CSE 105
THEORY OF COMPUTATION

"Winter" 2018

http://cseweb.ucsd.edu/classes/wi18/cse105-ab/

Today's learning goals

Sipser Section 3.1

- Design TMs using different levels of descriptions.
- Determine whether a Turing machine is a decider.
- Prove properties of the classes of recognizable and decidable sets.

Describing TMs

Sipser p. 159

- **Formal definition**: set of states, input alphabet, tape alphabet, transition function, state state, accept state, reject state.
- **Implementation-level definition**: English prose to describe Turing machine head movements relative to contents of tape.
- **High-level desciption**: Description of algorithm, without implementation details of machine. As part of this description, can "call" and run another TM as a subroutine.

Language of a TM

Sipser p. 144

L(M) = { w | M accepts w}

If w is in L(M) then the computation of M on w halts and accepts.

If the computation of M on w halts and rejects, then w is not in L(M).

If the computation of M on w doesn't halt, then w is not in L(M)

Deciders and recognizers *Sipser* p. 144 Defs 3.5 and 3.6

- L is **Turing-recognizable** if some Turing machine recognizes it.
- M is a **decider** TM if it halts on all inputs.
- L is **Turing-decidable** if some Turing machine that is a decider recognizes it.

An example

Which of the following is an implementation-level description of a TM which **decides the empty set**?

M = "On input w:

A. reject."
B. sweep right across the tape until find a non-blank symbol. Then, reject."
C. If the first tape symbol is blank, accept. Otherwise, reject."
D. More than one of the above.
E. I don't know.

Extension

- Give an implementation-level description of a Turing machine which **recognizes** (but does not decide) the empty set.

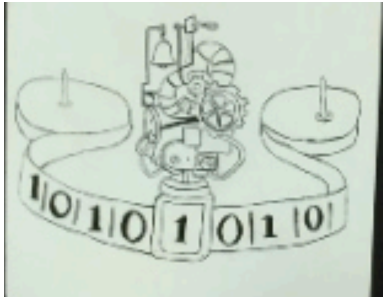
- Give a high-level description of this Turing machine.

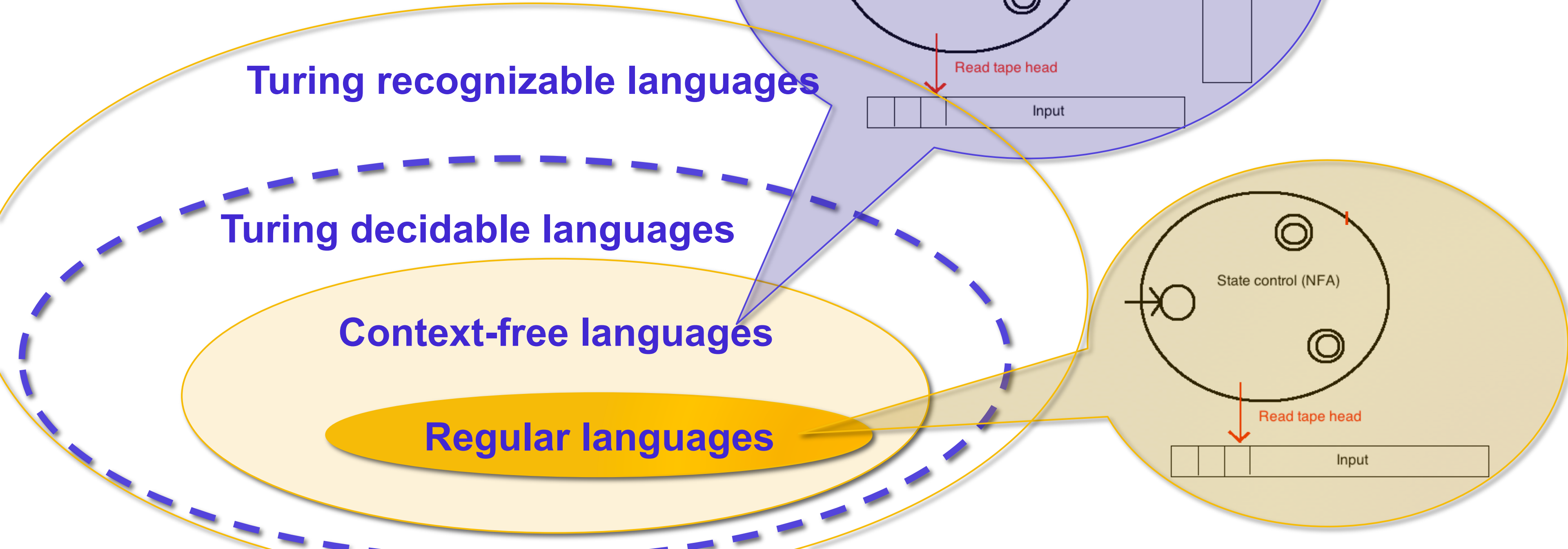
Classifying languages

A language L is

Turing-recognizable if there is a TM M such that L(M) = L.

Turing-decidable if there is a TM M such that M is a decider and L(M) = L.


1 0 1 0 1 0 1 0


Turing recognizable languages
Turing decidable languages
Context-free languages
Regular languages
Push/pop to stack
State control (NFA)
Stack
Read tape head
Input
State control (NFA)
Read tape head
Input

Closure

Theorem: The class of decidable languages over fixed alphabet Σ is closed under union.

Proof: Let …

WTS …

Closure

Theorem: The class of decidable languages over fixed alphabet Σ is closed under union.

Proof: Let L_1 and L_2 be languages over Σ and suppose M_1 and M_2 are TMs deciding these languages. We will **define** a new TM, M, via a high-level description. We will then show that **$L(M) = L_1 \cup L_2$ and that M always halts.**

Closure

Theorem: The class of decidable languages over fixed alphabet Σ is closed under union.

Proof: Let L_1 and L_2 be languages and suppose M_1 and M_2 are TMs deciding these languages. Construct the TM M as "On input w,

1. Run M_1 on input w. If M_1 accepts w, accept. Otherwise, go to 2.
2. Run M_2 on input w. If M_2 accepts w, accept. Otherwise, reject."

Correctness of construction:

WTS $L(M) = L_1 \cup L_2$ and M is a decider.

Where do we use decidability?

Closure

Good exercises – can't use without proof! (Sipser 3.15, 3.16)

The class of decidable languages is closed under	**The class of recognizable languages is closed under**
• Union ✓ • Concatenation • Intersection • Kleene star • **Complementation**	• Union • Concatenation • Intersection • Kleene star

For next time

GroupHW4 due Saturday, February 17

For Monday, pre-class reading: pp.