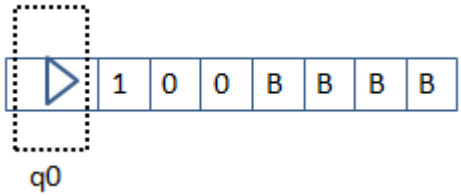
Advanced Algorithms and Complexity : Lecture 1 The Complexity Class P

August 3, 2018

Alphabets are finite sets of symbols. Examples: the binary alphabet $\{0, 1\}$
Strings are concatenation of zero or more symbols from alphabet. Examples are : ϵ (string of length 0),$0, 10, 00, 000, 101, 100, 110, ...$ Length of a string w is denoted by $|w|$. For example $|\epsilon| = 0$ and $|110| = 3$.
Languages are sets of strings. For example, we can define a language L_e as the set of all binary strings ending in 0: $L_e = \{0, 10, 00, 000, 010, 100, 110, ...\}$
Turing Machines are a formal model of computation.
Example of a TM: We will design a TM for accepting the language L_e as follows: TM's have a tape which is divided into cells. For simplicity we are considering a single-tape TM. We can also have multi-tape TMs having more than one tape. Again, for simplicity we are assuming that the tape is semi-infinite tape which starts from a cell having a special start symbol which we denote by $\triangleright$. We can also have $2-$way infinite tape, and also $2-$dimensional tapes. TM has a head that scans a single cell at a time. Initially it scans the start symbol. In any move of the TM, the head can either move Left (L), Right (R) or Stay (S) at the current cell. TMs can be in one of possible (finite) states. Initially, the TM is in the start state which we denote by q_s. With each move TMs can change the state. There is a state called halting state which we denote by q_h. When TM is in q_h state, then there is no further move possible and we say that the TM has accepted its input. If the TM halts in any other state other than q_h, then we say that the TM has rejected

its input. TMs can be thought of as an algorithm that takes some input on its tape and gives some output (yes if it accepts, no if it rejects). The 'code' of a TM is the transition function δ which specifies how the TM changes its state, moves its head, and modifies the current cell on a particular state and the current cell content. For the example language L_e we have:

$L_e = \{0, 00, 10, 000, 010, 100, 110, ...\}$ Initially the TM on input 100 will be:



We can easily design the transition function δ for this TM as follows:
$\delta(q_0, \triangleright) = (q1, \triangleright, R)$
$\delta(q_1, 0) = (q_1, 0, R)$
$\delta(q_1, 1) = (q_1, 1, R)$
$\delta(q_1, B) = (q_2, 0, L)$
$\delta(q_2, 0) = (q_n, 0, S)$
We can run this TM with input 100 and see that it halts in state q_h. Therefore the TM accepts the input 100. On input 101, it will halt in state q_2. Therefore the TM accepts the language L_e.
The example TM that we have considered is a deterministic TM (DTM) because each transition function specifies exactly one possible move. We can also have a Non-Deterministic TM (NTM) in which the transition function specifies more than one possible moves.
Formally we can define a DTM (one-tape) as a $3-$tuple (Γ, Q, δ) where Γ is the finite set of possible tape symbols (including the input alphabet and

the symbols $\triangleright$ and B). Q is the finite set of states (including q_s and q_h). $\delta : Q \times \Gamma \rightarrow (Q, \Gamma, \{L, R, S\})$ is the transition function specifying the moves of the DTM. We say that the DTM accepts a language L if for any x, $x \in L \implies$ DTM halts in state q_h when started with input x with initial state q_s and head scanning the $\triangleright$ symbol. $x \notin L \implies$ DTM halts in state $q \in Q - q_h$ when started with input x with initial state q_s and the head scanning the $\triangleright$ symbol.

Time Complexity of DTMs: For the example DTM, we observe that it makes $\Theta(n)$ moves on input of size n. We say that its time-complexity is $\Theta(n)$. Time-complexity of a DTM is independent of the input (whether it belongs to the language on not). It only depends on the input size n.

The time complexity class DTime: Let $T : \mathbb{N} \rightarrow \mathbb{N}$ be some function. A Language $L \in \text{Dtime}(T(n)) \iff \exists$ DTM that runs in time $O(T(n))$ and accepts L.

The time complexity class P: $P = \cup_{c \geq 1} \text{DTime}(n^c)$
The time complexity class P is defined for languages (decision problems with Yes/No as output).

Some examples of problems in P: Given two integers x and y of n bits each, we can multiply them in $O(n^2)$ time on a RAM TM (TM with RAM: Random Access Memory). A $T(n)-$ time RAM TM can be simulated in $T(n^2)$ time by a multitape DTM. This will not change the time complexity class P whether we define it by using a multitape DTM or by using a RAM TM. So for proving that a given language belongs to the class P, it is sufficient to give a polynomial time algorithm (RAM TM) for the language.
The multiplication probelm is: given x and y (integers) find an integer z such that $z = xy$. This is not a decision problem. We create a language L_{mult} as follows: $L_{\text{mult}} = \{(x, y, z) \mid x, y, z \text{ are binary integers such that } z = xy\}$.
For example $(10, 11, 110) \in L_{\text{mult}}$ and
$(10, 11, 111) \notin L_{\text{mult}}$
because $(10)_2 \times (11)_2 = (110)_2$ and
$(10)_2 \times (11)_2 \neq (111)_2$
A RAMTM for accepting L_{mult}: calculate $z' = xy$ in time $O(n^2)$ and check

whether $z' = z$ in time $O(n)$ with total time complexity as $O(n^2)$
$\implies L_{\text{mult}} \in P$.

P is closed under complementation ($P = Co - P$): $Co - P$ is defined as a time complexity class as follows:
We say that a language $L \in Co - P$ if $\overline{L} \in P$, $Co - P$ is not the complement of the class P ($\overline{P}$).
For example: $L_{\text{mult}} \in P \implies \overline{L_{\text{mult}}} = \{(x, y, z) \mid x, y, z \text{ are integers such that } z \neq xy\} \in Co - P \iff \overline{L} \in P$. $L \in P \implies \exists$ DTM T accepting L in polynomial time. We modify T as follows: convert q_h into a new non-halting state q'. When T halts on an input in a non-halting state (original states), then add a move so that T halts in q_h state. Now the new DTM T' has the property that it has the same time-complexity as T and also that whenever T accepts an input, T' rejects it, and whenever T rejects an input, T' accepts it
$\implies T'$ accepts $\overline{L}$ in polynomial time $\implies \overline{L} \in P$.

$Co - P \subseteq P$: Let $L \in Co - P$. We will prove that $L \in P$ as follows: $L \in Co - P \iff \overline{L} \in P \implies \exists$ DTM T accepting $\overline{L}$ in polynomial time. Now we modify T to get a DTM T' as before that accepts L in polynomial time $\implies L \in P$.
Example RAM TM for $\overline{L_{\text{mult}}}$: On input (x, y, z), calculate $z' = xy$ in time $O(n^2)$. Compare z with z' in time $O(n)$ and accept the input if $z \neq z'$, otherwise reject the input. The total time complexity of the RAM TM is $O(n^2) \implies \overline{L_{\text{mult}}} \in P$.