Theory of Computation: Mathematical study of computing machines, their fundamental capabilities and their limitations, Q1: What problems are solvable, in PRACTICE, by computer and what problems are not? \Rightarrow Complexity theory answers these questions. Note that, in human history, no reduction has yet been found for $L_1 \leq_p L_2$, where $L_1 \in \mathbf{P}$, $L_2 \in \mathbf{NPC}$. Q2: What problems are solvable, in PRINCIPLE, by computer and what problems are not? $P=NP? \text{ or } P \neq NP?$ ⇒ Computability theory answers these questions. less bounded (powerful expression) L RE Recursively UTM M_U halts on input "M" "w" not algorithm LRL **Enumerable Languages** $\leq_p | \mathsf{NPC} |$ iff M halts on w. L NP Halting Problem The class **NPC** consists of all decision problems L such that LP Running time O(|w|!)Recursive Languages $1.L \in \mathbf{NP}$ Verification time $O(|w|^c)$ $\overline{3}$ -SAT LCFL 2.for every $L' \in \mathbf{NP}, L' \leq_p L$. algorithm NP ◀ $L[\mathsf{Reg}]$ Running time $O(|w|^c)$ Reduction $L_1 \leq_p L_2$ Integer Factorization $L_{1\searrow M_1}$ Read/write head NP-Complete b 3-SAT а а (halting) Turing Machine (unlimited memory) $\{a^n b^n c^n : n \ge 0\}$ f: polynomial time reduction from L_1 to L_2 Context-Free Pushdown Automata а Finite control grammar $-q_1$ Languages Stack (=limited memory $G: S \to aSb$ ynRegular language (Non) Deterministic Finite Automata Languages^{*} word Reading head Run $\{a\}$ Build alphabet Input Automatic (Manual) $\{aba, aabb, 000, 01\#10, \ldots\}$ L $aba, a, 0101, \dots \mathcal{W}$ **expressed** within a *finite* amount of space and time. **proceeds** through a *finite* number of states. $a, b, z, 0, 1, \#, \$, \dots \sum$ more bounded (weak expression)