Advanced Algorithms and Complexity : Lecture 2 The Complexity Class NP

August 6, 2018

Non deterministic TM (NTM): NTM's are generalization of DTM in which at each step the TM may have more than one possible choice of moves. If this is the case, then it will select one possible next move, and the moves will continue in this way. Thus, at each step, NTM makes a "guess" of next move out of the possible choices. There is one important property of NTM's that they always make correct "guess". By "correct" guess we mean that the TM will select a move from which it is possible to accept the input (if there is any possibility of accepting the input). The selection of next moves is biased towards acceptance and we assume that the TM always "knows" the correct next move to take without any lookahead of future moves.

Formally, we can define an NTM (one-tape) as a 3-tuple (Γ, Q, δ) where Γ is the finite set of tape symbols (including the input alphabet and the symbols $\triangleright$ and B). Q is the finite set of states (including q_0 and q_h). $\delta : Q \times \Gamma \rightarrow (Q, \Gamma, \{L, R, S\})^*$ is the transition function specifying the possible moves of NTM. We say that the NTM accepts a language L if for any $x : x \in L \Rightarrow \exists$ choice of moves for NTM such that it halts in state q_h when started with input x with initial state q_0 and head scanning $\triangleright$. $x \notin L \implies$ NTM halts in state $q \in Q - q_h$ for all possible moves when started with input x with initial state q_0 and head scanning $\triangleright$.

Time-complexity of NTM: We say that a given NTM M runs in $T(n)$ time if for every input $x \in \{0,1\}^*$ and every sequence of non deterministic choices, M reached a halting state (either accepting or non-accepting) within T($|x|$) steps.

NTime($T(n)$): For every function $T : \mathbb{N} \to \mathbb{N}$ and $L \subseteq \{0,1\}^*$ we say that $L \in \text{NTime}(T(n))$ if $\exists$ NTM M such that M accepts L and it runs in time $O(T(n))$.
The time complexity class NP is defined as $NP = \cup_{c \in \mathbb{N}} \text{Ntime}(n^c)$.

Example of NTM: Now we will generalize the DTM for L_e = {0, 00, 10, 000, 010, 110, ...} into an NTM as follows:
$\delta(q_0, \triangleright) = \{(q_1, \triangleright, R)\}$
$\delta(q_1, 0) = \{(q_1, 0, R), (q_3, 0, R)\}$
($(q_3, 0, R)$ is an incorrect guess. NTM will always ignore this choice for accepting inputs.)
$\delta(q_1, 1) = \{(q_1, 1, R)\}$
$\delta(q_1, B) = \{(q_2, B, L)\}$
$\delta(q_2, 0) = \{(q_h, 0, S)\}$

We can easily verify that this NTM accepts L_e. Its time complexity is $O(n) \implies L_e \in \text{Ntime}(T(n))$ and also $L_e \in NP$.

$P \subseteq NP$: Given any language $L \in P$, it will have a DTM M running in polynomial time. Since DTM is a special case of NTM, M can also be viewed as an NTM $\implies L \in NP \implies P \subseteq NP$.

Certificate definition of NP: P can be viewed as a set of languages representing problems that can be efficiently solved. NP can be viewed as a set of languages representing problems that can be efficiently verified given a possible solution. This can be seen in an alternative definition of NP using DTM that runs in polynomial time, takes as input x, and a "certificate" u which can be viewed as a possible "solution" to the problem "$x \in L$?". The DTM should work in polynomial time. This restricts the length of u as polynomial in $|x| : |y| = p(|x|)$ where $p()$ is a polynomial.

A Language $L \subseteq \{0,1\}^*$ is in NP if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$

and a polynomial-time DTM M (called the verifier for L) such that for every $x \in \{0,1\}^*, x \in L \iff \exists u \in \{0,1\}^{p(|x|)}$ such that M accepts (x, u).
If $x \in L$ and $u \in \{0,1\}^{p(|x|)}$ satisfy M accepts (x, u), then we call u a certificate for x (with respect to the language L and machine M).

Proof of $P \subseteq NP$ using certificate definition of NP: Let $L \in P \implies \exists$ polynomial-time DTM M such that $x \in L \iff M$ accepts x. This can be viewed as belonging to the class NP by setting $p(|x|) = 0 \implies L \in NP$.

Example of some problems in NP: We consider the independent set problem. We can create a language corresponding to the decision version of independent set problems as follows:

INDSET $= \{(G, k) \mid G$ is the adjacency matrix of an undirected graph having a subgraph of at least k vertices having no edge between them$\}$.

INDSET $\in NP$ by polynomial time NTM:

1. On input $(G_{n\times n}, k)$, append string of length n after the input by using the first non-deterministic choice as writing 0, and the second non-deterministic choice as writing 1.
2. Deterministically verify that the vertices corresponding to 1 make an independent set of size at least k.

Both steps 1 and 2 can be done in polynomial time using NTM.
If $(G_{n\times n}, k) \in$ INDSET, then step 1 will correctly guess an independent set of size at least k. NTM will verify it correctly and accept the input in step 2.
If $(G_{n\times n}, k) \notin$ INDSET, then every guess in step 1 will not be able to make an independent set of size at least k. NTM will reject the input in step 2 after verifying it to be either not an independent set, or an independent set of size less than k.

INDSET $\in NP$ by using polynomial-time DTM that takes cartificates as input: On input $(G_{n\times n}, k)$ and the certificate u, the DTM verifies that $|u| = n$ and that the vertices corresponding to 1 make an independent set of size at least k.

If $(G_{n \times n}, k) \in$ INDSET then there exists an independent set of size at least k. Then u which encodes this independent set will make the DTM accept the input $((G_{n \times n}, k), u)$.

If $(G_{n \times n}, k) \notin$ INDSET then there does not exist any independent set of size at least k. Then every possible encoding of u will not be able to make the DTM accept the input $((G_{n \times n}, k), u)$.

We can clearly see that the DTM runs in polynomial time, and that the length of the certificate $= |u| = n$ is also a polynomial $\implies$ INDSET $\in NP$.

The two definitions of NP (using NTM $N(x)$ and DTM $D(x, u)$) are equivalent:

Suppose we are given a polynomial-time NTM $N(x)$. The DTM $D(x, u)$ will simulate $N(x)$ where u will be the encoding of non-deterministic choices of $N(x)$ runs in polynomial-time $\implies$ $|u|$ is a polynomial.

Suppose we are given a DTM $D(x, u)$ that runs in polynomial-time and $|u|$ is a polynomial. On input $x, N(x)$ will non-deterministically guess u in polynomial time ($|u|$ is polynomial) and then it will simulate $D(x, u)$ in polynomial-time. Also, the simulation of $N(x)$ by $D(x, u)$ in the previous case will run in polynomial-time since $N(x)$ runs in poly-time.