Readings from Sipser

- Section 4.2, Subsections:
  - Introduction (Skip Diagonalization Method)
  - An Undecidable Language

From Last Time…

- Recall that we can create Turing Machines that take the descriptions (i.e., “programs”) of other Turing Machines as input, for example:
  - Translators
  - Compilers/Typecheckers
  - Interpreters, e.g., Universal Turing Machines

The Language A_{TM}

- The language A_{TM} is the set of all <M, w> pair encodings where M accepts w

$$A_{\mathsf{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$$

- Question: Is the language A_{TM} *recognizable*?
  - In other words, can we construct a TM that accepts *exactly* those strings that are in A_{TM}, and either rejects or loops infinitely on everything else?

Hmmm…. I think we already have something that does this!

Behavior of UTM

On input <M, w>, if…	UTM…
M accepts w	Accepts
M rejects w	Rejects
M loops forever on w	Loops forever

Review: Deciders vs. Recognizers

On input x, if…	Decider	Recognizer
$x \in L$	Accepts	Accepts
$x \notin L$	Rejects	Rejects or Loops forever

An Even Bigger Question…

- Is the language A_{TM} *decidable*?
  - UTM doesn't work for this here

On input <M, w>, if…	UTM… (recognizer)	Decider for A_{TM} must…
M accepts w	Accepts	Accepts
M rejects w	Rejects	Rejects
M loops forever on w	Loops forever	Rejects

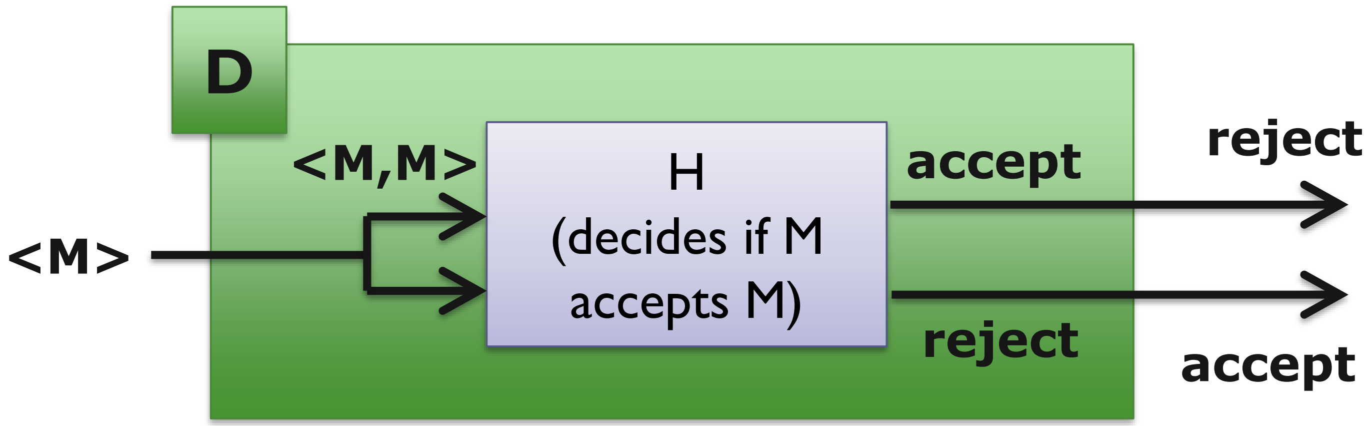
Extremely Important Theorem

- Theorem: The language A_{TM} is <u>not</u> decidable
- Proof (by contradiction): Assume such a decider exists! Let's call it H. Use it to construct a new decider D that checks if a machine M accepts its own code M, and then inverts the result:

On input <M>, if…	H	D
M accepts M	Accepts	Rejects
M rejects or loops forever on M	Rejects	Accepts

Extremely Important Theorem

- Theorem: The language A_{TM} is <u>not</u> decidable
- Proof (by contradiction): Assume such a decider exists! Let's call it H. Use it to construct a new decider D that checks if a machine M accepts its own code M, and then inverts the result:



Now, let's break stuff!

Proof (continued)

- Now, run D using its own code <D> as input!
  1. If it <u>rejects</u>, this means that H accepted <D, D>, so <D, D> must be in A_{TM}, which means that D should <u>accept</u> <D> as input – a contradiction!
  2. If it <u>accepts</u>, this means that H must have rejected <D, D>, and so <D, D> is not in A_{TM}, which means that D <u>cannot accept</u> <D> – another contradiction!
- Thus, no such decider H exists for A_{TM}