

Turing Machines

Dr. Chuck Rocca
roccac@wcsu.edu

<http://sites.wcsu.edu/roccac>



Table of Contents

- 1 Machines
- 2 Turing Machines
- 3 Multitape Machines
- 4 Nondeterministic Machines
- 5 Enumerators
- 6 Algorithms
- 7 Next Class

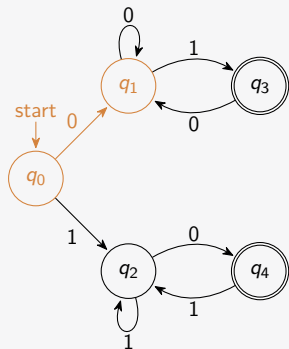
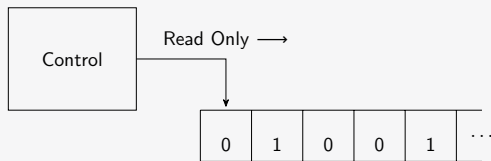


Table of Contents

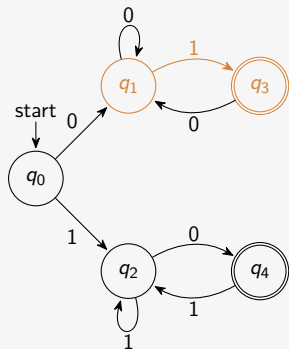
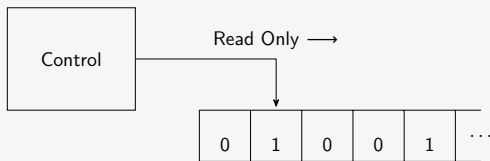
- 1 Machines
- 2 Turing Machines
- 3 Multitape Machines
- 4 Nondeterministic Machines
- 5 Enumerators
- 6 Algorithms
- 7 Next Class



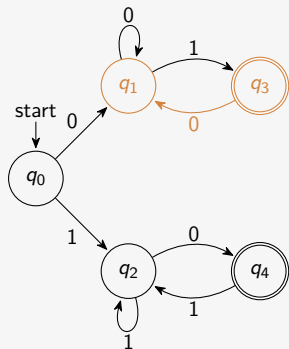
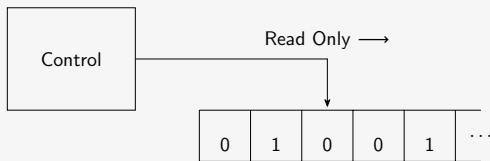
Finite Automata



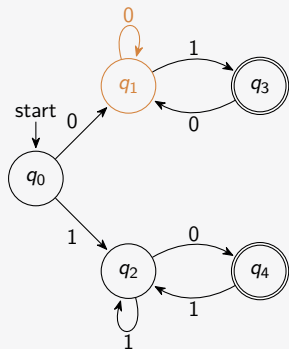
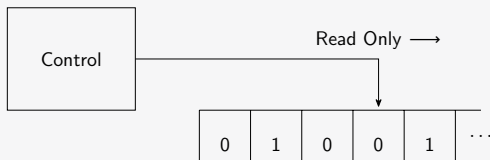
Finite Automata



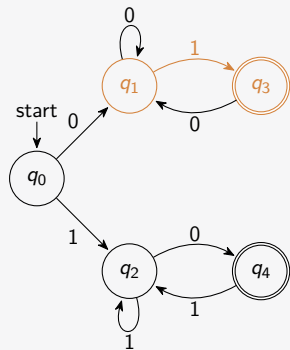
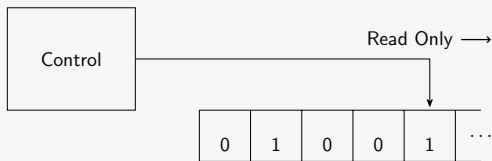
Finite Automata



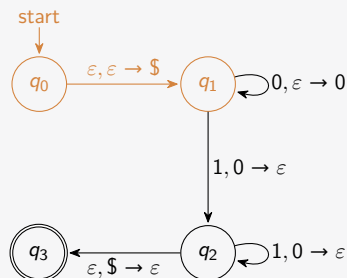
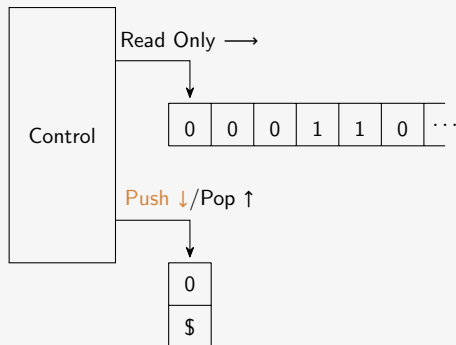
Finite Automata



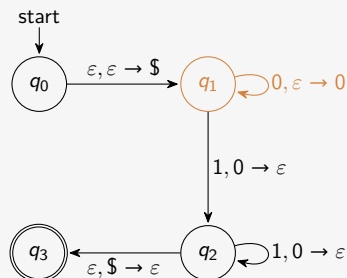
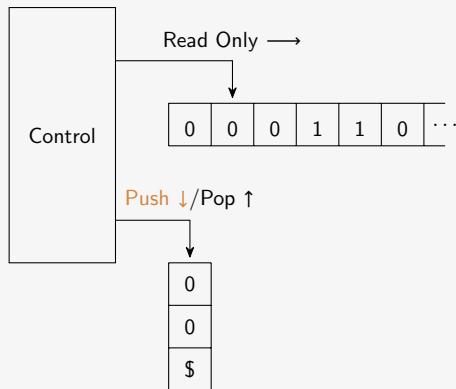
Finite Automata



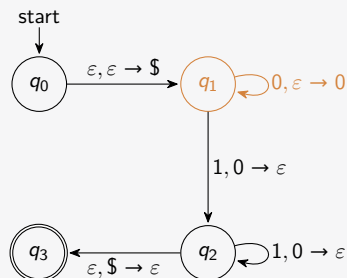
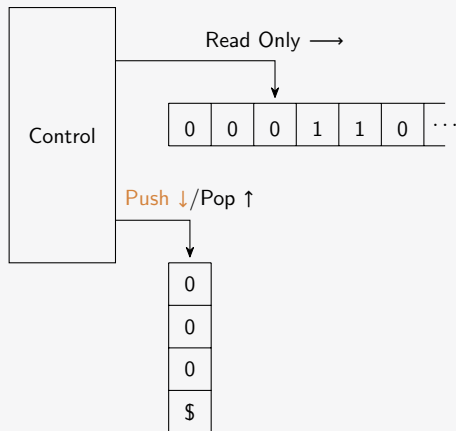
Pushdown Automata



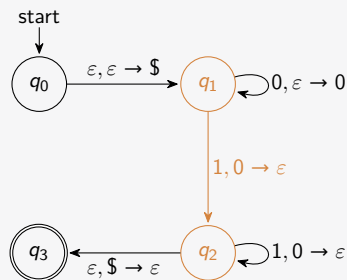
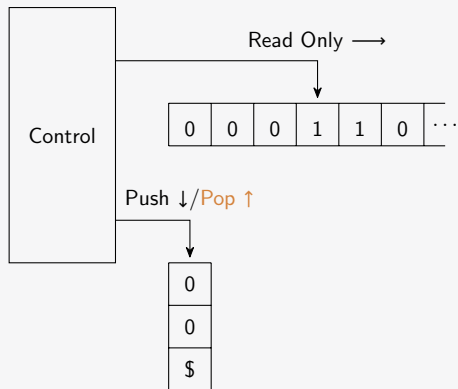
Pushdown Automata



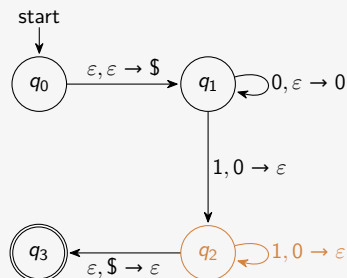
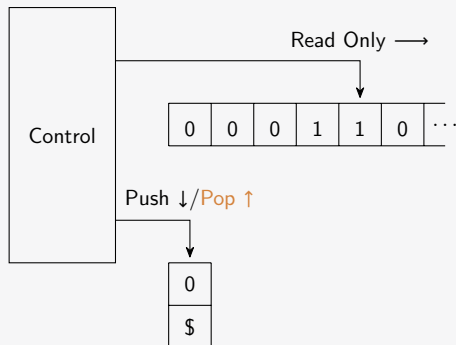
Pushdown Automata



Pushdown Automata



Pushdown Automata



Pushdown Automata

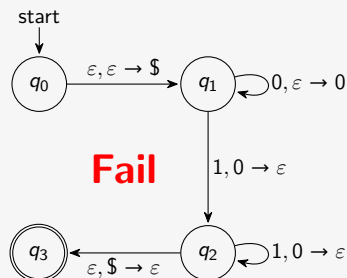
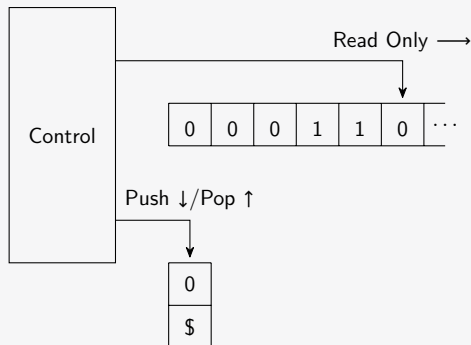
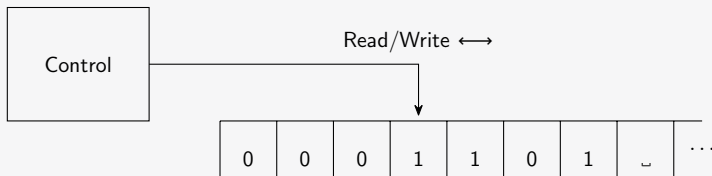


Table of Contents

- 1 Machines
- 2 Turing Machines**
- 3 Multitape Machines
- 4 Nondeterministic Machines
- 5 Enumerators
- 6 Algorithms
- 7 Next Class

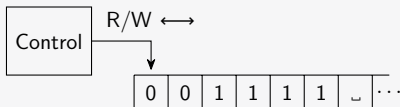


Turing Machine Example



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



● 001111

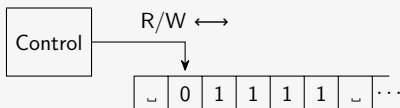
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



• 001111

• 01111

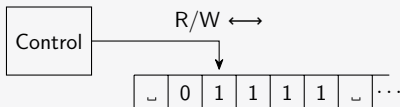
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



- 001111
- 01111

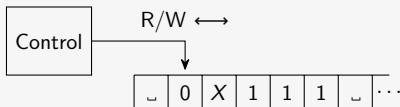
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



- 001111
- 01111
- 0X111

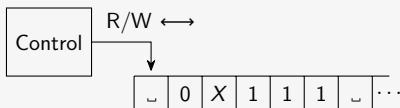
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



- 001111
- 001111
- 0X111

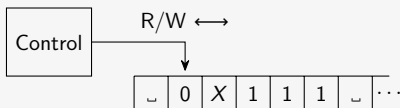
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



- 001111
- 01111
- 0X111

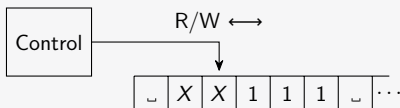
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



- 001111
- 001111
- 0X111
- XX111

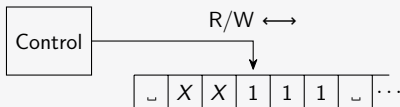
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



- 001111
- 001111
- 0X111
- XX111

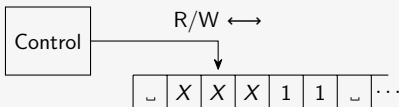
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



- 001111
- 001111
- 0X111
- XX111
- XXX11

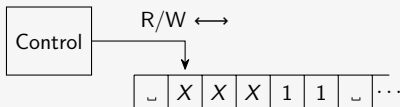
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



- 001111
- □01111
- □0X111
- □XX111
- □XXX11

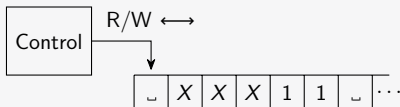
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



- 001111
- 001111
- 0X111
- XX111
- XXX11

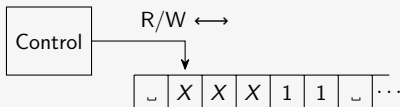
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



- 001111
- 001111
- 0X111
- XX111
- XXX11

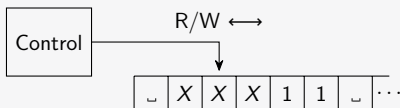
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



- 001111
- 001111
- 0X111
- XX111
- XXX11

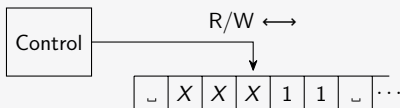
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



- 001111
- 001111
- 0X111
- XX111
- XXX11

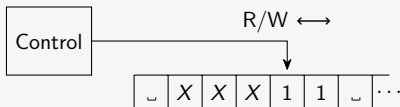
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



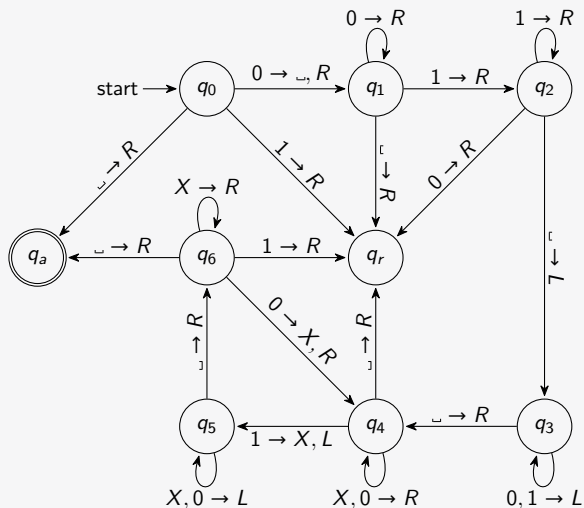
$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"

- 001111
- 001111
- 00X111
- XX111
- XXX11
- **Fail!**



Turing Machine Diagram



$$L = \{0^n 1^n \mid n \geq 0\}$$

- $X \rightarrow R$
- $X, \sqcup \rightarrow L$
- $0 \rightarrow X, R$



Formal Definitions

Definition (Turing Machine)

A **Turing machine** is a 7-tuple, $(Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$, where Q, Σ, Γ are finite sets and

- 1 Q is the set of states,
- 2 Σ is the input alphabet not containing \sqcup ,
- 3 Γ is the tape alphabet, where $\sqcup \in \Gamma$ and $\Sigma \subset \Gamma$,
- 4 $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ is the transition function,
- 5 q_0 is the start state,
- 6 q_a is the accept state, and
- 7 q_r is the reject state, where $q_r \neq q_a$.



Level of Description

- **Formal Description:** This spells out all seven components of the formal definition of the Turing machine.



Level of Description

- **Formal Description:** This spells out all seven components of the formal definition of the Turing machine.
- **Implementation Description:** Describes in words the way the Turing machine moves its head and stores data on the tape.

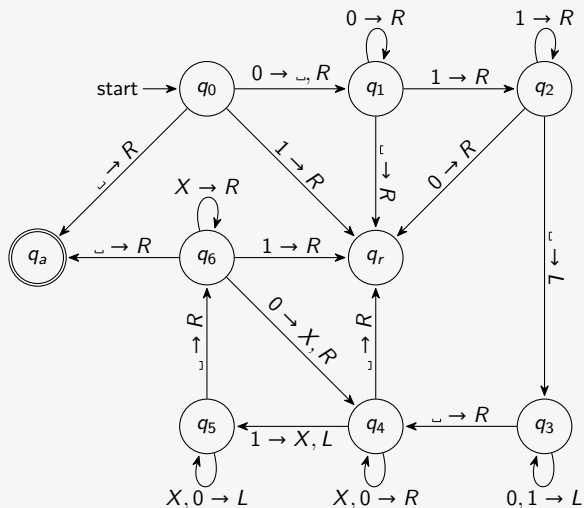


Level of Description

- **Formal Description:** This spells out all seven components of the formal definition of the Turing machine.
- **Implementation Description:** Describes in words the way the Turing machine moves its head and stores data on the tape.
- **High-Level Description:** Describes in words an algorithm for how the Turing machine recognizes a language ignoring implementation details.



Turing Machine Diagram



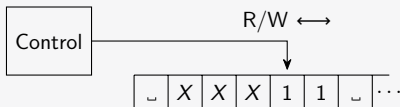
$$L = \{0^n 1^n \mid n \geq 0\}$$

- $X \rightarrow R$
- $X, \sqcup \rightarrow L$
- $0 \rightarrow X, R$



Turing Machine Example

Given $L = \{0^n 1^n \mid n \geq 0\}$, is $w \in L$?



$M =$ " On input string w :

- 1 Check for the correct $w = 0^* 1^*$ format.
- 2 Overwrite the first 0.
- 3 Scan right for a 1 to cross off; reject if none exist.
- 4 Move all the way back to the left.
- 5 Scan left to right for another 0,
 - if all 0's and 1's are gone, accept,
 - if no 0's are left but there are still 1's, reject,
 - if a 0 exists, cross it off and return to step 3"

- 001111
- 001111
- 0X111
- XX111
- XXX11
- Fail!

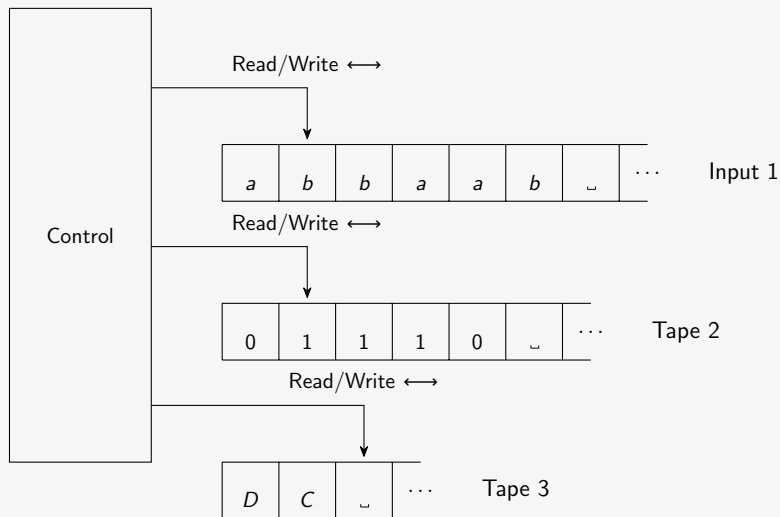


Table of Contents

- 1 Machines
- 2 Turing Machines
- 3 Multitape Machines**
- 4 Nondeterministic Machines
- 5 Enumerators
- 6 Algorithms
- 7 Next Class



Multitape Turing Machines



Multiple Tapes to A Single Tape

Theorem

A language is Turing recognizable if and only if some multiple tape Turing machine recognizes it.

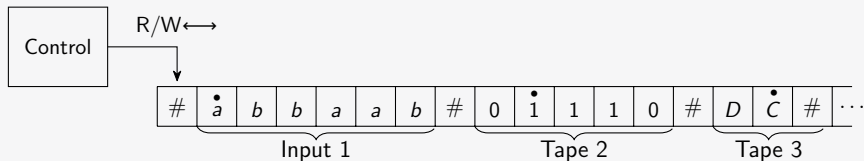
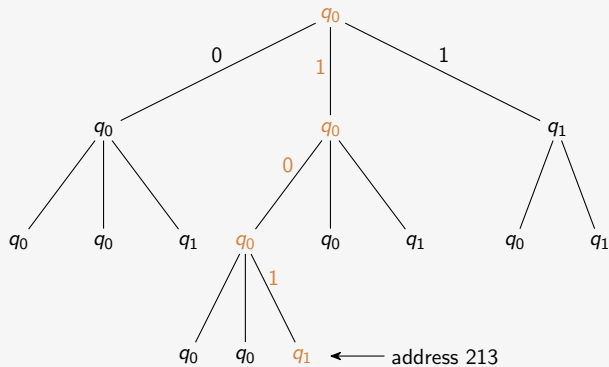
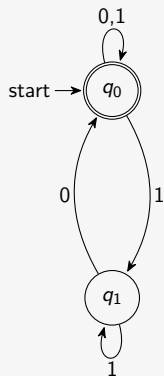


Table of Contents

- 1 Machines
- 2 Turing Machines
- 3 Multitape Machines
- 4 Nondeterministic Machines**
- 5 Enumerators
- 6 Algorithms
- 7 Next Class

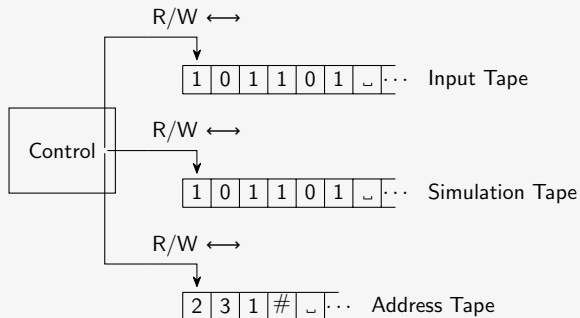


Nondeterministic Turing Machines



Simulating a Nondeterministic Turing Machine

Simulating a Nondeterministic T.M. N with a Deterministic One D :



Simulating a Nondeterministic Turing Machine

Simulating a Nondeterministic T.M. N with a Deterministic One D :

- 1 Place the input word w on tape 1, the input tape.



Simulating a Nondeterministic Turing Machine

Simulating a Nondeterministic T.M. N with a Deterministic One D :

- 1 Place the input word w on tape 1, the input tape.
- 2 Initialize tape 3, the address tape, to ϵ .



Simulating a Nondeterministic Turing Machine

Simulating a Nondeterministic T.M. N with a Deterministic One D :

- 1 Place the input word w on tape 1, the input tape.
- 2 Initialize tape 3, the address tape, to ε .
- 3 Copy tape 1 to tape 2, the simulation tape.



Simulating a Nondeterministic Turing Machine

Simulating a Nondeterministic T.M. N with a Deterministic One D :

- 1 Place the input word w on tape 1, the input tape.
- 2 Initialize tape 3, the address tape, to ε .
- 3 Copy tape 1 to tape 2, the simulation tape.
- 4 Simulate N using tape 2 following the address on tape 3.



Simulating a Nondeterministic Turing Machine

Simulating a Nondeterministic T.M. N with a Deterministic One D :

- 1 Place the input word w on tape 1, the input tape.
- 2 Initialize tape 3, the address tape, to ε .
- 3 Copy tape 1 to tape 2, the simulation tape.
- 4 Simulate N using tape 2 following the address on tape 3.
 - If you reach a fail state, the end of an address, or an invalid address, go to step 5.



Simulating a Nondeterministic Turing Machine

Simulating a Nondeterministic T.M. N with a Deterministic One D :

- 1 Place the input word w on tape 1, the input tape.
- 2 Initialize tape 3, the address tape, to ε .
- 3 Copy tape 1 to tape 2, the simulation tape.
- 4 Simulate N using tape 2 following the address on tape 3.
 - If you reach a fail state, the end of an address, or an invalid address, go to step 5.
 - If you reach an accept state, then accept.



Simulating a Nondeterministic Turing Machine

Simulating a Nondeterministic T.M. N with a Deterministic One D :

- 1 Place the input word w on tape 1, the input tape.
- 2 Initialize tape 3, the address tape, to ε .
- 3 Copy tape 1 to tape 2, the simulation tape.
- 4 Simulate N using tape 2 following the address on tape 3.
 - If you reach a fail state, the end of an address, or an invalid address, go to step 5.
 - If you reach an accept state, then accept.
- 5 Replace the address on tape 3 with the next address in line and go to step 3.

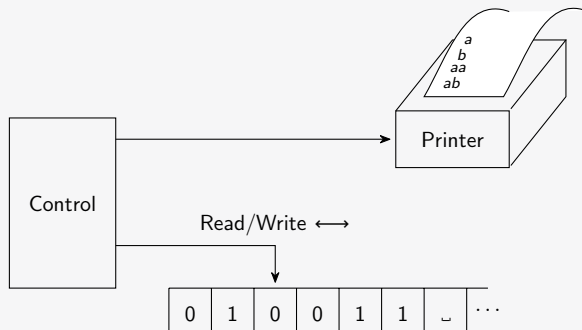


Table of Contents

- 1 Machines
- 2 Turing Machines
- 3 Multitape Machines
- 4 Nondeterministic Machines
- 5 Enumerators**
- 6 Algorithms
- 7 Next Class

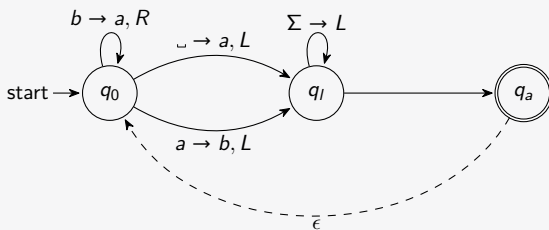


Enumerators



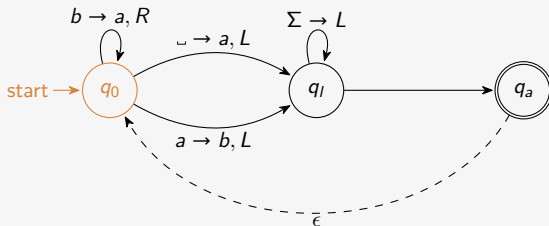
Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

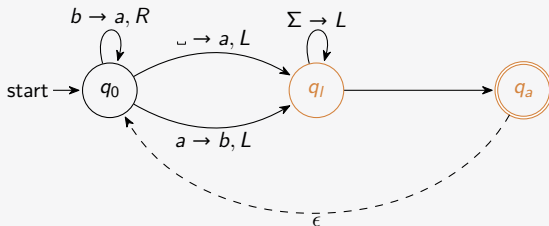


↓ \sqcup



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

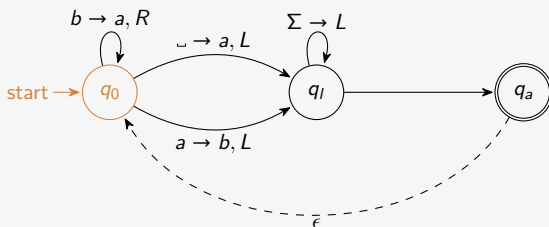


↓ **a**␣



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

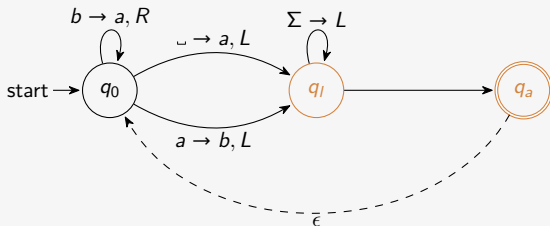


↓ **a**␣



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

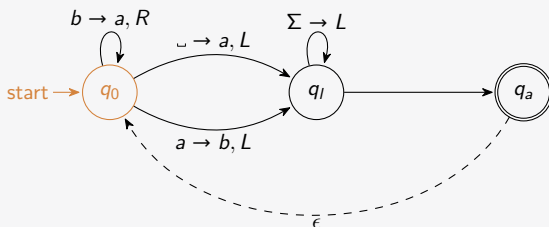


↓ **b**␣



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

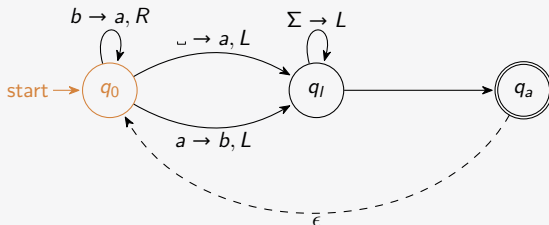


↓ **b**␣



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

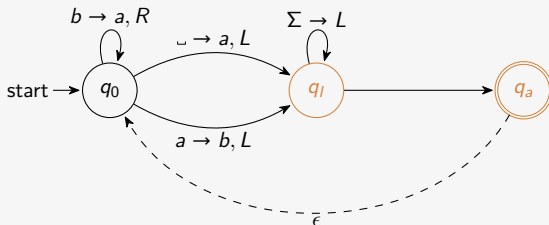


a ↓ \sqcup



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

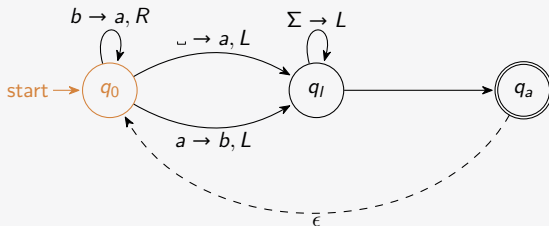


↓ **aa**␣



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

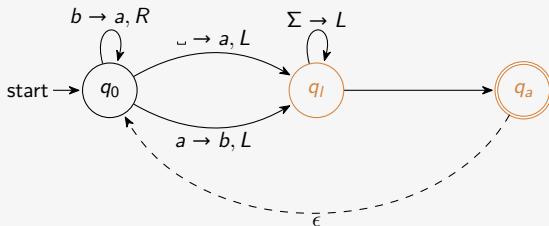


↓ **aa**_␣



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

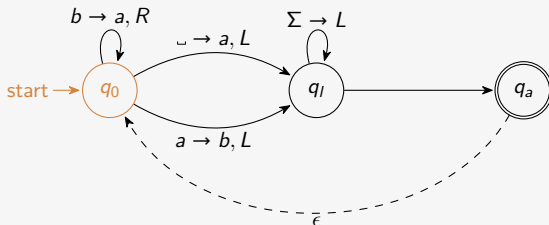


↓ **ba**_␣



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

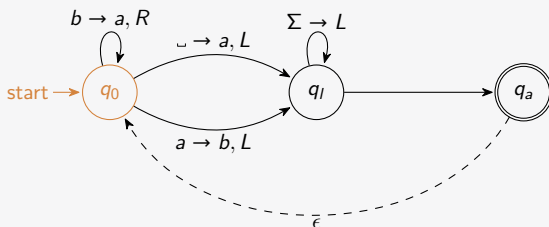


↓ **ba**_␣



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

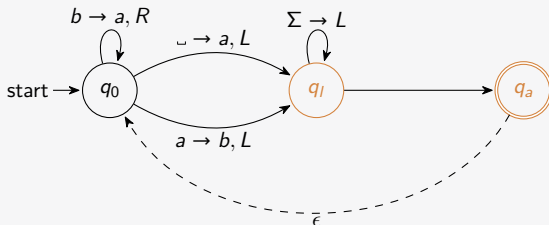


a ↓ a_␣



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

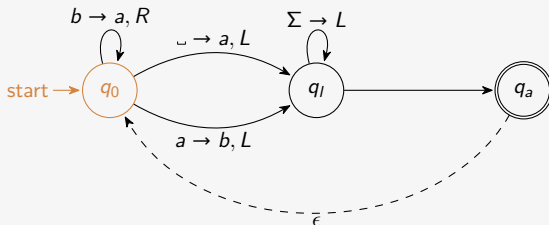


↓ **ab**_␣



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

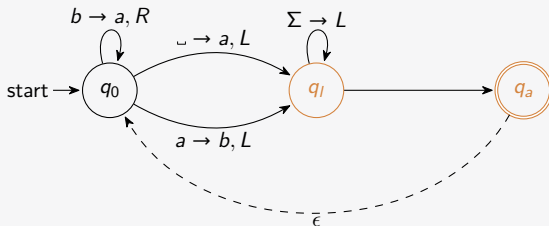


↓ **ab**_␣



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:

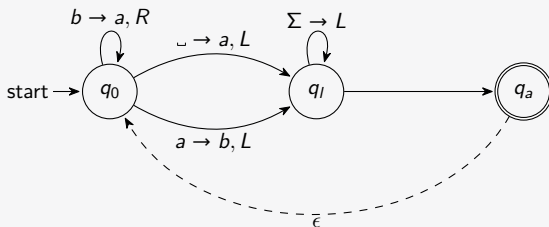


↓ **bb**␣



Sample Enumerator Diagram

The following produces $a, b, aa, ba, ab, bb, aaa, baa, aba, \dots$:



etc.



Equivalence of Enumerators and Turing Machines

Theorem

A language is Turing recognizable if and only if some enumerator enumerates it.

Proof.

- Define M : Given a word w , run the given enumerator and if w appears on its list, accept.



Equivalence of Enumerators and Turing Machines

Theorem

A language is Turing recognizable if and only if some enumerator enumerates it.

Proof.

- Define M : Given a word w , run the given enumerator and if w appears on its list, accept.
- Define E : Given a Turing machine, M , and a list of words in Σ^* , s_1, s_2, s_3, \dots , repeat the following for each $i = 1, 2, 3, \dots$:



Equivalence of Enumerators and Turing Machines

Theorem

A language is Turing recognizable if and only if some enumerator enumerates it.

Proof.

- Define M : Given a word w , run the given enumerator and if w appears on its list, accept.
- Define E : Given a Turing machine, M , and a list of words in Σ^* , s_1, s_2, s_3, \dots , repeat the following for each $i = 1, 2, 3, \dots$:
 - 1 For each $j \leq i$, run s_j for i steps through M ,



Equivalence of Enumerators and Turing Machines

Theorem

A language is Turing recognizable if and only if some enumerator enumerates it.

Proof.

- Define M : Given a word w , run the given enumerator and if w appears on its list, accept.
- Define E : Given a Turing machine, M , and a list of words in Σ^* , s_1, s_2, s_3, \dots , repeat the following for each $i = 1, 2, 3, \dots$:
 - ① For each $j \leq i$, run s_j for i steps through M ,
 - ② If any computation accepts, print the corresponding s_j .



Table of Contents

- 1 Machines
- 2 Turing Machines
- 3 Multitape Machines
- 4 Nondeterministic Machines
- 5 Enumerators
- 6 Algorithms**
- 7 Next Class



Algorithm

Definition

An *algorithm* is a computational process that is describable in terms of a Turing machine.



Connected Graphs Example

$M =$ "On input $\langle G \rangle$, the string encoding of a graph G :

- 0 Check that $\langle G \rangle$ is in the correct format
- 1 Select the first node in G and mark it.
- 2 Repeat the following until no new nodes are marked:
 - 3 For each node in G , mark it if it is attached by an edge to a node that is already marked.
- 4 Scan all the nodes of G , if they are all marked, *accept*; otherwise *reject*."



Decidable Unions

Given two *decidable* languages L_1 and L_2 and corresponding Turing machines M_1 and M_2 the union of the languages can be decided by:
M=“On input w :

- 0 Check that w is in the correct format.
- 1 Run M_1 on w . If it accepts, *accept*.
- 2 Run M_2 on w . If it accepts, *accept*.
- 3 Otherwise *reject*”



Recognizable Unions

Given two *Turing recognizable* languages L_1 and L_2 and corresponding Turing machines M_1 and M_2 the union of the languages can be recognized by:

$M =$ "On input w :

- 0 Check that w is in the correct format.
- 1 Run M_1 and M_2 alternately on w step by step. If either accepts, *accept*.
- 2 If both halt or reject, *reject*."



Decidable vs. Recognizable

Definition

A Language is *Turning-decidable* or simply *decidable* if some Turing machine decides it; the machine always reaches an accept or reject state. Given any word there is a TM that can tell if the word is or is not in the language.



Decidable vs. Recognizable

Definition

A Language is *Turning-decidable* or simply *decidable* if some Turing machine decides it; the machine always reaches an accept or reject state. Given any word there is a TM that can tell if the word is or is not in the language.

Definition

A Language is *Turning-recognizable* if some Turing machine recognizes it; in this case the machine reaches an accept state, reject state, or it may loop (fail to accept). There is a TM that accepts words in the language, but may fail to reach a verdict if a word is not in the language.



Decidable Intersections

Given two *decidable* languages L_1 and L_2 and corresponding Turing machines M_1 and M_2 the intersections of the languages can be decided by:
M=“On input w :

- 0 Check that w is in the correct format.
- 1 Run M_1 and M_2 on w . If they both accept, *accept*.
- 2 Otherwise *reject*”



Decidable Intersections

Given two *decidable* languages L_1 and L_2 and corresponding Turing machines M_1 and M_2 the intersections of the languages can be decided by:
M=“On input w :

- 0 Check that w is in the correct format.
- 1 Run M_1 and M_2 on w . If they both accept, *accept*.
- 2 Otherwise *reject*”

Why didn't we say “Run M_1 and M_2 alternately on w step by step?”



Decidable Complements

Given a *decidable* language L_1 and corresponding Turing machine M_1 the complement of the languages can be decided by:

$M =$ "On input w :

- 0 Check that w is in the correct format.
- 1 Run M_1 on w . If it accepts, *reject*.
- 2 Otherwise *accept*"



Table of Contents

- 1 Machines
- 2 Turing Machines
- 3 Multitape Machines
- 4 Nondeterministic Machines
- 5 Enumerators
- 6 Algorithms
- 7 Next Class**



Next Class

- Decidable Languages



Next Class

- Decidable Languages
- Decidable Problems and Regular Languages



Next Class

- Decidable Languages
- Decidable Problems and Regular Languages
- Decidable Problems and Context-Free Languages



Turing Machines

Dr. Chuck Rocca
roccac@wcsu.edu

<http://sites.wcsu.edu/roccac>

