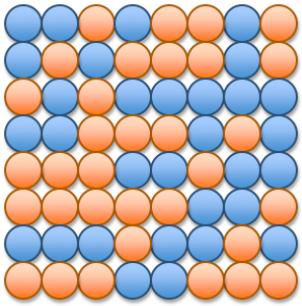


UNIT 14C

The Limits of Computing: Non-computable Functions

Problem Classifications

- Tractable Problems
  - Problems that have reasonable, polynomial-time solutions
- Intractable Problems
  - Problems that may have no reasonable, polynomial-time solutions
- Noncomputable Problems
  - Problems that have no algorithms at all to solve them

The Barber Paradox

Suppose there is a town with one male barber; and that every man in the town keeps himself clean-shaven: some shave themselves and some are shaved by the barber. Only the barber can shave another man. The barber shaves all and only those men who do not shave themselves.

Does the barber shave himself?

Program Termination

- Can we determine if a program will terminate given a valid input?
- Example:

```
def mystery1(x):
    while (x != 1):
        x = x - 2
```

  - Does this algorithm terminate when x = 15?
  - Does this algorithm terminate when x = 110?

Another Example

```
def mystery2(x):
    while (x != 1):
        if x % 2 == 0:
            x = x / 2
        else:
            x = 3 * x + 1
```

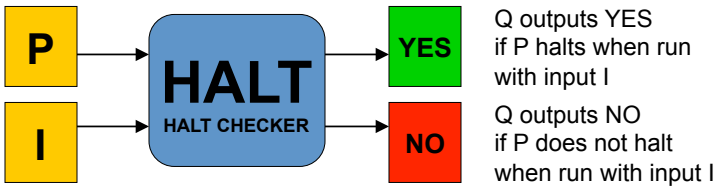
- Does this algorithm terminate when x = 15?
- Does this algorithm terminate when x = 110?
- Does this algorithm terminate for any positive x?

The Halting Problem

- Does a universal program HALT exist that can take any program P and any input I for program P and determine if P terminates/ halts when run with input I?
- Alan Turing showed that such a universal program HALT cannot exist.
  - This is known as the Halting Problem.

Proof by Contradiction

- Assume a program HALT exists that requires a program P and an input I.
  - Q determines if program P will halt when P is executed using input I. No assumptions are made on how it does this. Anything is possible.



  - **We will show that Q cannot exist by showing that if it did exist we would get a logical contradiction.**

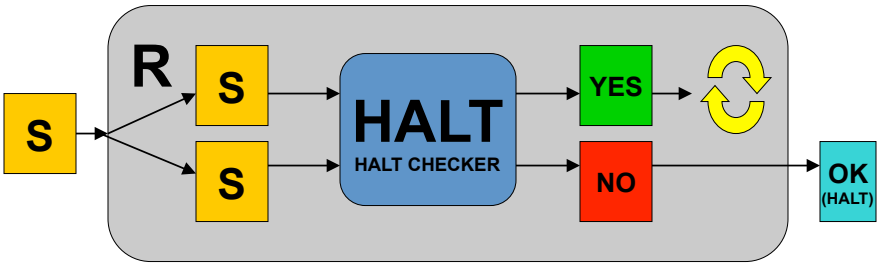
Can a program have as its input a program?

- A compiler is a program that takes as its input a program that needs to be translated from a high-level language (e.g. Python) to a low-level language (e.g. machine language).
  - In general, a program can process any data, so it can have a program as its input to process.
- Deep thought: Could a compiler compile itself since a compiler is a program? (Yes!)

Proof (cont'd)

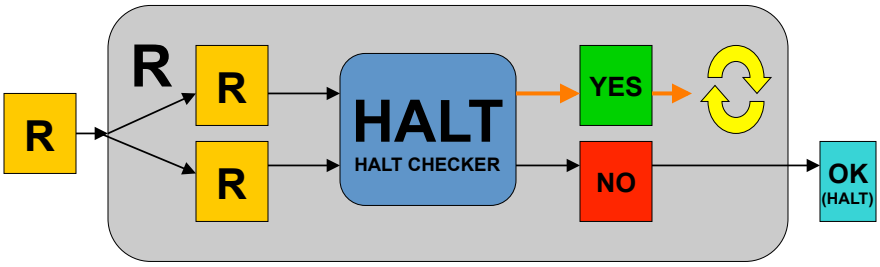
- Let R be a program that takes input S, where S is a program.
- R asks the halt checker HALT what happens if S runs with itself as input?
- If HALT answers that S will halt if it runs with itself as input, then R goes into an infinite loop (and does not halt).
- If HALT answers that S will not halt if it runs with itself as input, then R halts.

How R Works



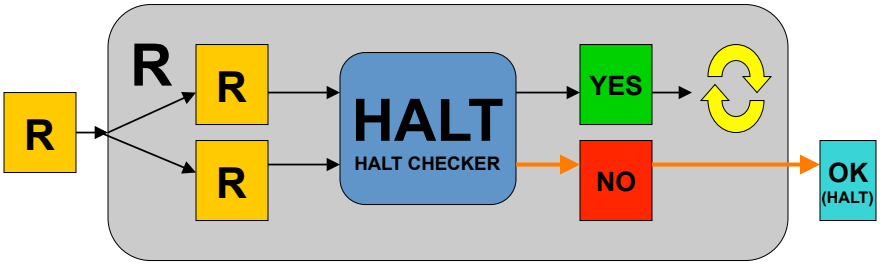
R gets evil

- What happens if R tests itself?
  - If Q answers yes (R halts), then R goes into an infinite loop and does not halt.



R gets evil

- What happens if R tests itself?
  - If Q answers no (R does not halt), then R halts.



A Simpler Way To Look At It

```
def r(r):
    if halt(r,r):
        while True:
            None
    else:
        return None
```


Contradiction!

- No matter what HALT answers about R, R does the opposite, so HALT can never answer the halting problem for the specific program R.
  - Therefore, a universal halting checker HALT cannot exist.
- Conclusion: We can never write a computer program that determines if ANY program halts with ANY input.
  - It doesn't matter how powerful the computer is.
  - It doesn't matter how much time we devote to the computation.

Contradiction in Real Life