Thm 5.16 $P \subseteq NP \subseteq EXP$.

Proof:

1. $P \subseteq NP$

Let $L \in P$. We shall prove $L \in NP$.

Let $p(n) = 0$, and let M be a TM that decides L efficiently.

If $x \in L$, then $M(x, \epsilon) = 1$. If $x \notin L$, then $M(x, \epsilon) = 0$.

2. $NP \subseteq EXP$

Let $L \in NP$. We will prove $L \in EXP$.

Since $L \in NP$, there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM M s.t. 1,2 hold.

Construct a TM M' which enumerates all $w \in \{0,1\}^{p(|x|)}$ and check if $M(x, w) = 1$.

If there exists one w s.t. $M(x, w) = 1$, then M accepts x.

The total running time is $2^{p(n)} \cdot poly(n) = 2^{n^{O(1)}} \cdot n^{O(1)} = 2^{n^{O(1)}}$

- $poly(n) = n^{O(1)}$

Nondeterministic TMs

At any point, the machine may proceed according to severl possibilities. The transition function

$$\delta : Q \times \Gamma \to P(Q \times \Gamma \times \{L, R, S\})$$

Formally, an NTM $M = (\Sigma, \Gamma, Q, \delta, q_0, q_{accept}, q_{reject})$

M **accepts** x **if there is a computation path that accepts** x.

Def 5.17 **Say an NTM** M **runs in time** $T(n)$ **if for exery input** $x \in \{0,1\}^n$**, and every sequence of nondeterministic choice,** M **reaches an end state in** $T(n)$ **steps.**

Def 5.18 **Say an NTM** M **decides** L **if for every** $x \in \{0,1\}^*$, $x \in L \Leftrightarrow M(x) = 1$

Def 5.19 **(Binary-choice NTM)**

$$M = (Q, \Sigma, \Gamma, \delta_1, \delta_2, q_0, q_{accept}, q_{reject}),$$

where $\delta_1, \delta_2 : Q \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$

At each step, M **applies** δ_1 **or** δ_2 **arbitrarily.**

Lem 5.20 **Let** $L \subseteq \{0,1\}^*$**. If** L **can be decided by an NTM** M **in time** $T(n)$**, then** L **can be decided by a binary-choice NTM in time** $O_M(T(n))$**.**

Proof: At each step, M can have at most $2^{|Q| \times |\Gamma| \times 3}$ choices. This can be simulated in $\log_2 2^{|Q| \times |\Gamma| \times 3} = |Q| \times |\Gamma| \times 3$ steps by a binary-choice NTM.

So, the total running time is $\leq |Q| \times |\Gamma| \times 3 \times T(n) = O_M(T(n))$. *Q.E.D.*

Def 5.21 **Let $T : \mathbb{N} \to \mathbb{N}$. Let $NTIME(T(n))$ be the set of languages that can be decided by an NTM in time $O(T(n))$.**

Thm 5.22 $NP = \underset{c\geq 1}{\cup} NTIME(n^c)$

Proof:

1. $(\underset{c\geq 1}{\cup} NTIME(n^c) \subseteq NP)$ Let $L \in \underset{c\geq 1}{\cup} NTIME(n^c)$, we will prove $L \in NP$.

   Since $L \in NTIME(n^c)$, there exists a binary-choice NTM N that decides L in time $d \cdot n^c$, where $d > 0$ is a constant. Let $p(n) = d \cdot n^c$, and let the certificate $w \in \{0,1\}^{p(n)}$ indicates which transition function to apply. The verifier checks if N accepts x (given the certificate).
2. $(NP \subseteq \underset{c\geq 1}{\cup} NTIME(n^c))$ Let $L \in NP$. We will prove $L \in \underset{c\geq 1}{\cup} NTIME(n^c)$.

   ...... On input x, construct an NTM as follows:

   1. Nondeterministically guess $w \in \{0,1\}^{p(|x|)}$
   2. Simulate M on input (x, w). Accept if and only if M accepts (x, w).

   It is clear that N runs in polynomial time, and N decides L. *Q.E.D.*

Def 5.23 **(Karp Reduction) Let $L, K \subseteq \{0,1\}^*$. Say L is Karp-reducible to K, denoted by $L \leq_P K$, if there exists a polynomial-time TM M s.t. for all $x \in \{0,1\}^*$, $x \in L \Leftrightarrow M(x) \in K$.**

- $L \leq_P K \Rightarrow$ **If $K \in P$, then $L \in P$. If $L \notin P$, then $K \notin P$.**

Lem 5.24 **If $L_1 \leq_P L_2$ and $L_2 \leq_P L_3$, then $L_1 \leq_P L_3$.**

- Let $M_3 = M_2(M_1(x))$, which also runs in polynomial time. We can check $x \in L_1 \Leftrightarrow M_3(x) \in L_3$.

Def 5.25 **(NP-Hard) $L \subseteq \{0,1\}^*$ is NP-hard if for all language $K \in NP$, $K \leq_P L$.**

Def 5.26 **(NP-Complete) L is NP-complete if L is in NP, and L is NP-hard.**

- $NP\text{-}complete = NP \cap NP\text{-}hard$

Lem 5.27 **If L is NP-hard, and $L \in P$, then $P = NP$.**

Lem 5.28 **Let $L \in$ NP-complete. Then $L \in P \Leftrightarrow P = NP$.**

Cook-Levin Theorem

- **Cook(1971), Levin(1973) proved the first NP-complete problem SAT.**

SAT (Boolean Satisfiability Problem)

- Variable: $x, y, z, ...$ can take TRUE or FALSE.
- Literal: A variable or its negation. eg. $x, \neg x, y, \neg y, ...$
- Clause: OR(Disjunction) of one or more literals. eg. $\neg x \vee y, \neg y \vee z, ...$
- Formula: AND(Conjunction) of one or more clauses.

$SAT = \{< \phi > | \phi \; is \; satisfiable\}$

Thm 5.29 **SAT is NP-complete.**