CS310 Lecture 14
Introduction to Turing Machines

Vedang Asgaonkar (200050154), Virendra Kabra (200050157)

28 September 2022

1 Recap

- We have seen FSA and PDA, and their limitations. For example, $\{a^n b^n \mid n \geqslant 1\}$ and $\{a^n b^n c^n \mid n \geqslant 1\}$ cannot be expressed using FSA and PDA, respectively.
- Deterministic PDA (DPDA) accept all regular languages, but only a proper subset of CFLs. For instance, $\{ww^R \mid w \in \{0,1\}^*\}$ has no DPDA, but is context-free. In this sense, non-deterministic PDA are more powerful than DPDA.
- PDA with two stacks are as powerful as Turing Machines. For example, $\{a^n b^n c^n \mid n \geqslant 1\}$ can be expressed as follows: Push a's and b's in separate stacks (ensuring that b's follow a's). For every c, pop from each stack. Accept iff both stacks are empty and the entire input string is consumed. Other data structures, such as queues, can also be used.

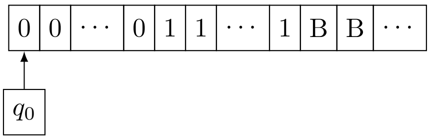
2 Turing Machines

- Formal notation: 7-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$
  - Q: Finite set of states.
  - Σ: Finite set of input symbols.
  - Γ: Complete set of tape symbols. That is, Σ, B, and other tape symbols.
  - δ: Transition function ($Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$). This is a partial mapping. $\delta(q, X) = (p, Y, D)$, where q, X are current state and tape symbol, p, Y, D are next state, replacement symbol, and direction, respectively.
  - q_0: Start state.
  - B: Blank symbol. $B \in \Gamma, B \notin \Sigma$.
  - F: Set of final/accepting states.

  The tape is divided into *cells*, each holding an element of Γ. The tape can be doubly-infinite, or bounded at one end (this does not change the computational power). The *tape head* is always positioned at one of the tape cells.
- Instantaneous Description (ID): At any point, the TM can be represented as $X_1 X_2 \cdots X_{i-1} q X_i X_{i+1} \cdots X_n$, where
  - q is the current state.
  - Tape head points to X_i.
  - $X_1 \cdots X_n$ is the portion of the tape between the leftmost and rightmost blank cells. If the head points to a blank cell, then some prefix or suffix of $X_1 \cdots X_n$ will be blank.

  ID's are used to describe moves of the TM. If $\delta(q, X_i) = (p, Y, L)$, then $X_1 \cdots X_{i-1} q X_i X_{i+1} \cdots X_n \vdash X_1 \cdots X_{i-2} p X_{i-1} Y X_{i+1} \cdots X_n$. If $i = 1$, we get $pBYX_2 \cdots X_n$. If $i = n$ and $Y = B$, we get $X_1 \cdots X_{n-2} p X_{n-1}$. $\vdash^*$ is used for zero or more moves.

- Example 1: $L = \{0^n 1^n \mid n \geqslant 1\}$



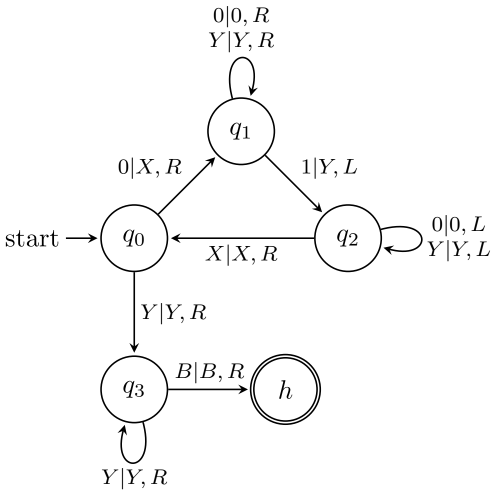


Figure 1: TM for $\{0^n 1^n \mid n \geqslant 1\}$

  - Formally, the TM is $M = (\{q_0, q_1, q_2, q_3, h\}, \{0, 1\}, \{0, 1, X, Y, B\}, \delta, q_0, B, \{h\})$. δ is given by the state transitions, and h is the *halt state*. A left-bound is assumed on the tape. As mentioned earlier, this has no effect on the computational power.
  - Starting at the left end, the loop $q_0 - q_1 - q_2 - q_0$ changes a 0 to X, and moves right over 0's and Y's. Then, it changes a corresponding 1 to Y, and moves left over Y's and 0's. In this process, if some other symbol is encountered, the machine 'crashes'. After finding an X, it checks if a 0 is present at its immediate right. If present, the entire loop repeats. Else, M moves to q_3 (if the symbol is a Y). Then, after moving right over several Y's, it accepts on a B.
  - Note: Merging q_0 and q_3 changes the language. The resulting TM would accept strings such as $0101 \notin L$ too.
  - To accept $L \cup \{\varepsilon\}$, an additional transition $\delta(q_0, B) = (h, B, R)$ can be added.

- Example 2: $L = \{a^n b^n c^n \mid n \geqslant 1\}$

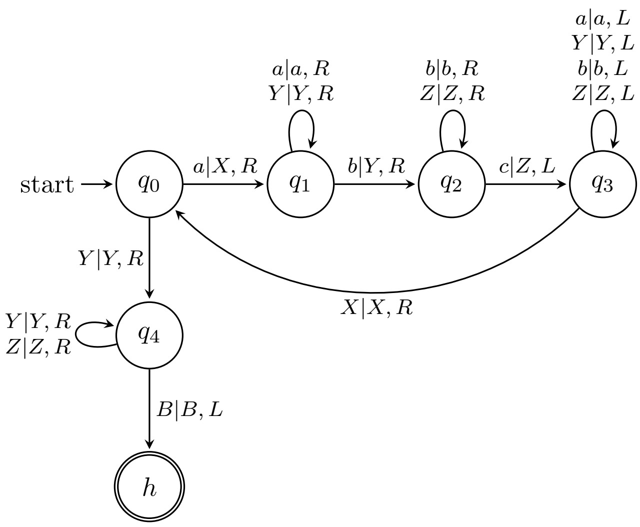


Figure 2: TM for $\{a^n b^n c^n \mid n \geqslant 1\}$

This follows from the previous example. Here, the loop $q_0 - q_3 - q_0$ marks corresponding input symbols with X, Y, Z. Then, we accept on the path $q_0 - q_4 - h$.

3 Computations with Turing Machines

- Suppose we wish to compute an n-ary function on a TM. The arguments (which are assumed to be integers) are provided on the tape in unary format. The unary alphabet is $\{0\}$ and we separate the arguments using 1's.
- The function's return value is written on the tape when it finishes execution.
- Example 1: Consider the *monus* function, defined as $f(x_1, x_2) = x_1 \dot{-} x_2 = \max(0, x_1 - x_2)$. The following machine computes this function:

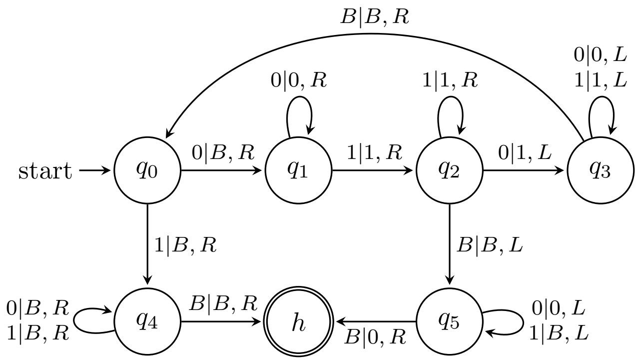


Figure 3: TM for *monus* function

This machine cancels out 0's from the arguments one by one, till one of them runs out (this is the loop $q_0 - q_3 - q_0$). The left branch $q_0 - q_4 - h$ deals with $x1 \leqslant x2$, resulting in a blank tape, while $q_2 - q_5 - h$ deals with $x_1 > x_2$. In either case, M halts with $0^{x_1 \dot{-} x_2}$ on the tape, blanking out everything else.

- Example 2: Consider a TM that performs addition on positive numbers x_1 and x_2. To do this, we simply modify the separator 1 to 0, and blank out the last 0. This results in a block of $x_1 + x_2$ contiguous 0's.

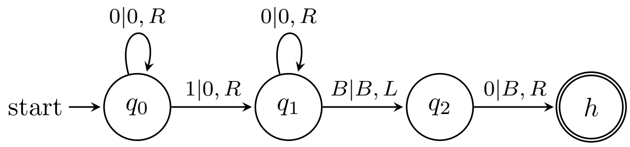


Figure 4: TM for unary adder

4 References

- Section 8.2 of Hopcroft, Motwani, Ullman.