Lecture 17
Midterm Review

Midterm Mechanics

Friday

In Class

One page of notes allowed; otherwise closed book.

Covers:

- Sipser, Chapters 3, 4, 5;
- Lectures 1-13;
- Homework to date

Turing Machines

A simple model of "mechanical computation"

Details:

- state/config
- move left/right/(not stay still, except...)
- left end
- halt/accept/reject
- 1 tape / multi-tape
- computation histories (accepting/rejecting)

Church-Turing Thesis

All "reasonable" models are alike in capturing the intuitive notion of "mechanically computable"

Unprovable (because it's loosely defined)

Support:

- provable equivalence of various "natural" models
- inequivalence of really weird models?
  - "Run ∞ steps and then..."
  - "Ask the gods whether M halts on w and if not then..."

Decidable/Recognizable

Does it halt?

Languages :: accept/reject :: yes/no :: 0/1

(Turing) Decidable:

answer and *always halt*

(Turing) Recognizable

halt and accept, but may reject by *looping*

Undecidability

Diagonalization

Cardinality:

Uncountably many languages

Only countably many recognizable languages

Only countably many decidable languages

A specific Turing recognizable, but undecidable, language:

A_{TM} = { <M,w> | TM M accepts w }

A specific non-Turing-recognizable language:

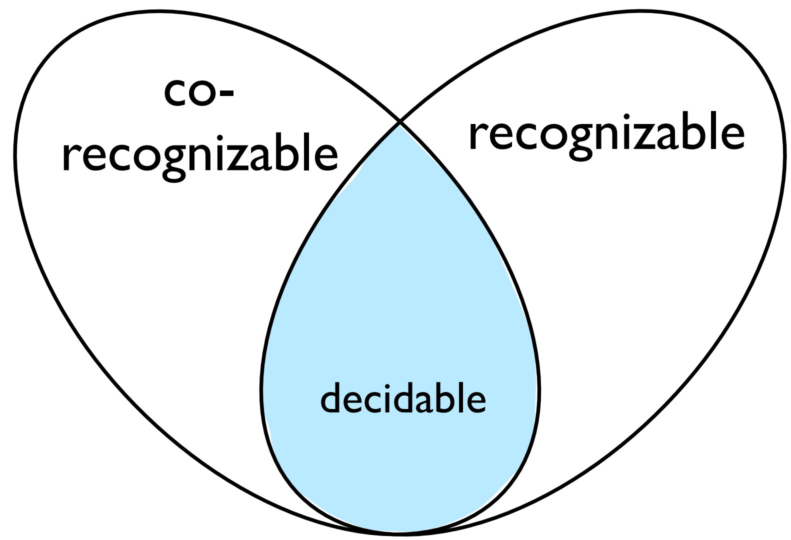
$\overline{A_{TM}}$

Decidable = Rec ∩ co-Rec

L decidable iff both L & L^c are recognizable

Pf:
(⇐) on any given input, dovetail a recognizer for L with one for L^c; one or the other must halt & accept, so you can halt & accept/reject appropriately.

(⇒):
decidable languages are closed under complement (flip acc/rej)



Reduction

"A is reducible to B" (notation: $A \leq_T B$) means I could solve A *if* I had a subroutine for B

Key Facts:

$A \leq_T B$ & B decidable implies A decidable (almost the definition)

$A \leq_T B$ & A *un*decidable implies B undecidable (contrapositive)

$A \leq_T B$ & $B \leq_T C$ implies $A \leq_T C$

Many Undecidable Problems

About Turing Machines

- $HALT_{TM}$ EQ_{TM} $EMPTY_{TM}$ $REGULAR_{TM}$ ...
- Rice's Theorem

About programs

- Ditto! *And*: array-out-of-bounds, unreachability, loop termination, assertion-checking, correctness, ...

About Other Things

- $EMPTY_{LBA}$ ALL_{CFG} EQ_{CFG} PCP DiophantineEqns ...

Mapping Reducibility

Defn: A is *mapping reducible* to B ($A \leq_m B$) if there is computable function *f* such that $w \in A \Leftrightarrow f(w) \in B$

A special case of $\leq_T$:

Call subr only once; its answer is *the* answer

Theorem:

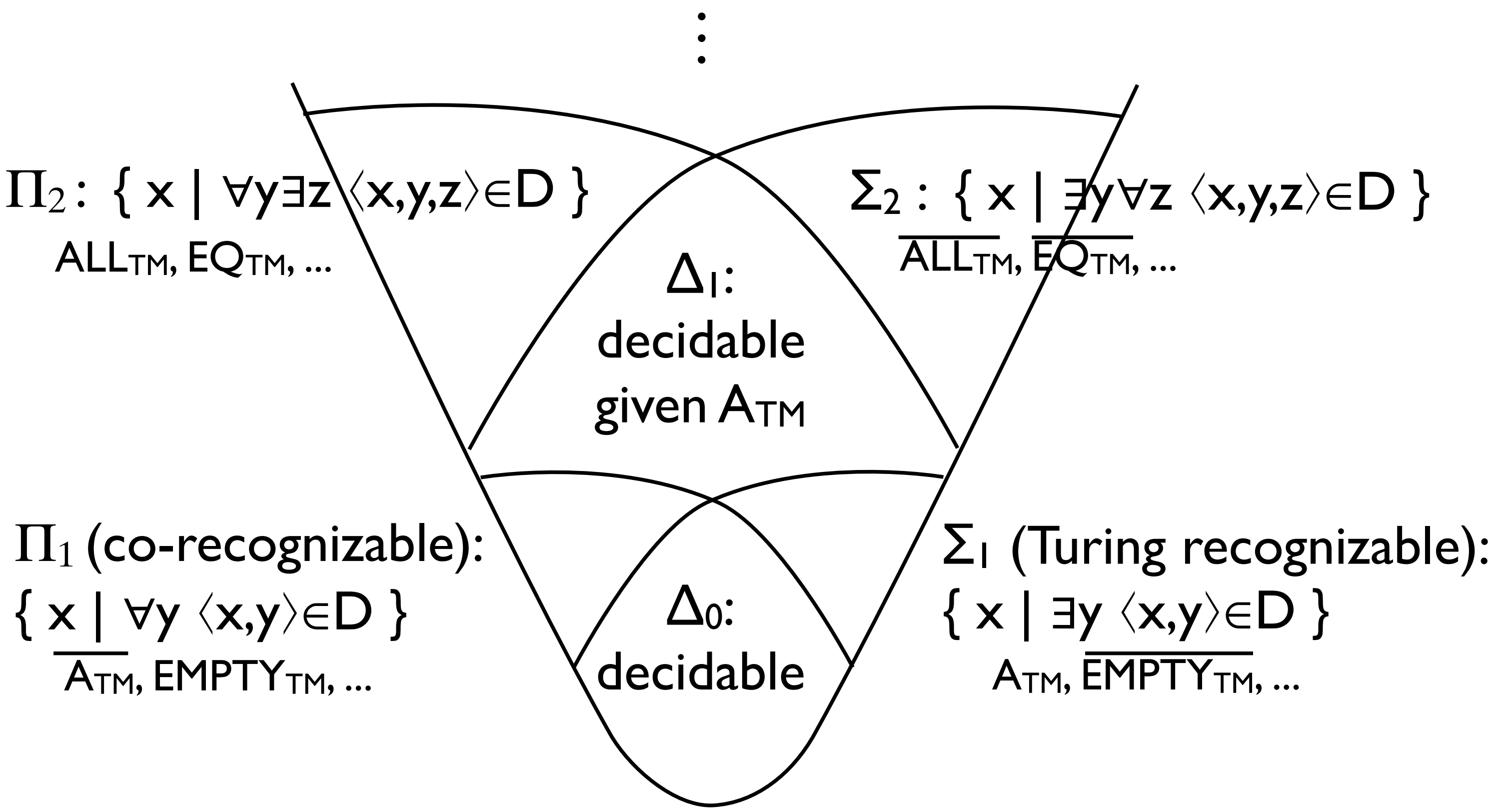
$A \leq_m B$ & B decidable (recognizable) $\Rightarrow$ A is too

$A \leq_m B$ & A *un*decidable (*un*recognizable) $\Rightarrow$ B is too

$A \leq_m B$ & $B \leq_m C \Rightarrow A \leq_m C$

Most reductions we've seen were actually $\leq_m$ reductions.
(And if not, then $A \leq_m \overline{B}$ is likely.)

The "Arithmetical Hierarchy"



Potential Utility: It is often easy to give such a quantifier-based characterization of a language; doing so suggests (but doesn't prove) whether it is decidable, recognizable, etc. and suggests candidates for reducing to it.