Turing Machines and Their Variants

From finite control to general-purpose computation.

Formal model • Worked tape traces • Equivalent variants

Learning outcomes

Model

Define a Turing machine (TM) and interpret its transition function.

Construction

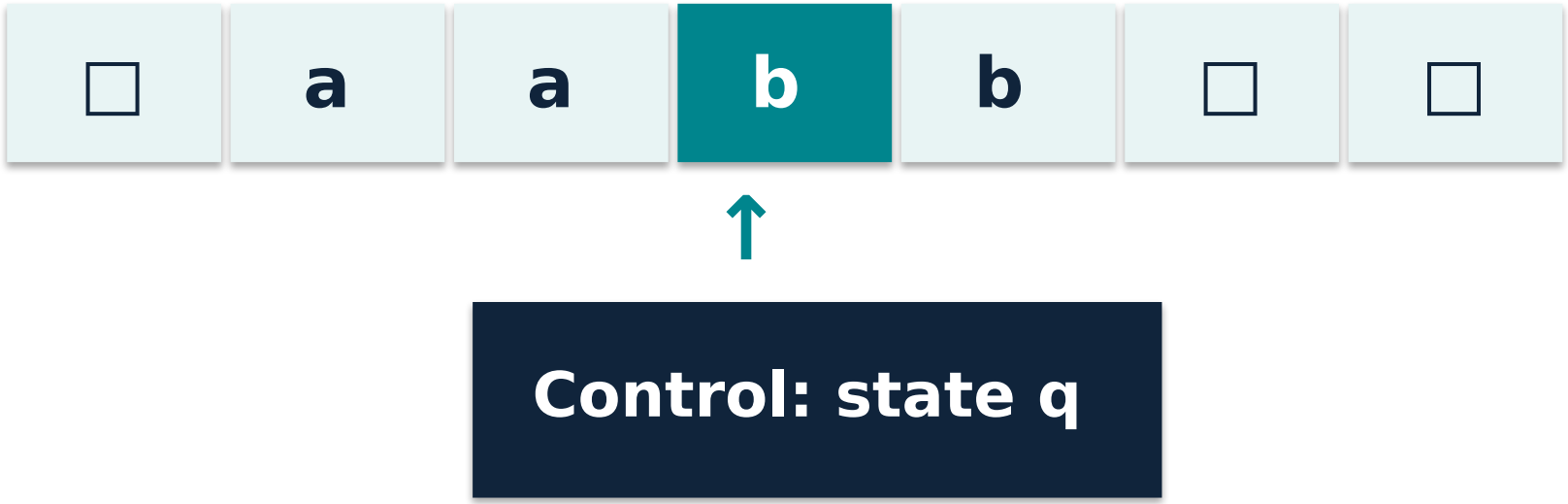
Design machines for language recognition and trace their configurations.

Computational power

Distinguish recognizers from deciders and explain equivalent TM variants.

Architecture of a Turing machine

Unbounded tape: read and write symbols, one cell at a time



The head may move left or right; unlike a PDA, tape contents can be revisited and changed.

Formal definition: seven components

$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$

Q: finite states; Σ: input alphabet; Γ: tape alphabet, containing Σ and blank $\square$.

Transition function

$\delta: (Q - \{q_{accept}, q_{reject}\}) \times \Gamma \to Q \times \Gamma \times \{L,R\}$.

Initial and halting conditions

Start in q_0 at the first input cell (blank for ε). Accept and reject states are distinct and halt. We use a two-way infinite tape.

Reading a transition and configuration

Example transition

$\delta(q_0,a) = (q_1,X,R)$: replace a with X, move right, enter q_1.

Configuration notation

u q v denotes tape contents uv, state q and the head scanning the first symbol of v. Omitted cells are blank.

One computation step

For input aab: $q_0aab \Rightarrow Xq_1ab$ under the example transition. A computation is a sequence of such configurations.

Recognizers and deciders

Recognizer

Accepts every word in its language. On a nonmember it may reject or run forever.

Decider

Halts on every input, accepting members and rejecting nonmembers.

Key relationship

Every decidable language is Turing-recognizable. Not every recognizable language is decidable (next lecture).

Example 1: decide L = {$a^n b^n : n \geq 0$}

Idea: match one a with one b

Mark the leftmost unmarked a as X, scan right to find the first unmarked b and mark it Y.

Repeat

Return to the blank immediately left of the input, move right and repeat. Skip X marks when looking for an a.

Finish

When the first non-X symbol is Y, check that only Y symbols remain. Accept ε as well; reject missing matches or unexpected symbols.

Example 1: transition table

State	a	b	X	Y	□
q_0	q_1,X,R	reject	q_0,X,R	q_3,Y,R	accept
q_1	q_1,a,R	q_2,Y,L	reject	q_1,Y,R	reject
q_2	q_2,a,L	q_2,b,L	q_2,X,L	q_2,Y,L	q_0,□,R
q_3	reject	reject	reject	q_3,Y,R	accept

Entries: next state, written symbol, move. Halting entries keep the scanned symbol and move R.

Example 1: tape trace for aabb



Highlighted cell = head position. Next transition accepts.

Example 1: correctness and termination

Invariant

After k successful rounds, exactly k a symbols and k b symbols have been marked. For $a^n b^n$, all remaining a symbols precede all remaining b symbols.

Acceptance and rejection

Acceptance requires all symbols to be matched and correctly ordered. If a b is missing, extra b remains, or an a occurs after a Y in the final scan, reject.

Termination

Every round marks one previously unmarked a and b, and all scans are finite. The machine halts on every input; time is $O(n^2)$.

Example 2: unary increment (a transducer)

Task

Input 1^n; output 1^{n+1} on the tape. This illustrates computing a function rather than merely accepting a language.

Transition rules

$\delta(q_0,1) = (q_0,1,R)$; $\delta(q_0,\square) = (q_accept,1,R)$.

Trace for 111

$q_0111 \Rightarrow 1q_011 \Rightarrow 11q_01 \Rightarrow 111q_0\square \Rightarrow 1111q_accept\square$. Output: 1111.

Example 3: decide binary palindromes

Match the endpoints

Find the leftmost unmarked bit, remember 0 or 1 in the state, and mark it X. Scan to the right blank, then move left past marks.

Compare

If no unmarked bit remains, accept (middle symbol). Otherwise compare the rightmost unmarked bit with the remembered bit; reject mismatch, or mark X.

Repeat and halt

Return to the left boundary. If all bits are marked, accept. Each round removes at least one unmarked bit, so the algorithm halts.

Multi-tape Turing machines

Model

k tapes, each with its own head. A step reads k symbols and writes k symbols, then moves the heads independently.

Simulation

A single tape stores all k tape contents separated by delimiters, with marked head positions. Scan to read, then update the encoded configuration.

Consequence

Multi-tape TMs have the same recognizable and decidable languages as single-tape TMs, though running times can differ.

Nondeterministic Turing machines

Model

A state and tape symbol may admit several successor transitions. Input is accepted if at least one computation branch accepts.

Deterministic simulation

Explore the finitely branching computation tree in breadth-first order. An accepting branch is eventually discovered if one exists.

Deciders

If every branch halts, the finitely branching tree is finite (König’s lemma). Exhaustive exploration halts, preserving decidability.

Other equivalent variants

Multi-track and stay-put moves

Encode several tracks as a tuple in each tape cell. A stay-put move can be simulated by two opposite moves with an auxiliary state.

One-way infinite vs. two-way infinite tape

Fold the two-way tape into two tracks of a one-way tape, recording the side and simulating movements across the origin.

Equivalence

These variants do not increase computability. A linear bounded automaton, however, is a restricted TM with input-length-bounded tape.

Universal TM and Church–Turing thesis

Universal Turing machine

U takes an encoding ⟨M,w⟩ and simulates M on w, step by step. It accepts, rejects or diverges as M does.

Stored-program idea

Both the machine description and input are data. A fixed interpreter can simulate many different machines.

Church–Turing thesis

Every effectively computable function can be computed by a TM. This is a thesis about effective computation, not a formal theorem.

Classroom exercises

Exercise 1

Trace the matching machine on ab, aab and ba. Identify the halting decision for each.

Exercise 2

Design a TM that decides whether a binary input has an even number of 1s. Specify states and transitions.

Exercise 3

Explain why depth-first exploration of a nondeterministic TM can fail to discover an accepting branch.

Solutions and recap

Exercise 1

ab: one match, then accept. aab: second a has no matching b, reject. ba: first symbol is b, reject.

Exercise 2

Use states E and O; on 0 keep state, on 1 toggle state, preserve symbols and move R. At blank, accept in E and reject in O. Start in E.

Exercise 3 and takeaway

Depth-first search may follow an infinite branch forever; breadth-first search avoids this. Next: hierarchy, decidability and undecidability.