Today: § 4.2 + § 5.1 (in part)

Exam: One 2-sided 8½ x 11 sheet of notes,

Textbook: Formal description: you specify D

Implementation level: you give phases, how many tapes, how they move, etc.

High level: algorithm without any of discussion of types

\[ \text{Input} \xrightarrow{\text{Turing machine}} \begin{cases} \text{accept (yes)} \\ \text{reject (no)} \end{cases} \]

"loop" never halt

Only countably many "standard \( T.M.'s \)" \( \mathcal{Q} = \{1, \ldots, q\} \)

\( \mathcal{M} = \{1, \ldots, m\} \)

\[ A_{TM} = \{ \langle M, w \rangle \mid M \text{ is a Turing machine that accepts } w \} \]

undecidable.

Proof that \( A_{TM} \) is undecidable: If \( H \) decides \( A_{TM} \)

(assume to get a contradiction), then build a Turing machine, using \( H \) as a subroutine, so this Turing machine can't exist.

Oracle Turing machines

Build a machine \( D \) such

\( D \) given \( \langle M \rangle \), \( D \) figures out if \( w = \langle M \rangle \)

for some Turing machine.

Recall: Turing machine description was based on \( \Sigma = \{0, \ldots, 9, L, R, \#\} \)

But \( \Sigma \) really should be \( \{0, \ldots, 9, 10, 11, 12\} \)
(2) If \( w = \langle M \rangle \), feed \( \langle M, \langle M \rangle \rangle \) to \( H \) (i.e. run \( H \) as a subroutine).

3. If \( H \) accepts \( \langle M, \langle M \rangle \rangle \), then \( D \) rejects.
   
   If \( D \) rejects \( \langle M, \langle M \rangle \rangle \), then \( D \) accepts.

What happens to \( D \) on input \( \langle D \rangle \)?

If \( D \) accepts \( \langle D \rangle \), then \( H \) rejects \( \langle D, \langle D \rangle \rangle \) \( \Rightarrow \) \( D \) reject \( \langle D \rangle \)

**Remark:** \( \overline{A_{TM}} \) is recognized by a Univ Turing machine.

But \( \overline{A_{TM}} = \Sigma^{*} \setminus A_{TM} \) is not even recognized by any Turing machine.

Why? Say you can recognize \( L \) by a Turing machine \( M_{1} \) and \( \overline{L} \) by a Turing machine \( M_{2} \).

Given \( w \), can run in parallel \( M_{1} \) and \( M_{2} \).

A lot of this course: the following are undecidable:

- \( A_{TM} \), \( \text{halt}_{TM} \), \( \text{DoWeReachState}_{TM} \), ...
- which problems should we not work: \( \text{SAT} \), \( \text{3-COLOR} \), ...
- too long to completely solve: \( \text{PSPACE-complete} \), ...

So, \( L = \{ \langle M, w, q \rangle \mid M \text{ is Turing machine, } w \text{ input to } M, \text{ we do encounter state } q \} \)

\( \text{DoWeReachSameState} \) along the computation of \( M \) given \( q \).

\( \text{DO-WE-REACH-SAME-STATE} \) is recognizable, but undecidable.
If we simulate $M$ on input $w$, then if we ever land in $q$, we do so in finite number of steps, so we can stop and say ‘yes.’

But if $L$ were decidable, to solve $A_{TM}$ we just ask given $<M, w>$ does $<M, w, q_{acc}> \in L$?

Gwen, Le, Lz, ... are recognizable, then $L_1, L_2, L_3, ...$ are also recognizable.