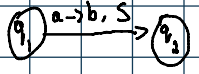
Lecture 23 : Variations of Turing Machines

December 8, 2020 11:18 AM

① Allow a stay option for the tape head
② Semi-infinite tape (infinite only in 1 direction)
③ off line tape
④ Multi-tape (with headers for each)
⑤ Multi-dimensional tapes
⑥ Non-Deterministic Machines.

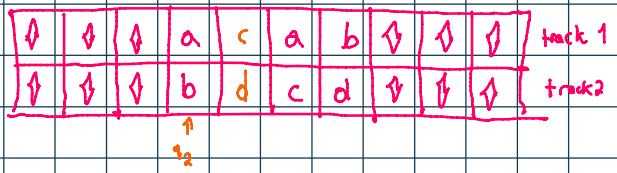
. If we ever modify the standard turing machine model to include any of these, the resulting variation forms different classes of Turing machines. Each class, has exactly the same power as the standard Turing machine. That does not refer to the efficiency, it just means both will accept the same languages. $L(M_1) = L(M_2)$

① Turing Machine with Stay-option

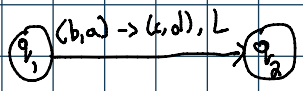


. To prove that this variation has same power as standard TM we need to prove simulation both ways.
① Stay option Machine, is a standard TM that never uses that option
② A standard TM can simulate stay option. For every transition with Stay step can be simulated with 2 transitions in standard TM, 1 right, 1 left.

TM with multiple track tape: (This is not a new class of Turing Machines, because it still has only one head it's just a useful representation for proving simulations. All the standard Turing Machine specs are respected. The alphabet is Γ, Γ^2 or Γ^3. Browse symbols are now pairs.



So to represent this transition in a standard TM:

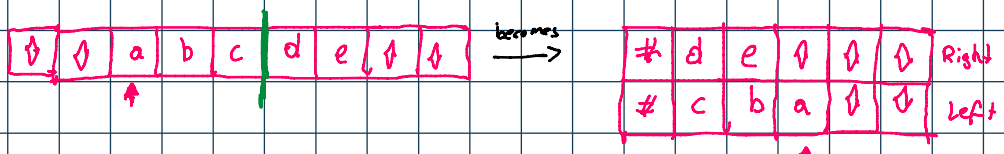


② Semi-infinite tape

. Now we use another symbol to represent the end of tape #.

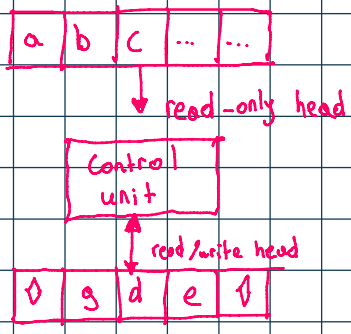
. To prove that this variation has some power as standard TM we need to prove simulation both ways.
① Semi-infinite TM, is a standard TM that doesn't use half of the tape that is infinite to the left.

② To make a semi-infinite TM simulate a standard TM we fold the tape and use a multi-track tape.



So in the new machine if we move to the left of c, we scan the # symbol, if we go left, move the head to top track

③ The off-Line Machine



. To prove that this variation has same power as standard TM we need to prove simulation both ways.

① off-line machines can simulate standard Turing machines by copying the input file to the tape then continue as a standard TM

② A standard TM can simulate OFF-Line by using 4 track tapes to keep track of the off-line input file and tape content.

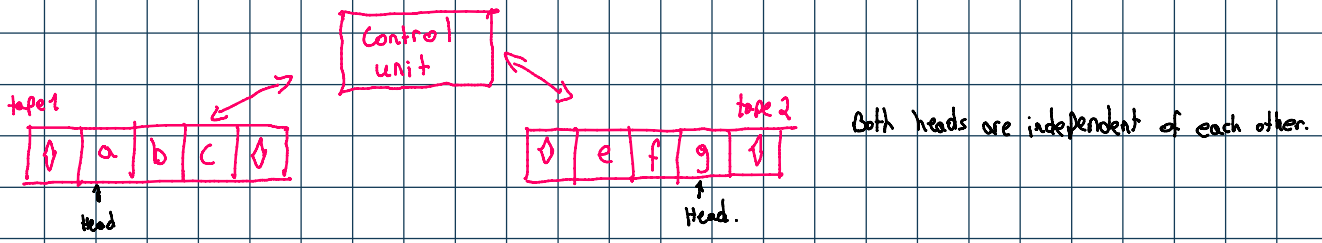
Track 1: A pointer to where the read head is in the input file/tape.

Track 2: The input file content (non-modifiable)

Track 3: A pointer to where the read-write head is in the working tape.

Track 4: The work tape content.

④ Multi-tape Turing Machines



Both heads are independent of each other.

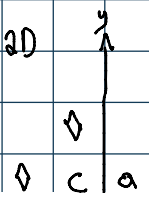
. To prove that this variation has same power as standard TM we need to prove simulation both ways.

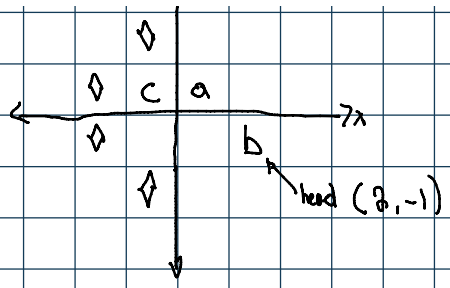
① Just use one tape to simulate a standard TM

② standard TM can simulate Multi-tape by using a pair of tracks for every tape. pointer to head and content.

⑤ Multi-dimensional Turing machines

They use multi-dimensional tapes, for example 2D

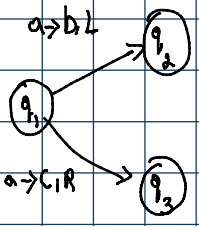




Again we use multi-track tapes to simulate the behavior.

(6) Non-Deterministic Turing machines

- They have a non-deterministic choice to make, by having this kind of ambiguous definition of transitions.



- An input string w is accepted if this is possible:

$$q_0 w \overset{*}{\Rightarrow} x q_p y$$

* A Universal Turing Machine.

- It is a reprogrammable Turing machine. So its not hardwired for only one purpose and it can simulate any other TM
- Inputs to a UTM: ① Description of transitions of M
  ② initial contents of tape of M

- We can represent this UTM by using a TM with 3 tapes (we know that standard works too w.l.o.g as we proved earlier.

  Tape 1: Description of M. (Transition Function encoded in unary, seperated by 0s)

  $\delta(q_1, a) = (q_2, b, L)$ becomes 10101101101

  $\delta(q_2, b) = (q_3, c, R)$ becomes 1011011101110111

  How do we encode all the transition functions in the tape? concatenate + add a new 00 seperator

  10101101101001011011101110111

  Tape 2: Inputs / Tape content of M

  Tape 3: State of M. Which state M is currently in.

- This simulation works because by scanning tape 3 it figures out what state M is in. Then by reading the content in tape 2 it knows where the tape head is and what symbol it is scanning. Then it looks at tape 1 which tells it what to do based on the state and input. So it can go update Tape 2 and 3.

- So we've shown that A Turing machine is described using a binary string of 0s and 1s

Therefore, the set of all turing machines forms a language where each string of the language is the binary encoding of a TM.