Lecture 27
ECE664

Polynomial Transformability of Problems in Class NP

AVI KAK
PURDUE U.

- First we note that $P \subseteq NP$. That is because every decision problem that can be solved in polynomial time by a deterministic algorithm can also be solved in polynomial time by a nondeterministic algorithm.

  Consider, for example, the decision problem whether an integer is divisible by 4. We can easily devise a DTM to solve this in a number of steps that is polynomial function of the size of the integer. (The size of the integer N is $\log_2 N$ since that is how many bits it takes to represent N.) The DTM will check the last two bits of the integer. If they are both zero, the answer is 'yes'; otherwise, the answer is 'no'. The same problem can be solved in polynomial time by an NDTM. The Guessing Module will write down either '1' for the 'yes' answer or '0' for the 'no' answer. The checking module will then check in polynomial time whether the guess is consistent with the last two bits of the integer.

- At the same time we believe that $NP - P$ is not empty since there exist several problems that do not lend themselves to solution by polynomial time deterministic algorithms. Therefore, at this point our picture of class NP looks like

- To better understand $NP - P$, we now introduce the notion of polynomial transformability between the problems in class NP. (Polynomial transformability is also referred to as polynomial reducibility.)

- DEFINITION: A polynomial transformation from a language $L_1 \subseteq \Sigma_1^*$ to a language $L_2 \subseteq \Sigma_2^*$ is a function

$$\boxed{f: \Sigma_1^* \rightarrow \Sigma_2^*}$$

  that satisfies the following two conditions: (1) There is a polynomial time DTM program that computes $f(x)$ for every $x \in \Sigma_1^*$; and (2) For all $x \in \Sigma_1^*$, $x \in L_1$ iff $f(x) \in L_2$

- NOTATION: If it is possible to construct a polynomial transformation from language L_1 to language L_2, we denote that fact by

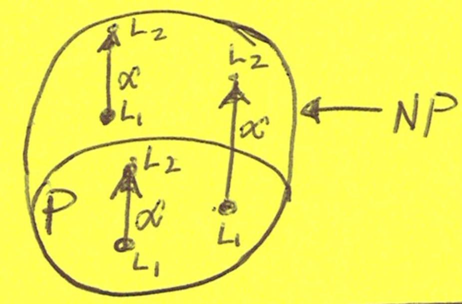
$$\boxed{L_1 \propto L_2}$$

  We may also depict this pictorially by

  where the direction of the arrow indicates that L_1 is being transformed to L_2. The fact that L_2 is displayed above L_1 is supposed to convey the idea that L_2 is possibly a larger language compared to L_1. The lemma that follows conveys the same thing but from the standpoint of the relative complexity of the two languages.

- **LEMMA:** $\boxed{\text{If } L_1 \propto L_2 \text{, then } L_2 \in P \text{ implies } L_1 \in P \text{, and equivalently, } L_1 \notin P \text{ implies } L_2 \notin P.}$

This diagram makes the lemma clear:



- **DEFINITION OF $\Pi_1 \propto \Pi_2$:**

If Π_1 and Π_2 are decision problems, with associated encoding schemes e_1 and e_2, we will write $\boxed{\Pi_1 \propto \Pi_2}$ whenever

$$L(\Pi_1, e_1) \propto L(\Pi_2, e_2)$$

$$\Pi_1 \xrightarrow{\propto} \Pi_2$$

- We can also interpret $\Pi_1 \propto \Pi_2$ more directly as a polynomial transformation $f: D_{\Pi_1} \to D_{\Pi_2}$ that satisfies the two conditions: ① $f(I)$ is computable in polynomial time for every $I \in D_{\Pi_1}$, and, ② for all $I \in D_{\Pi_1}$, we have $I \in Y_{\Pi_1}$ iff $f(I) \in Y_{\Pi_2}$.

An example of polynomial transformation from one problem to another in class NP:

Our example will show a polynomial transformation from the HAMILTONIAN CIRCUIT (HC) problem to the TRAVELING SALESMAN (TS) problem.

HC	TS
HAMILTONIAN CIRCUIT INSTANCE: A graph $G = (V, E)$ (vertices, edges) QUESTION: Does G contain a Hamiltonian circuit? Note: Suppose $V = \{v_1, v_2, \ldots v_k\}$ is the set of vertices in G. A simple circuit in G is the sequence $(v_{i_1}, v_{i_2}, \ldots, v_{i_m})$ such that $(v_{i_p}, v_{i_{p+1}}) \in E$ for every $p = 1, 2, \ldots m-1$ and such that $(v_{i_m}, v_{i_1}) \in E$. A Hamiltonian circuit is a simple circuit that connects all the vertices	TRAVELING SALESMAN INSTANCE: A finite set $C = \{c_1, c_2, \ldots c_m\}$ of cities and the distances $d(c_i, c_j) \in Z^+$ between every pair $c_i, c_j \in C$ of the cities, and a bound $B \in Z^+$. QUESTION: Is there a tour of all the cities having a total length of B or less? Each vertex can appear only once in a HC. A graph like ⋈ does not have an HC.

- To establish $HC \propto TS$, we must specify a function f that maps each instance of HC into an instance of TS and that satisfies the two conditions ① and ② stated above. This function f is specified as follows: Suppose $G = (V, E)$ with $|V| = m$ is a given instance of HC. The corresponding instance of TS has a set C of cities that is the same as V. As for the intercity distances, we set $d(v_i, v_j) = 1$ if $(v_i, v_j) \in E$ and $d(v_i, v_j) = 2$ otherwise. Finally, we set $B = m$. It is easy to argue that if ~~the~~ an instance of HC contains a Hamiltonian circuit, then the corresponding instance of TS will contain a tour of length m, and vice versa. It is shown trivially that the mapping from HC to TS can be computed in polynomial time.

- So we have proved $\mathrm{HC} \xrightarrow{\propto} \mathrm{TS}$, which implies that $\boxed{\text{TS is at least as hard as HC.}}$ Also implied is the fact that $\boxed{\text{HC is no harder than TS}}$

- Transitivity of $\propto$: If $\Pi_1 \propto \Pi_2$ and $\Pi_2 \propto \Pi_3$, then $\Pi_1 \propto \Pi_3$