Rice's Theorem

- If:
  - L = { <M> | where L(M) has property P}, and
  - L is not ∅ or the set of all TMs
- Then L is undecidable (and possibly unrecognizable)
- Proof Sketch: We can show that if we had a decider for L, then we could build a decider for A_{TM} or for $\overline{A_{TM}}$ out of it

Proof – Two Cases

- Since, L is not everything or nothing, we know that there are TMs K and J such that:
  - <K> $\in$ L, and <J> $\in \overline{L}$
- Assume D_L is a decider for L
  - We will then show D_L can't exist
- Case 1: The language $\varnothing$ has property P
- Case 2: The language $\varnothing$ does not have property P

Proof – Case 1

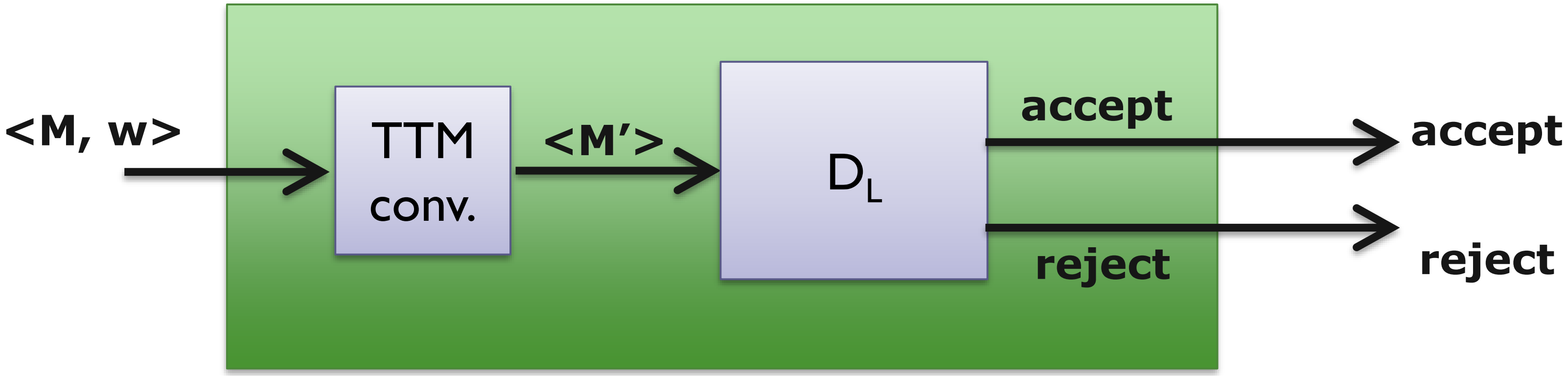
- Case 1: The language $\emptyset$ has property P
- We can create a TTM code converter that transforms a <M, w> pair into a single TM M':

M': On input x:

1. Run M on w:
   a) If it rejects, then <u>reject</u>
   b) If it accepts, then run J on x, and output the same result for M'

Proof – Case 1

- Case 1: The language ∅ has property P
- Using the converter and D_L, we now create the following decider for $\overline{A_{TM}}$:



- If <M, w> in $\overline{A_{TM}}$, then M' accepts nothing and is thus in L
- Otherwise, M' behaves just like J, which isn't in L

Proof – Case 2

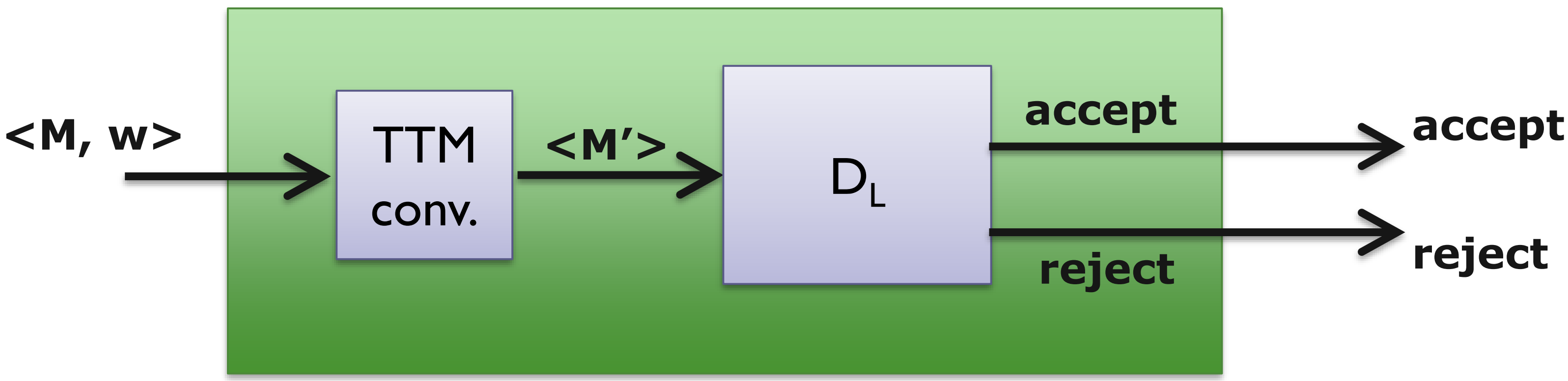
- Case 1: The language ∅ doesn't have property P
- We can create a TTM code converter that transforms a <M, w> pair into a single TM M':

M': On input x:

1. Run M on w:
   a) If it rejects, then <u>reject</u>
   b) If it accepts, then run K on x, and output the same result for M'

Proof – Case 2

- Case 1: The language ∅ doesn't have property P
- Using the converter and D_L, we now create the following decider for A_{TM}:



- If <M, w> in A_{TM}, then M' behaves just like K, which is in L
- Otherwise, M' accepts nothing, and thus not in L

Rice's Theorem Uses

- Now, we automatically know that non-trivial languages of the form {<M> | L(M) has prop. P} are in one of the following categories:
  - ~~Decidable~~
  - Recognizable, but not decidable
  - Unrecognizable
- Common strategy: See if you can find a recognizer for the given language or its complement

Common Examples

- Languages of the form:

{ <M> | M accepts at least n strings }

{ <M> | M accepts everything in F (maybe more) }

where F is finite, are recognizable

- Proofs: Exercise

Common Examples

- Languages of the form

{ <M> | M accepts at most n strings }

{ <M> | M accepts exactly those strings in S }

{ <M> | M accepts an infinite number of strings }

where S is some decidable set, are unrecognizable

- Proofs: First one is easy (given previous proofs); Other two are challenging!