CSCI 335 Theory of Computing

Lecture 13 – Turing Machines III: Variants of Turing Machines

DATE : December 04 2022

Learning Outcomes

Learning Outcomes

By the end of this section you will be able to:

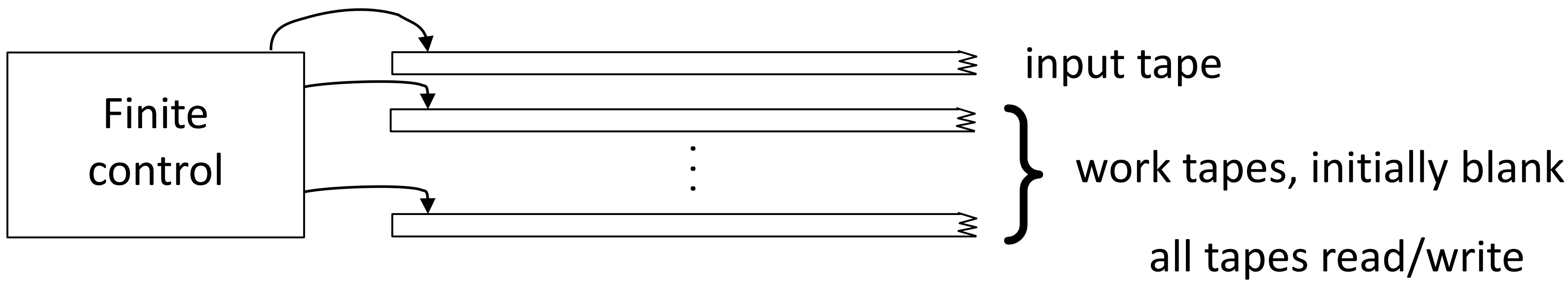
- Identify variants of Turing machines
- Analyze multi-tape Turing machines and show their equivalence to single-tape TMs
- Analyze nondeterministic Turing machines and show their equivalence to deterministic TMs
- Analyze enumerator-type Turing machines and show their equivalence to the conventional TMs

Outline

- Multi-tape Turing machines
- Nondeterministic Turing machines
- Enumerators

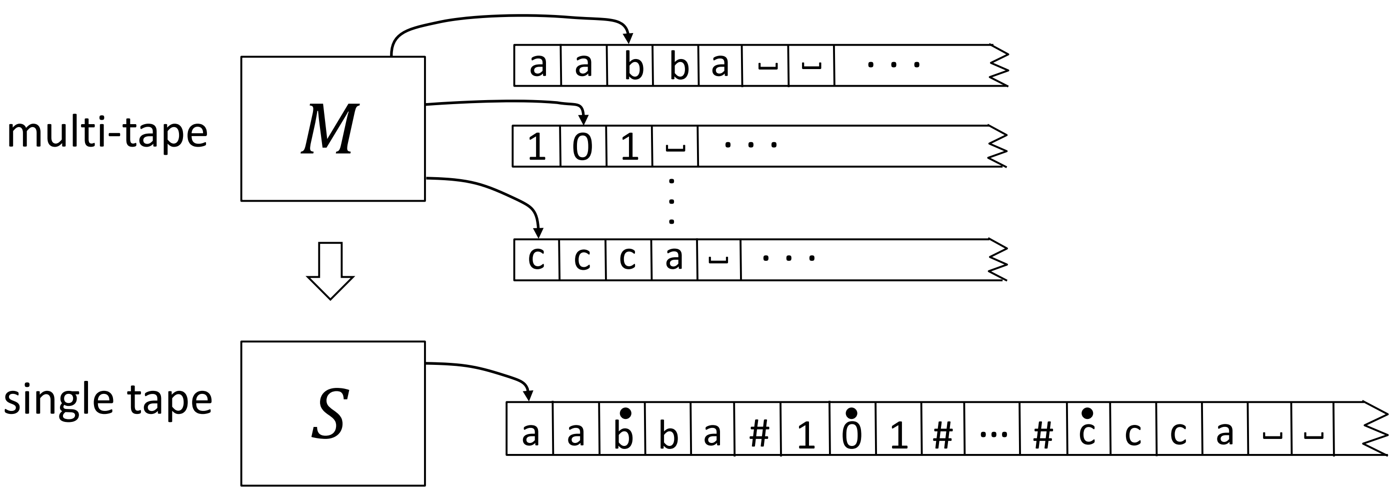
Multi-tape Turing machines

Multitape Turing machines



Theorem: A is T-recognizable iff some multi-tape TM recognizes A

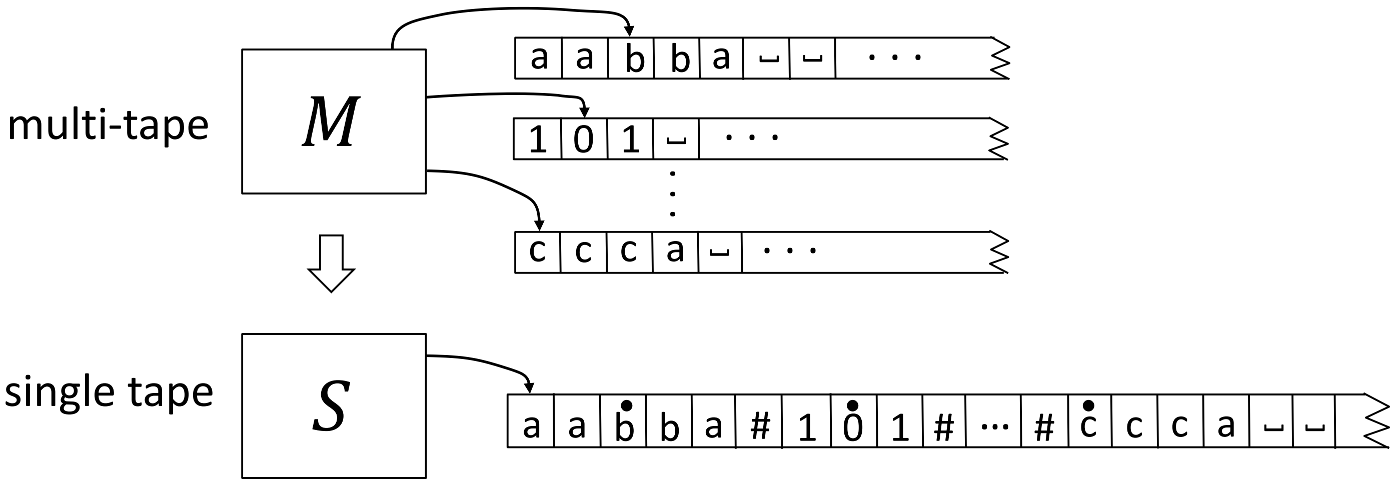
Proof: ($\rightarrow$) immediate (S can be simulated by an M restricted to a single tape)



Multitape Turing machines

Theorem: A is T-recognizable iff some multi-tape TM recognizes A

Proof: (←) Convert the multi-tape TM to a single-tape TM:



S simulates M by storing the contents of multiple tapes on a single tape in "blocks". Record head positions with dotted symbols.

Some details of S:

1) To simulate each of M's steps
   a. Scan entire tape to find dotted symbols.
   b. Scan again to update according to M's δ.
   c. Shift to add room as needed.
2) Accept/reject if M does.

Multitape Turing machines

$S =$ "On input $w = w_1 \cdots w_n$:

1. First S puts its tape into the format that represents all k tapes of M. The formatted tape contains

$$\#\dot{w}_1 w_2 \cdots w_n \#\dot{\sqcup}\#\dot{\sqcup}\# \cdots \#.$$

2. To simulate a single move, S scans its tape from the first #, which marks the left-hand end, to the $(k+1)$st #, which marks the right-hand end, in order to determine the symbols under the virtual heads. Then S makes a second pass to update the tapes according to the way that M's transition function dictates.
3. If at any point S moves one of the virtual heads to the right onto a #, this action signifies that M has moved the corresponding head onto the previously unread blank portion of that tape. So S writes a blank symbol on this tape cell and shifts the tape contents, from this cell until the rightmost #, one unit to the right. Then it continues the simulation as before."

Implementation-level description of a single-tape TM simulating a multi-tape TM

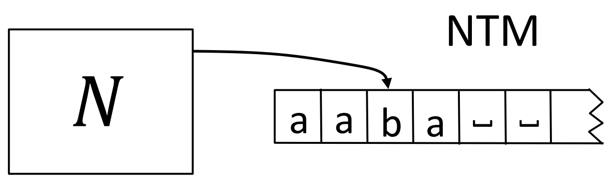
Nondeterministic Turing machines

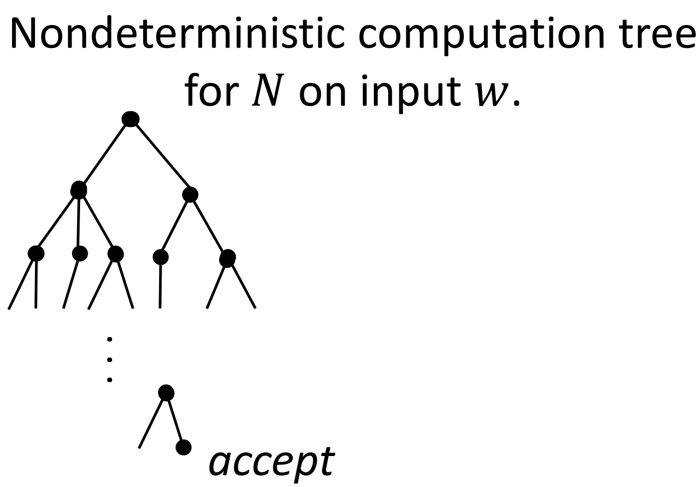
Nondeterministic Turing machines

A <u>Nondeterministic TM</u> (NTM) is similar to a Deterministic TM except for its transition function $\delta\colon Q \times \Gamma \to \mathcal{P}(\,Q \times \Gamma \times \{L, R\}\,)$.

Theorem: A is T-recognizable iff some NTM recognizes A

Proof: $(\rightarrow)$ immediate (A deterministic TM is a special case of a NTM).





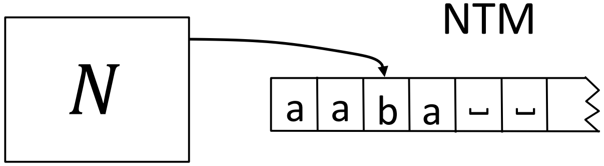
Nondeterministic Turing machines

A <u>Nondeterministic TM</u> (NTM) is similar to a Deterministic TM except for its transition function δ: $Q \times \Gamma \rightarrow \mathcal{P}(\, Q \times \Gamma \times \{L, R\}\,)$.

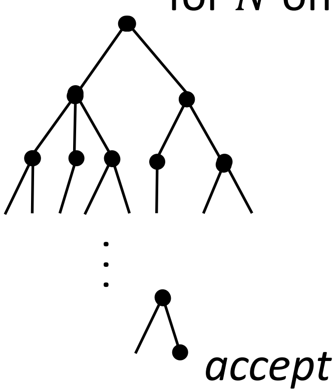
Theorem: A is T-recognizable iff some NTM recognizes A

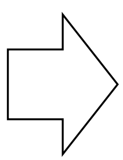
Proof: $(\leftarrow)$ Convert NTM to a Deterministic multi-tape TM…

NTM



Nondeterministic computation tree for N on input w.





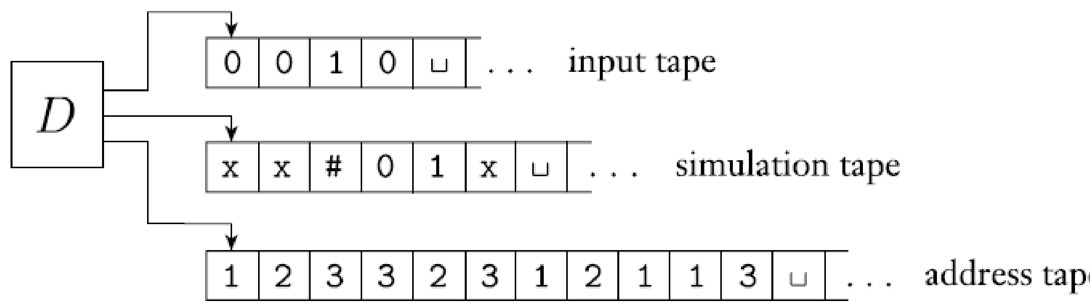


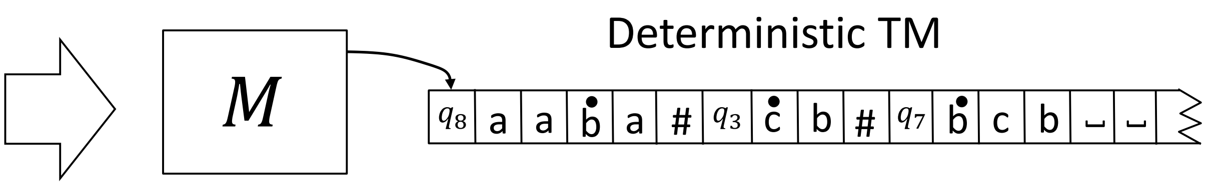
FIGURE 3.17
Deterministic TM D simulating nondeterministic TM N

Nondeterministic Turing machines

A <u>Nondeterministic TM</u> (NTM) is similar to a Deterministic TM except for its transition function $\delta\colon \mathrm{Q} \times \Gamma \to \mathcal{P}(\, Q \times \Gamma \times \{\mathrm{L, R}\}\,)$.

Theorem: A is T-recognizable iff some NTM recognizes A

Proof: $(\leftarrow)$ ...and then convert the multi-tape TM to a single-tape TM



M simulates N by storing each thread's tape in a separate "block" on its tape.
Also need to store the head location,
and the state for each thread, in the block.

If a thread forks, then M copies the block.

If a thread accepts then M accepts.

Nondeterministic Turing machines

COROLLARY 3.18

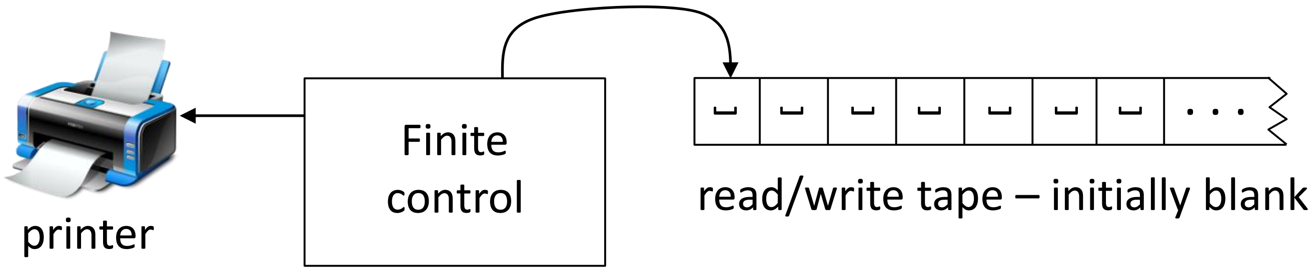
A language is Turing-recognizable if and only if some nondeterministic Turing machine recognizes it.

COROLLARY 3.19

A language is decidable if and only if some nondeterministic Turing machine decides it.

Enumerators

Enumerators



Defn: A <u>Turing Enumerator</u> is a deterministic TM with a printer.

It starts on a blank tape and it can print strings $w_1, w_2, w_3, \ldots$ possibly going forever.

Its language is the set of all strings it prints. **It is a generator, not a recognizer.**

For enumerator E we say $L(E) = \{w \mid E \text{ prints } w\}$.

Theorem: A is T-recognizable iff $A = L(E)$ for some T-enumerator E.

Proof: ($\leftarrow$) **Convert E to equivalent TM M.**

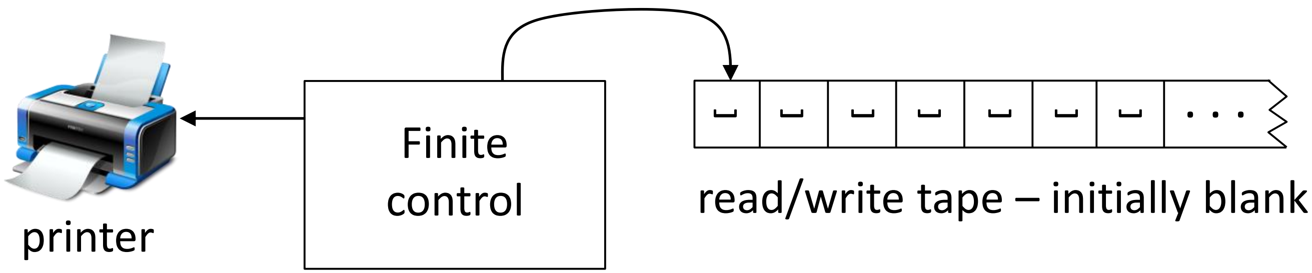
$M =$ for input w:

Simulate E (on blank input).

Whenever E prints x, test $x = w$.

Accept if $=$ and continue otherwise.

Enumerators



Defn: A <u>Turing Enumerator</u> is a deterministic TM with a printer.

It starts on a blank tape and it can print strings $w_1, w_2, w_3, \ldots$ possibly going forever.

Its language is the set of all strings it prints. **It is a generator, not a recognizer.**

For enumerator E we say $L(E) = \{w \mid E \text{ prints } w\}$.

Theorem: A is T-recognizable iff $A = L(E)$ for some T-enumerator E.

Proof: (→) **Convert TM M to equivalent enumerator E.**

$E =$ Simulate M on each w_i in $\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, \ldots\}$

If M accepts w_i then print w_i .

Continue with next w_i .

Problem: What if M on w_i loops?

Fix: Simulate M on $w_1, w_2, \ldots, w_i$ for i steps, for $i = 1, 2, \ldots$

Print those w_i which are accepted.

Summary

Summary

- Variants of Turing machines:
  - Multi-tape TMs
  - Nondeterministic TMs
  - Enumerators

Readings

- Variants of Turing machines: Section 3.2 (Sipser 2013)