Lecture 34

By definition any given Turing Machine runs a fixed program. We have discussed the fact that the program might be an interpreter than runs other programs. Today we see how that works.

Universal Turing Machine:
- has a fixed program permanently embedded in its finite control
- the fixed program can mimic the action of an arbitrary TM (i.e., it’s an interpreter)
- program is written on one tape and mimicked on another (so there are two tapes)

The fixed program in U is really just an interpreter:

1. Given: T's current state and input symbol
   Find: quintuple in description of T that applies
   \[(q, s, q', s', d), \text{ where}\]
   \[q = \text{current state}\]
   \[s = \text{current symbol}\]
   \[q' = \text{next state}\]
   \[s' = \text{symbol to write}\]
   \[d = \text{direction to move}\]

   *Note extension to standard TM. Why is this ok?

2. Record: state q'
   Simulate on tape 2: write s'; move tape 2's read/write head
Record: new symbol from tape 2

Universal machine U expects a particular format for the description of T's program:
  will use binary alphabet to represent
  • quintuples of T
  • T's tape symbols

(1) How many bits are needed for each quintuple?
  n states: need \( \lceil \log n \rceil \) bits to encode
  let \( k = \lceil \log n \rceil \)
  2 tape moves (L and R): 1 more bit to encode
  each tape square: 1 bit

  ⇒ 2k + 3 bits needed to encode each quintuple

(2) How to separate quintuples?
  ⇒ use special symbol X

(3) Boundary markers for program?
  ⇒ special symbol Y

U's interpreter program:
  (only part of it, actually)

**Phase 1**

locate next quintuple to be executed - i.e., find q & s

where will q and s be stored?
  • could use 3rd tape  -or-
  • use (k+1)-bit segment immediately to left of the left Y-marker
**Algorithm.**
- start with read/write head on left Y-marker
- scan to left
  - change 0 to A
  - change 1 to B
  - until blank
- scan to right
  - if A, change to 0
  - else if B, change to 1
  - move right and search for match
  - if you've moved past an X, need to start this cycle again (i.e., you've moved to the next quintuple without having matched)
- to guarantee skipping over a non-matching quintuple, change 0's to A and 1's to B.

![Diagram of TM states and transitions](image)

*Note difference in TM representation*

**Phase II**

**Algorithm.**
- record new state q' in workspace
- move head on tape 2
- record new symbol
at start:
  q and s from quintuple are A's and B's;
  but
  q' and s' and d are still binary

r/w head on left-hand Y-marker

**Algorithm at next level of detail.**
- scan right to binary squares
- copy to workspace (in terms of A's and B's)
- encounter s'; copy to workspace
- scan right and pick up d; no more room in workspace; so
  "remember" d
- pick up s'
  convert to 0/1 and write on tape 2
- move tape 2's r/w head
- read next s from tape 2

In the following TM
  squares = tape 2
  underlines = tape 2
Now -- how do things get started?
  r/w head #1 over left-hand Y-marker
  r/w head #2 over initial cell of tape 2
  initialize workspace