: Q \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \rightarrow 2^{Q \times \Gamma^*}$
TMs		$\delta\,(q_1 , a) = (q_2 , b , R)$ $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$

6. Formal Definition of TMs

- A standard TM M is defined by the septuple (7-tuple):

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

- Where:
  - Q is a finite and nonempty set of states of the transition graph.
  - Σ is a finite and nonempty set of symbols called input alphabet.
  - Γ is a finite and nonempty set of symbols called tape alphabet.
  - δ is called transition function and is defined as:

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

δ can be total or partial function.

  - $q_0 \in Q$ is the initial state of the transition graph.
  - $\square \in \Gamma$ is a special symbol called blank.
  - $F \subseteq Q$ is the set of accepting states of the transition graph.

6. Formal Definition of TMs

$$M = (\overbrace{Q}^{\text{Set of States}} , \underbrace{\Sigma}_{\text{Input Alphabet}} , \overbrace{\Gamma}^{\text{Tape Alphabet}} , \underbrace{\delta}_{\text{Transition Function}} , \overbrace{q_0}^{\text{Initial State}} , \underbrace{\square}_{\text{Blank Symbol}} , \overbrace{F}^{\text{Set of Final States}})$$

6. Formal Definition of TMs: Notes

1. $\Sigma \subseteq \Gamma - \{\square\}$
   - The input string cannot contain blank symbol.
2. There is no relationship between determinism and δ being total function.

   The following table clearly depicts this fact.

Class	Transition Function Type	Type of Machine
DFAs	Total	Deterministic
NFAs	Total	Nondeterministic
NPDAs	Total	Nondeterministic
TMs	Partial or Total	Deterministic

7. TMs vs NPDAs

Can TMs Do Whatever NPDAs Can Do?

- Let's assume that we've constructed an NPDA for an arbitrary language L.
- Can we always construct a TM for L?

- Recall that to compare previous machines (i.e. DFAs, NFAS, NPDAs), we used the "formal definition conversion" technique .
- For this case, we cannot do that.
- But there is another technique called "simulation".

- So, we convert the above question to:

Can we simulate NPDAs operations by TMs?

- Yes! How?

Can TMs Do Whatever NPDAs Can Do?

- Let M be an NPDA for the language L.
- We want to simulate M by an equivalent TM called M' such that:

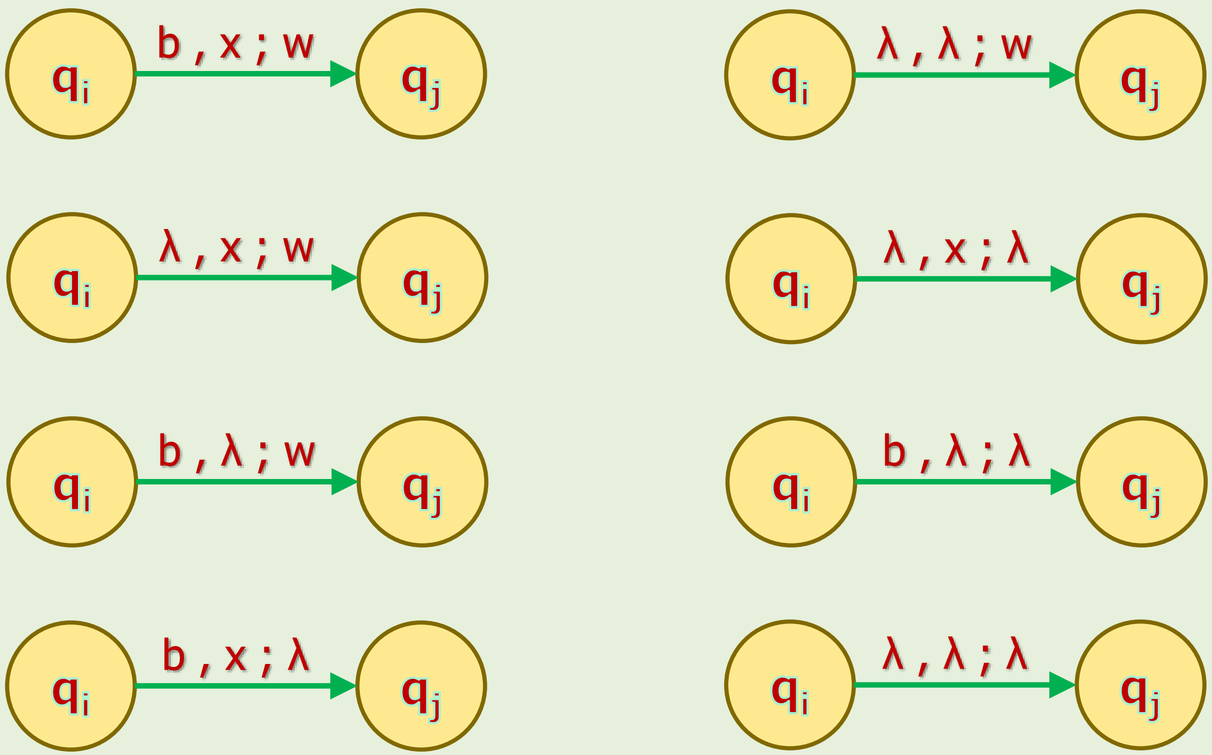
$$L(M) = L(M')$$

- M has some transitions and we should be able to simulate all of them by TM.

- Let's list all kind of transitions that an NPDA can have.
- If we can simulate them by TMs, then we'd be able to simulate any NPDAs by TMs

Can TMs Do Whatever NPDAs Can Do?

NPDAs All Possible Transitions



Can TMs Do Whatever NPDAs Can Do?

- We just show the simulation of one transition.
- And we leave the rest for the readers as exercise.

- I put the following file in Canvas for your reference:

Canvas → Files → Misc

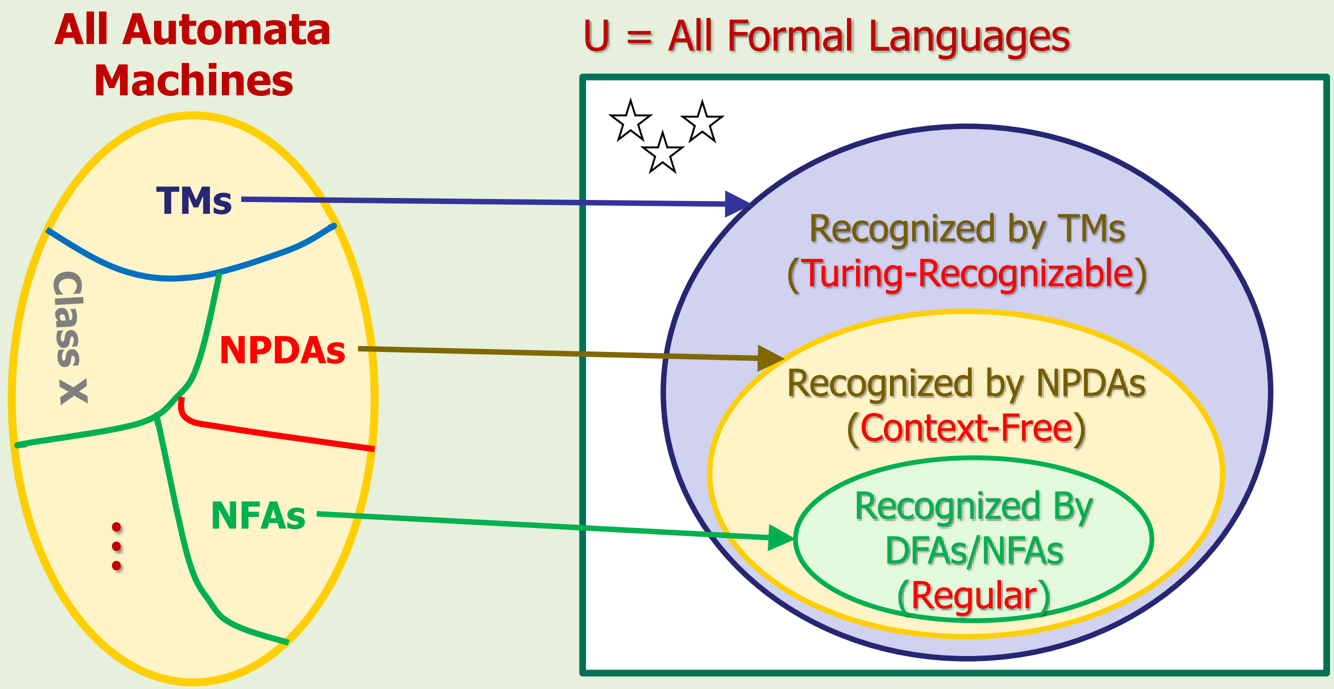
CS154-Ahmad Y-NPDAs-Transition-Simulation.pdf

- That's good experience for your term project too.

Can NPDAs Do Whatever TMs Can Do?

- Let's assume that we've constructed a TM for an arbitrary language L.
- Can we always construct an NPDA for L?
- No! Why?
- At least we know the following languages for which we can construct TMs but it is impossible to construct NPDAs.
  - $L = \{a^n b^n c^n : n \geq 1\}$
  - $L = \{ww : w \in \Sigma^*\}$
- Let's summarize our knowledge and figure out what would be the next step.

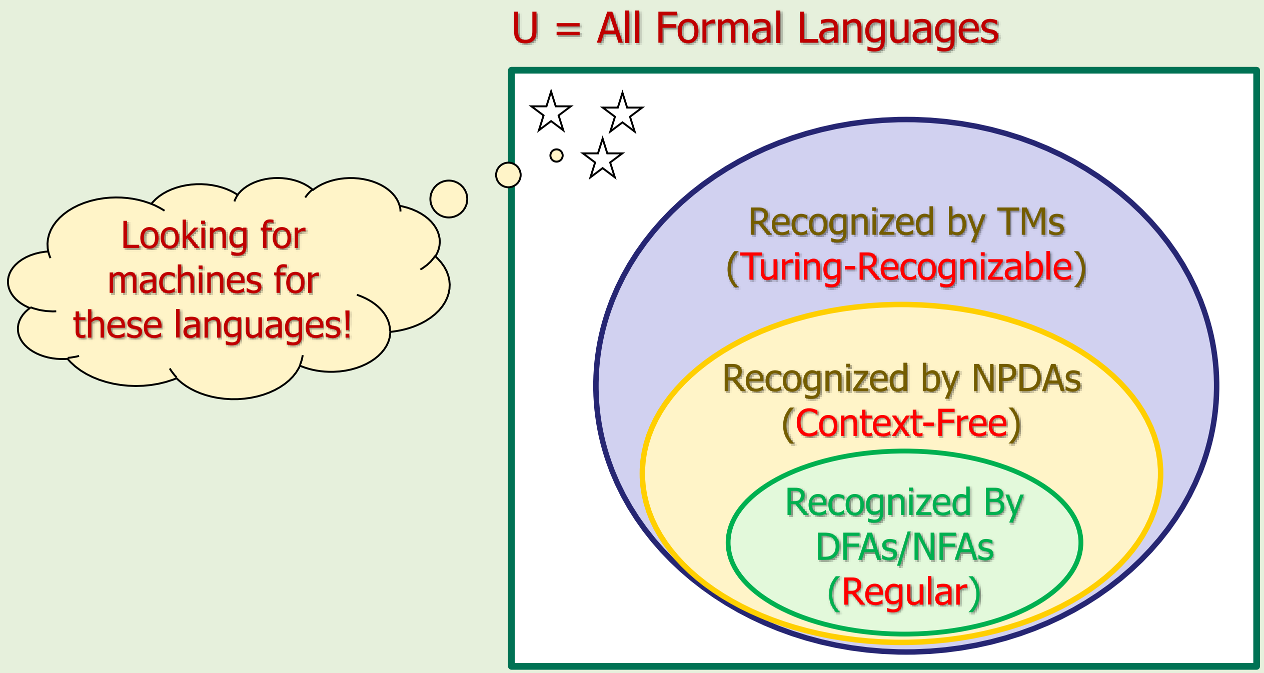
Machines and Languages Association



- The set of languages that NPDAs recognize is a proper subset of the set of languages that TMs recognize.
- So, TMs are more powerful than NPDAs.

8. What is the Next Step?

- TMs recognize some other non-regular languages called "Turing-recognizable".
- But there are still languages that are not Turing-recognizable!
- First, we need to find at least one of them, then we'll think about constructing a new class!



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
7. https://www.turing.org.uk/