30 Aug

HALTING PROBLEM (ver. 2)

Input: A program P, an input I

Output: Yes if P halts on I
No otherwise

THEOREM: There is no algo A' that solves the Halting problem (assuming A' always terminates)

PROOF (DETAILS) Assume for the sake of contradiction that there exists an algo magic_box s.t. on every pair (P,I):
(i) magic_box terminates and
(ii) Outputs Yes $\iff$ P halts on I

Now define

```
contradiction (P):
1. if magic_box (P,P): // Run an UTM to execute magic_box (P,P)
2.     while True:
3.         pass
4. return
```

Since we assumed magic_box exists, contradiction is a well-defined program
$\Rightarrow$ contradiction (contradiction) is a well defined function call.

We now do a case analysis:

Case 1: contradiction (contradiction) terminates

By case assumption and assumptions (i) & (ii) the call to magic_box in line 1 evaluates to True
$\Rightarrow$ Lines 2+3 get executed $\Rightarrow$ contradiction (contradiction) does NOT terminate

Case 2: contradiction (contradiction) does NOT terminate

By case assumption & assumptions (i) & (ii) the call to magic_box in line 1 evaluates to False
$\Rightarrow$ line 4 gets executed $\Rightarrow$ contradiction (contradiction) DOES terminate.

In both cases, we get a contradiction. ■

Proof idea: We'll use proof by contradiction

Assume ∃ an algo magic_box that solves the Halting problem.

We'll define another algo/program contradiction(P) that uses magic_box as a black box.

We'll consider the function call contradiction(contradiction) and derive a contradiction (using a case analysis). ∎