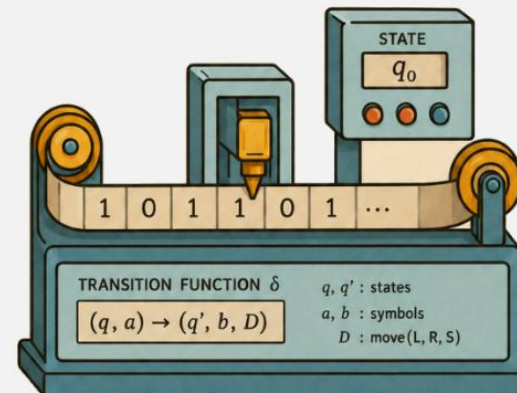


BU CS 332 – Theory of Computation

Lecture 11:

- Closure Properties
- Nondeterminism
- Church-Turing Thesis



Alexander Poremba
October 15, 2026



Why Are Multi-Tape TMs Useful?

Closure Properties

Closure Properties

Fact: The Turing-recognizable languages are closed under the union operation:

↙ $TM\ M_1$ ↘ $TM\ M_2$
If L_1 and L_2 are both Turing-recognizable, then $L_1 \cup L_2$

is also Turing-recognizable.



First attempt:

Construct a multi-tape machine as follows:

On input $w = w_1 w_2 \dots w_n$:

1. Copy w from tape #1 to tape #2.

2. Run M_1 on tape #1.
If M_1 accepts, **accept**.

3. Run M_2 on tape #2.
If M_2 accepts, **accept**.

4. Else, **reject**.

ISSUE:

This machine might run forever, for example when $w \notin L_1$ and M_1 loops but $w \in L_2$.

Closure Properties

Fact: The Turing-recognizable languages are closed under the union operation:

If L_1 and L_2 are both Turing-recognizable, then $L_1 \cup L_2$ is also Turing-recognizable.

On input w :

1. Scan tapes 1,2,3 left-to-right to copy w to tapes 2 and 3.
 2. Repeat forever:
 - a) Run M_1 for one step on tape 2.
 - b) Run M_2 for one step on tape 3.
 - c) If either machine accepts, **accept**.
 3. If both machines reject, **reject**.
- If $w \in L_1 \cup L_2$, either M_1 or M_2 accepts after finite no. of steps. ✓
 - If $w \notin L_1 \cup L_2$, then neither machine accepts. ✓

Closure Properties

The Turing-decidable languages are closed under:

- Union
- Concatenation
- Star
- Intersection
- Reverse
- Complement

The Turing-recognizable languages are closed under:

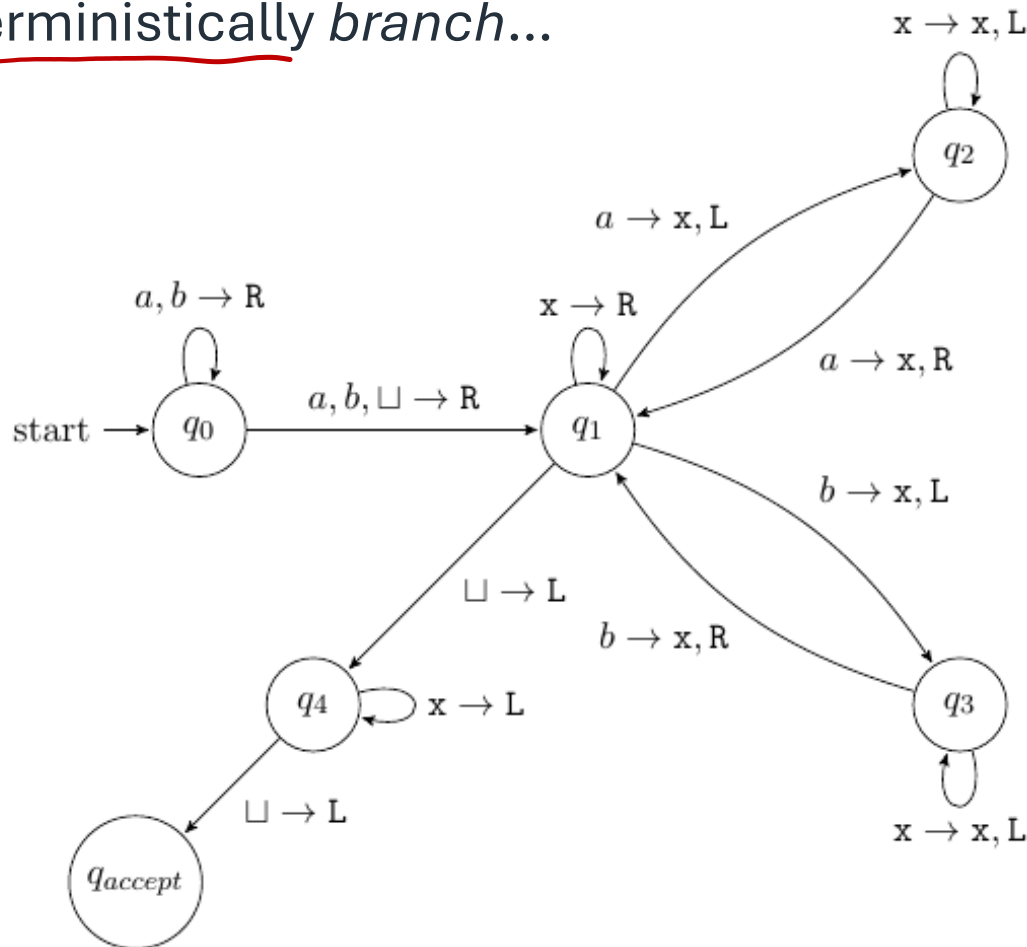
- Union
- Concatenation
- Star
- Intersection
- Reverse

TMs are equivalent to...

- TMs with “stay put”
- TMs with 2-way infinite tapes
- Multi-tape TMs
- Nondeterministic TMs today!
- Random access TMs
- Enumerators
- Finite automata with access to an unbounded queue
- Primitive recursive functions
- Cellular automata
- ...

Nondeterministic TMs

At any point in the computation, the machine may nondeterministically branch...



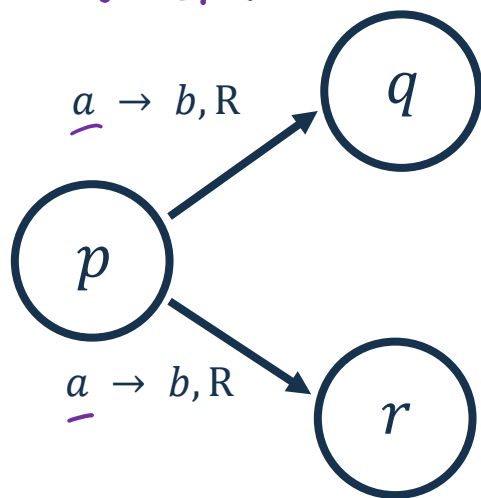
Nondeterministic TMs

Transition function $\delta : Q \times \Gamma \rightarrow P(Q \times \Gamma \times \{L, R, S\})$

$\uparrow$ $\uparrow$ $\underbrace{\hspace{10em}}$
current state current symbol Set of possibilities
(next state, new symbol, next move)

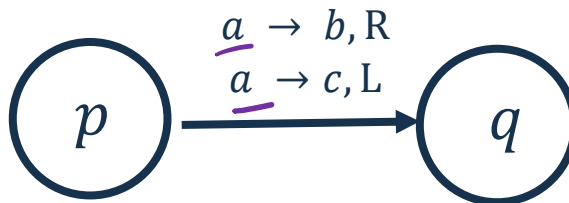
Case I

Multiple possible next states



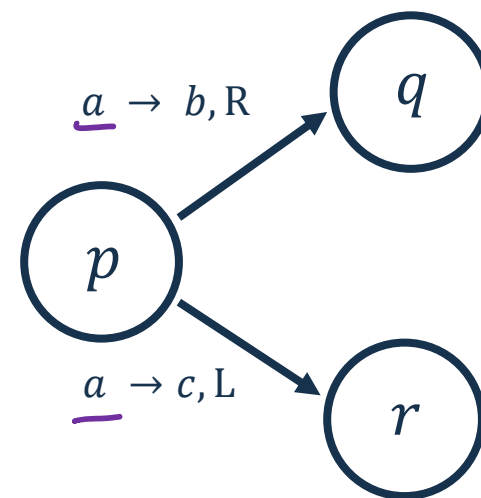
Case II

Multiple read/write and move instructions

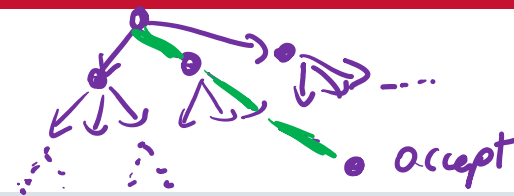


Case III

Both



Nondeterministic TMs



An NTM N **accepts** input w if when run on w it accepts on at least one computational branch

$$L(N) = \{w \mid N \text{ accepts input } w\}$$

- $w \in L(N) \Rightarrow$ there exists a branch of N 's computation path leading it to accept the input w .
- $w \notin L(N) \Rightarrow$ all other branches of N 's computation lead it to reject, run forever (loop), or fail to reach any state on w .

An NTM N is a **decider** if on **every** input, it halts on every computational branch !

- $w \in L(N) \Rightarrow$ there exists a branch of N 's computation which leads it to accept on input w .
(other branches may either accept or reject)
- $w \notin L(N) \Rightarrow$ all branches of N 's computation path lead it to reject on input w .

Nondeterministic TMs

At any point in computation, it may **nondeterministically** branch; accepts iff there is an accepting computation path

Implementation-Level Description: On input string w : $\Sigma = \{a, b\}$

1) Scan tape left-to-right. At some point during this scan, **nondeterministically** go to step 2.

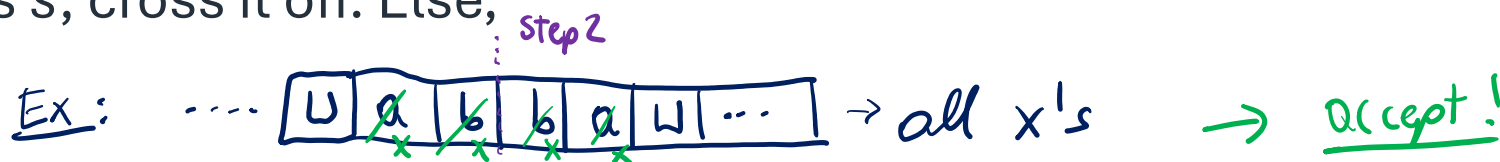
2a) Read the next symbol s and cross it off.

2b) Move the head left repeatedly until a non- x symbol is found. If it matches s , cross it off. Else, **reject**.

2c) Move the head right until a non- x symbol is found. If blank is hit, go to step 3.

2d) Go back to 2a).

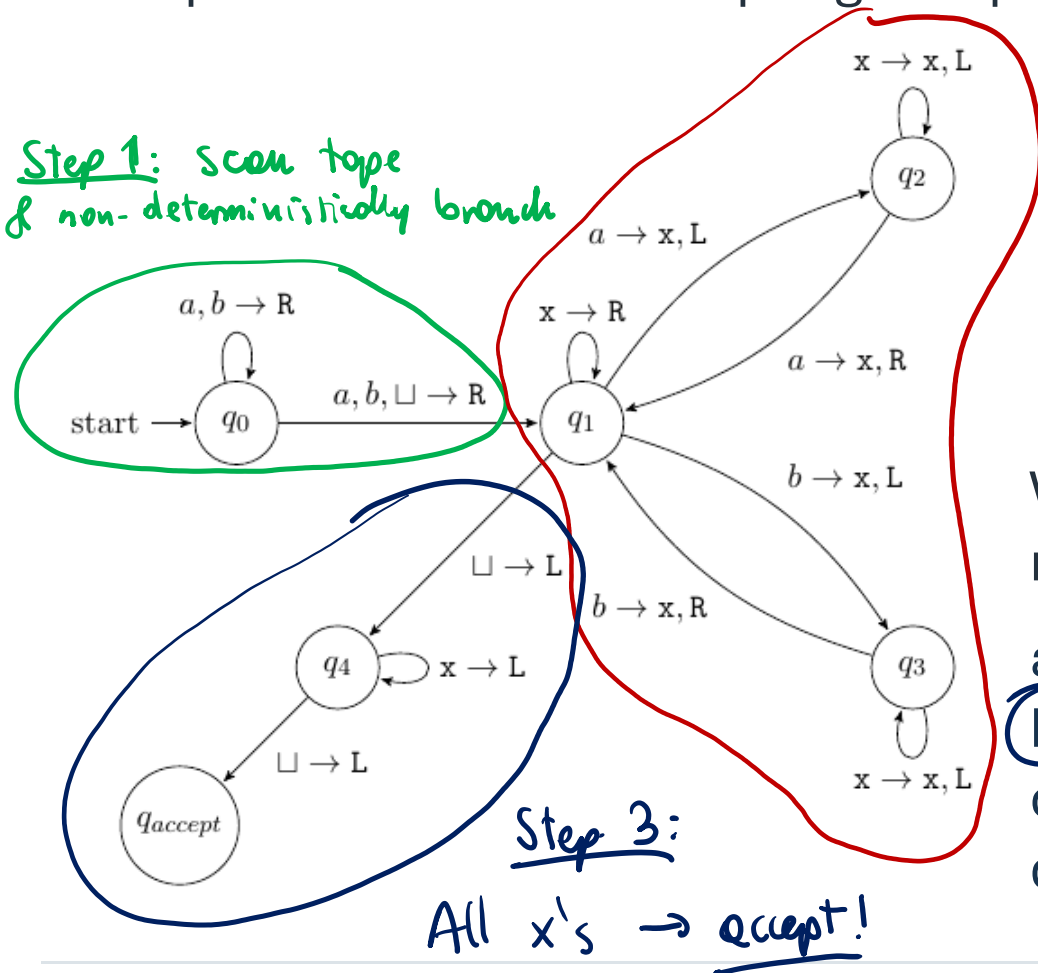
3) Check that the entire tape consists of x 's. If so, **accept**. Else, **reject**.



Question: Nondeterministic TMs

At any point in computation, it may **nondeterministically** branch; accepts iff there is an accepting computation path

Step 1: scan tape & non-deterministically branch



Step 2:

cross off matching symbols



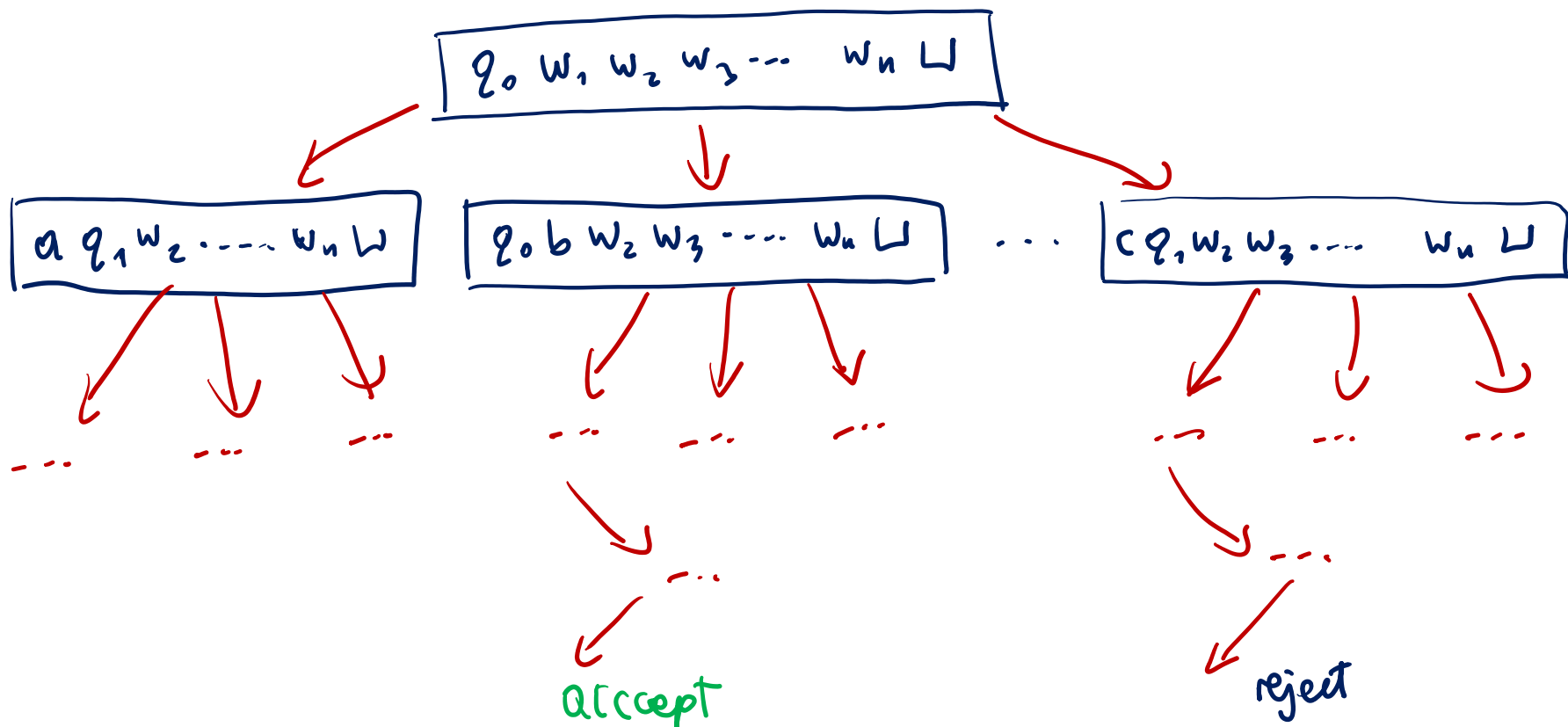
What is the language recognized by this NTM?

- a) $\{ ww \mid w \in \{a, b\}^* \}$
- b) $\{ ww^R \mid w \in \{a, b\}^* \}$**
- c) $\{ ww \mid w \in \{a, b, x\}^* \}$
- d) $\{ wx^n w^R \mid w \in \{a, b\}^*, n \geq 0 \}$

Nondeterministic TMs

Theorem: Every nondeterministic TM can be simulated by an equivalent deterministic TM

Proof idea: Explore “tree of possible computations”



Question: Simulating NTMs

Which of the following algorithms is always appropriate for searching the tree of possible computations for an accepting configuration?

- a) **Depth-first search:** Explore as far as possible down each branch before backtracking
- b) **Breadth-first search:** Explore all configurations at depth 1, then all configurations at depth 2, etc.
- c) **Both** algorithms will always work



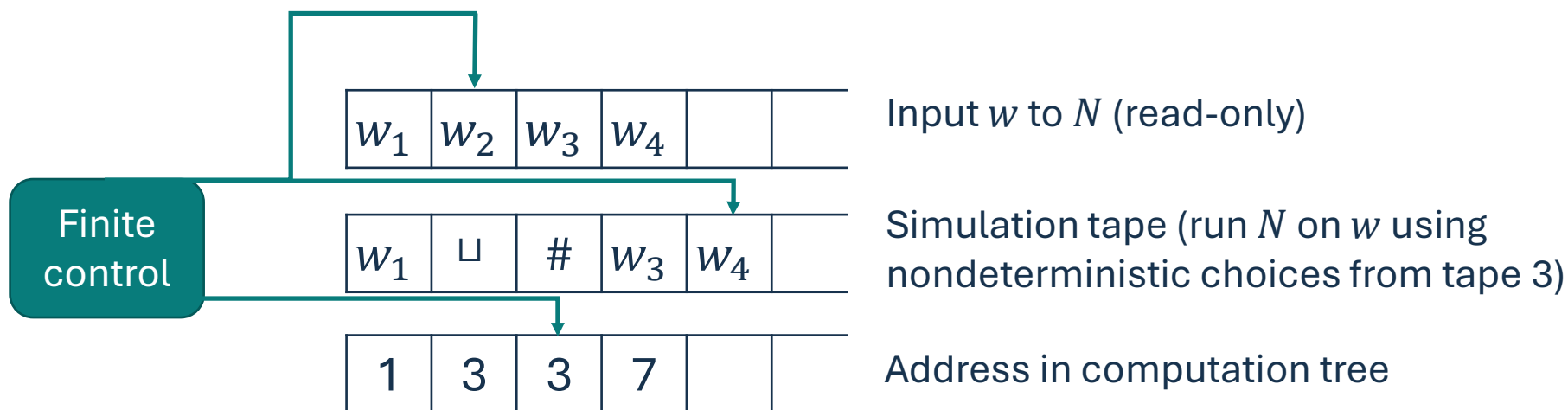
If the original NTM is a decider
then both of these work!



Nondeterministic TMs

Theorem: Every nondeterministic TM has an equivalent deterministic TM

Proof idea: Simulate an NTM N using a 3-tape TM
(See Sipser for full description)



TMs are equivalent to...

- TMs with “stay put”
- TMs with 2-way infinite tapes
- Multi-tape TMs
- Nondeterministic TMs
- Random access TMs
- Enumerators
- Finite automata with access to an unbounded queue
- Primitive recursive functions
- Cellular automata
- ...

Church-Turing Thesis

The equivalence of these models is a **mathematical theorem** (you can prove that each can simulate another)

Church-Turing Thesis v1: The basic TM (hence all of these models) captures our intuitive notion of algorithms

Church-Turing Thesis v2: Any physically realizable model of computation can be simulated by the basic TM

The Church-Turing Thesis is **not** a mathematical statement! Can't be mathematically proved