Intro to Theory of Computation

CS 464

LECTURE 19

Last time

- Reductions
- Mapping reductions

Today

- Mapping reductions
- Computation history method

Sofya Raskhodnikova

Mapping Reductions

Mapping reductions

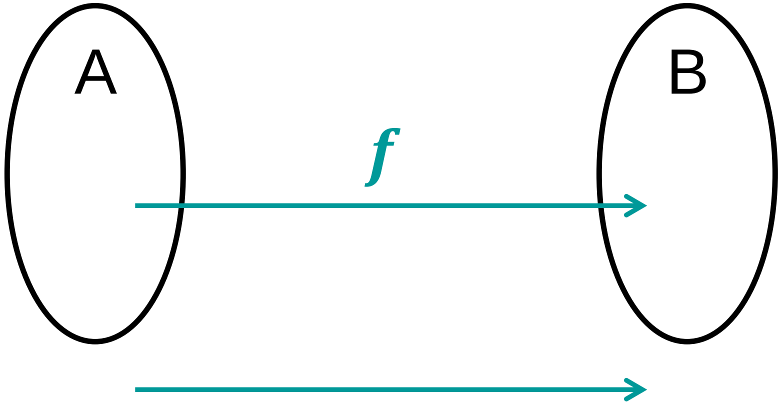
Given languages A and B,

$A \le_m B$

if there is a computable function f,

such that for all strings w,

$w \in A$ iff $f(w) \in B$.



Using mapping reductions to prove decidability

Theorem. If $A \leq_m B$ and B is decidable, then A is decidable.

Proof: Let M be a decider for B and f be a mapping reduction from A to B.
Construct a decider for A:
``On input w:
1. Compute f(w).
2. Run M on f(w).
3. If it accepts, accept. O.w. reject."

CS 464

Using mapping reductions to prove undecidability

Theorem. If $A \leq_m B$ and B is decidable, then A is decidable.

Corollary. If $A \leq_m B$ and A is undecidable, then B is undecidable.

Example: If $A_{TM} \leq_m B$, then B is undecidable.

Using mapping reductions to prove recognizability

Theorem. If $A \leq_m B$ and B is Turing-recognizable, then A is Turing-recognizable.

Proof: Let M be a TM that recognizes B and f be a mapping reduction from A to B.
Construct a TM that recognizes A:
``On input w:
1. Compute f(w).
2. Run M on f(w).
3. If it accepts, accept. O.w. reject."

CS 464

Using mapping reductions to prove unrecognizability

Theorem. If $A \le_m B$ and B is Turing-recognizable, then A is Turing-recognizable.

Corollary. If $A \le_m B$ and A is unrecognizable, then B is unrecognizable.

Example: If $\overline{A_{TM}} \le_m B$, then B is unrecognizable.

I-clicker question (frequency: AC)

If $\overline{\mathrm{A}} \leq_m \overline{B}$, we can conclude that

A. $A \leq_m B$

B. $B \leq_m \mathrm{A}$

C. $\overline{\mathrm{A}} \leq_m B$

D. $\overline{\mathrm{B}} \leq_m A$

E. **None of the above.**

Old proof that EQ_{TM} is undecidable

$EQ_{TM} = \{\langle M_1, M_2 \rangle \mid M_1, M_2$ are TMs, $L(M_1) = L(M_2)\}$

Proof: Suppose to the contrary that EQ_{TM} is decidable, and let R be a TM that decides it.

We construct TM S that decides A_{TM}.

S = `` On input $\langle M, w\rangle$, where M is a TM and w is a string:

1. Construct TMs M', M''.

   M' = `` On input *x*,
   1. Ignore the input.
   2. Run TM *M* on input w.
   3. If it accepts, accept."

   M'' = ``Accept."

2. Run TM *R* on input <M', M''>.
3. If $\boxed{\text{it accepts,}}$ accept. O.w. reject."

$A_{TM} \leq_m EQ_{TM}$

Proof: The following TM computes the reduction:

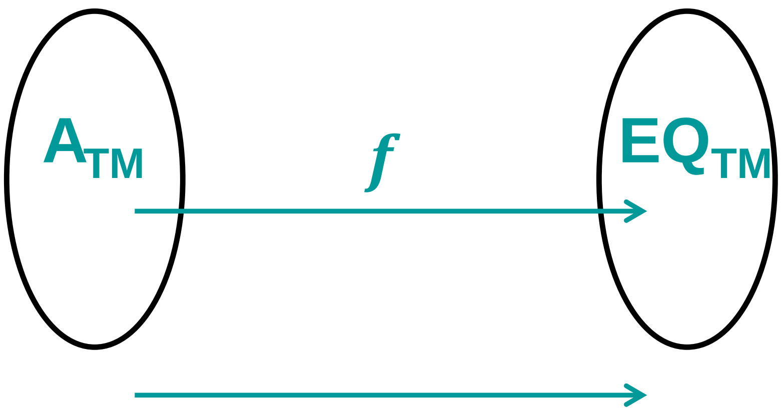
F = `` On input $\langle M, w \rangle$, where M is a TM and w is a string:

1. Construct TMs M', M''.

   M' = `` On input *x*,
   1. Ignore the input.
   2. Run TM *M* on input w.
   3. If it accepts, accept."

   M'' = ``Accept."

2. Output <M', M''>."



Conclusions from $A_{TM} \leq_m EQ_{TM}$

1. Since A_{TM} is undecidable, so is EQ_{TM}

2. $\overline{A_{TM}} \leq_m \overline{EQ_{TM}}$

   Since $\overline{A_{TM}}$ is unrecognizable, so is $\overline{EQ_{TM}}$

CS 464

Prove that EQ_{TM} is unrecognizable

Proof: We give a mapping reduction $\overline{A_{TM}} \leq_m EQ_{TM}$

The following TM computes the reduction:

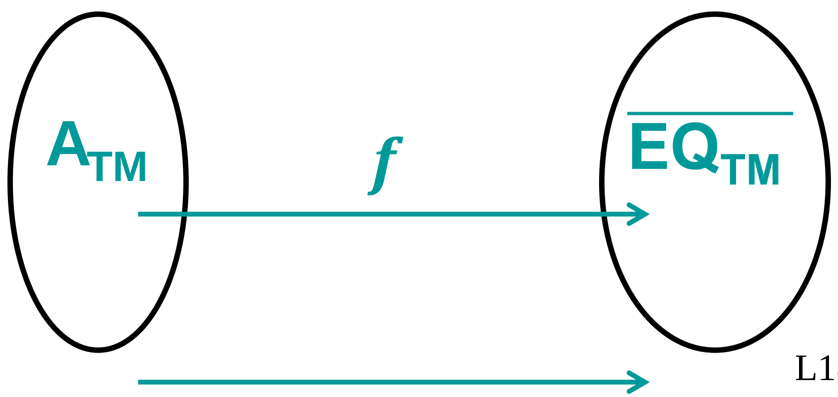
F = `` On input $\langle M, w \rangle$, where M is a TM and w is a string:

1. Construct TMs M', M''.

M' = `` On input *x*,
1. Ignore the input.
2. Run TM *M* on input w.
3. If it accepts, accept.”

M'' = ``Reject.”

2. Output <M', M''>.”



Problems in language theory

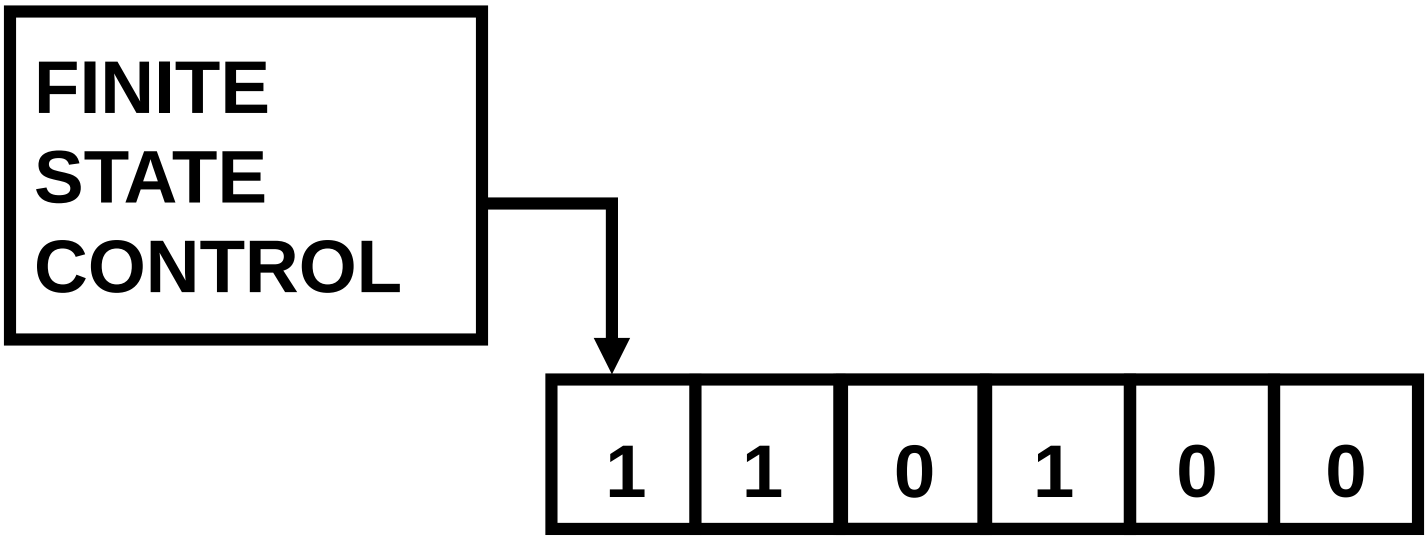
A_{DFA} decidable	A_{CFG} decidable	A_{TM} undecidable
E_{DFA} decidable	E_{CFG} decidable	E_{TM} undecidable
EQ_{DFA} decidable	EQ_{CFG} ?	EQ_{TM} undecidable

Proving undecidability for languages that do not involve TM descriptions

Computation history method

A linear bounded automaton (LBA)

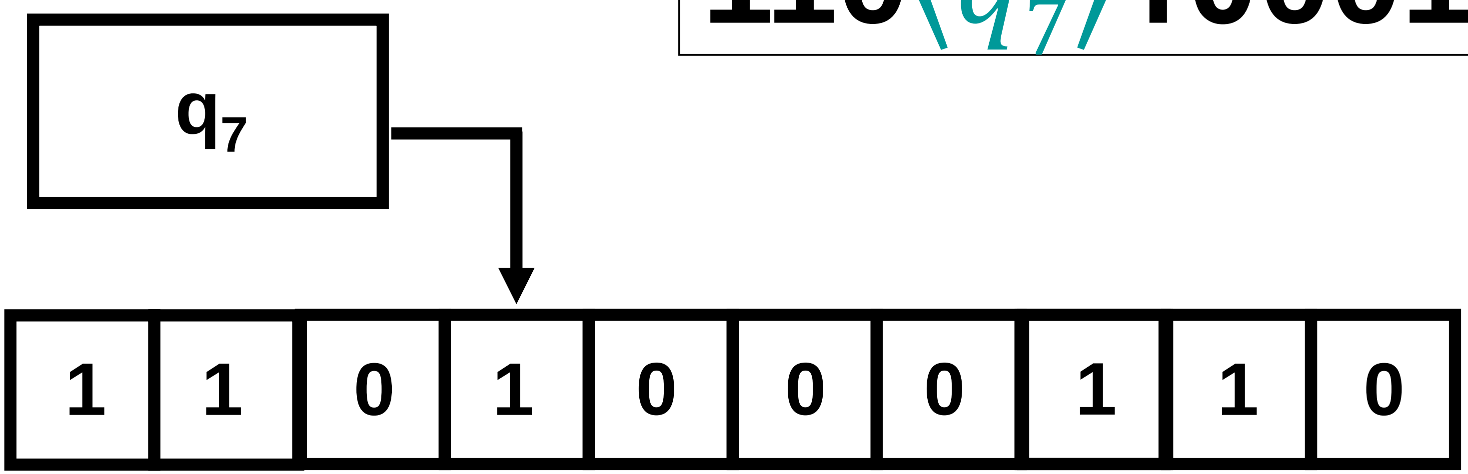
A **linear bounded automaton (LBA)** is a TM variant that has bounded tape, with the number of tape squares equal to the size of the input.



Configurations

A configuration of an LBA is a setting of state, head position and tape contents.

$$110\langle q_7\rangle 1000110$$



I-clicker question (frequency: AC)

How many distinct configurations does an LBA have if it has q states, g symbols in tape alphabet, and the tape of length n?

A. qgn

B. $q + g + n$

C. qg^n

D. qng^n

E. None of the above.

Prove that A_{LBA} is decidable

A_{LBA} = $\{\langle B, w\rangle \mid B$ is an LBA that accepts string $w\}$

Idea: Given $\langle B, w\rangle$, simulate B on w.

If it halts, we know the answer.

If it loops, we can detect because B repeats a configuration.

S = `` On input $\langle B, w\rangle$, where B is an LBA and *w* is a string:

1. Simulate B on w for qng^n steps.
2. If it accepts, accept.
3. If it rejects or does not halt, reject."

Computation history

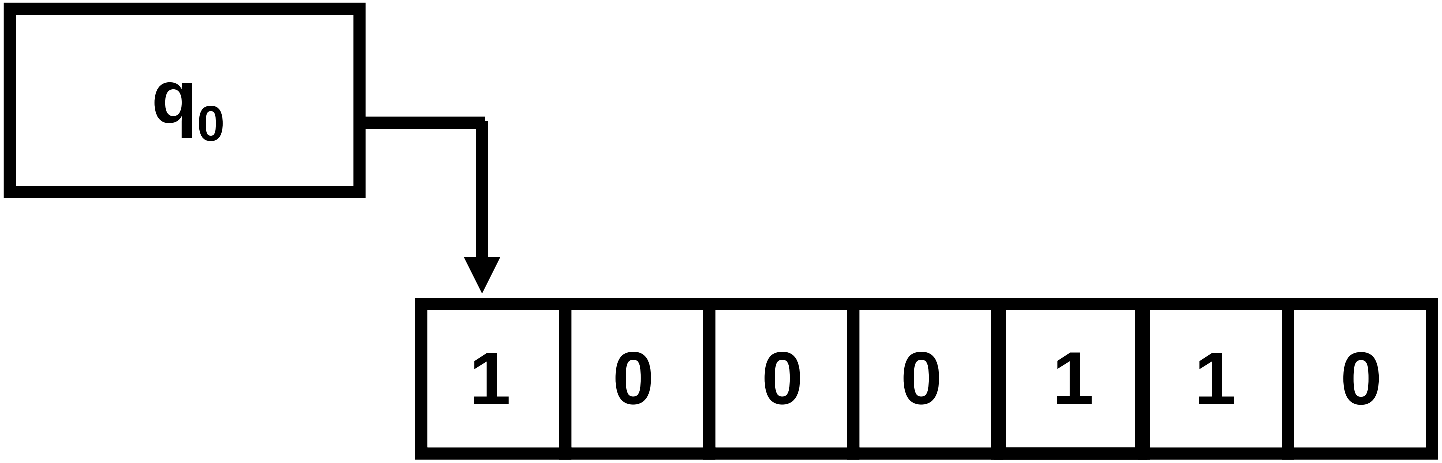
An accepting computation history for a TM M on input w is a sequence of configurations entered by M on input w:

$$\boxed{C_0 \# C_1 \# \ldots \# C_\ell}$$

starting configuration

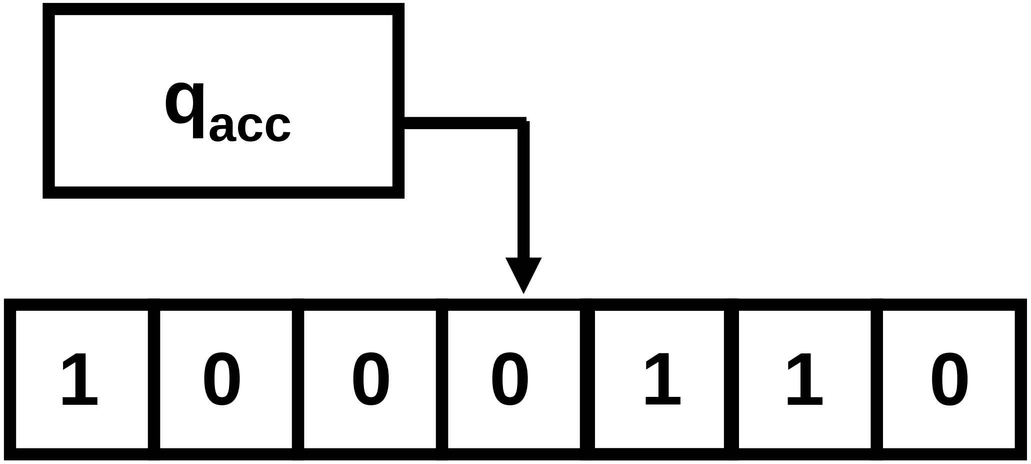
accepting configuration

Starting configuration



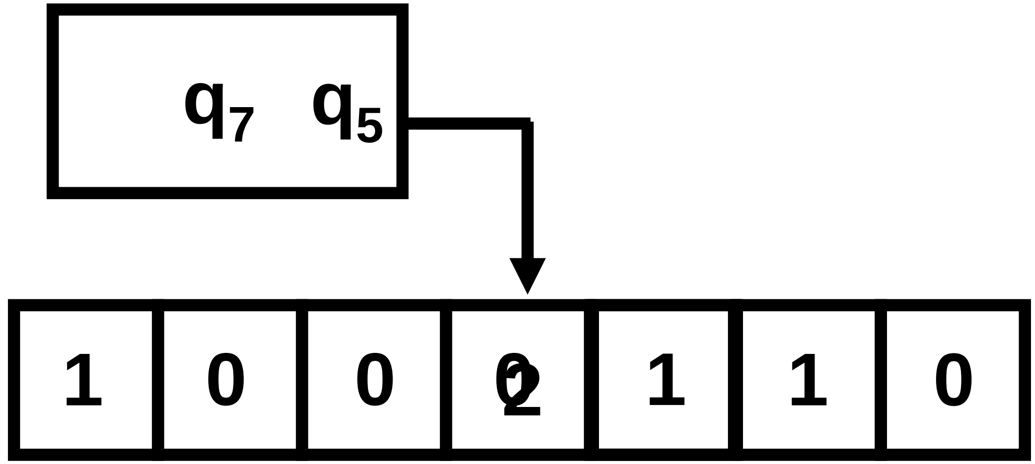
$$C_0 = \langle q_0 \rangle \mathbf{w}$$

Accepting configuration



$$C_\ell = \ldots \langle q_{acc} \rangle \ldots$$

Each C_{i+1} legally follows from C_i



$$C_i = 1\,0\,0\,\langle q_7\rangle\,0\;1\,1\,0$$

$$C_{i+1} = 1\,0\,0\;\mathbf{2}\;\langle q_5\rangle\,1\,1\,0$$

LBAs can check computation histories of TMs

Given a TM M and a string w, we can construct an LBA that checks whether its input is the accepting computation history of M on w.

Prove that E_{LBA} is undecidable

$E_{LBA} = \{ \langle B \rangle \mid B$ is a LBA and $L(B) = \emptyset \}$

Proof: Suppose to the contrary that TM R decides E_{LBA}.
We construct TM S that decides A_{TM}.

S = `` On input $\langle M, w\rangle$, where M is a TM and w is a string:

1. Construct an LBA B from M and w:

   B = `` On input *x*,
   Accept if x = $C_0\#\ldots\#C_\ell$ *is the accepting computation history of M on w:*
   1. C_0 is the starting configuration of M on w
   2. Each C_{i+1} legally follows from C_i
   3. C_ℓ is an accepting configuration for M ”

2. Run TM *R* on input <B>.
3. If $\boxed{\text{it rejects}}$, accept. O.w. reject.”