The class NP

**Definition:** \( \text{TIME}(t(n)) = \{L : \text{there exists a TM M that decides } L \text{ in time } O(t(n))\} \)

\[ P = \bigcup_{k \geq 1} \text{TIME}(n^k) \]

**Definition:** \( \text{NTIME}(t(n)) = \{L : \text{there exists a NTM M that decides } L \text{ in time } O(t(n))\} \)

\[ \text{NP} = \bigcup_{k \geq 1} \text{NTIME}(n^k) \]

### NP in relation to P and EXP

- \( P \subseteq \text{NP} \) (poly-time TM is a poly-time NTM)
- \( \text{NP} \subseteq \text{EXP} \)
  - configuration tree of \( n^k \)-time NTM has \( \leq b^n \) nodes
  - can traverse entire tree in \( O(b^n) \) time

*we do not know if either inclusion is proper*

### Poly-time verifiers

- \( \text{NP} = \{L : \text{L is decidable by a NTM}\} \)
- Very useful alternate definition

**Theorem:** language \( L \) is in \( \text{NP} \) if it is expressible as:

\[ L = \{ x \mid \exists y, |y| \leq |x|^k, (x, y) \in R \} \]

where \( R \) is a language in \( P \).
- poly-time TM \( M_R \) deciding \( R \) is a “verifier”

### Poly-time verifiers

- Example: \( 3\text{SAT} \) expressible as

\[ 3\text{SAT} = \{ \varphi : \varphi \text{ is a 3-CNF formula for which } \exists \text{ assignment A for which } (\varphi, A) \in R \} \]

\[ R = \{ (\varphi, A) : A \text{ is a sat. assign. for } \varphi \} \]

- satisfying assignment \( A \) is a “witness” of the satisfiability of \( \varphi \) (it “certifies” satisfiability of \( \varphi \))
- \( R \) is decidable in poly-time
Poly-time verifiers

$L = \{ x \mid \exists y, |y| \leq |x|^k, (x, y) \in R \}$

**Proof**: $(\Leftarrow)$ give poly-time NTM deciding $L$

phase 1: "guess" $y$ with $|x|^k$ nondeterministic steps

phase 2: decide if $(x, y) \in R$

---

**Proof**: $(\Rightarrow)$ given $L \in \text{NP}$, describe $L$ as:

$L = \{ x \mid \exists y, |y| \leq |x|^k, (x, y) \in R \}$

$L$ is decided by NTM $M$ running in time $n^k$

define the language

$R = \{ (x, y) : y$ is an accepting computation history of $M$ on input $x \}$

check: accepting history has length $\leq |x|^k$

check: $M$ accepts $x$ iff $\exists y, |y| \leq |x|^k, (x, y) \in R$

---

**Cook-Levin Theorem**

- Gateway to proving lots of natural, important problems NP-complete is:

**Theorem** (Cook, Levin): 3SAT is NP-complete.

- Recall: $3\text{SAT} = \{ \varphi : \varphi$ is a CNF formula with 3 literals per clause for which there exists a satisfying truth assignment $\}$

---

**Boolean Circuits**

- Boolean circuit $C$
  - directed acyclic graph
  - nodes: AND (\&); OR (\lor); NOT (\neg); variables $x_i$
  - $C$ computes function $f:\{0,1\}^n \rightarrow \{0,1\}$ in natural way
  - identify $C$ with function $f$ it computes
  - size = # nodes

- every function $f:\{0,1\}^n \rightarrow \{0,1\}$ computable by a circuit of size at most $O(n2^n)$
  - AND of $n$ literals for each $x$ such that $f(x) = 1$
  - OR of up to $2^n$ such terms
CIRCUIT-SAT is NP-complete

**Theorem:** CIRCUIT-SAT is NP-complete

CIRCUIT-SAT = {C : C is a Boolean circuit for which there exists a satisfying truth assignment}

**Proof:**

- Part 1: need to show CIRCUIT-SAT ∈ NP.
  - can express CIRCUIT-SAT as:
    \[ CIRCUIT-SAT = \{ C : C \text{ is a Boolean circuit for which } \exists x \text{ such that } (C, x) \in R \} \]
  - \[ R = \{(C, x) : C \text{ is a Boolean circuit and } C(x) = 1\} \]

- Part 2: for each language \( A \in \text{NP} \), need to give poly-time reduction from \( A \) to CIRCUIT-SAT
  - for a given language \( A \in \text{NP} \), we know \( A = \{ x | \exists y, |y| \leq |x|^k, (x, y) \in R \} \) and there is a (deterministic) TM \( M_R \) that decides \( R \) in time \( g(n) \leq n^c \) for some \( c \).

- **Tableau** (configurations written in an array) for machine \( M_R \) on input \( w = (x, y) \):
  - height = time taken = \( |w|^c \)
  - width = space used \( \leq |w|^c \)

- Important observation: contents of cell in tableau determined by 3 others above it:

- Can build Boolean circuit \( \text{STEP} \)
  - input (binary encoding of) 3 cells
  - output (binary encoding of) 1 cell
  - each output bit is some function of inputs
  - can build circuit for each
  - size is independent of size of tableau

- \( |w|^c \) copies of \( \text{STEP} \) compute row \( i \) from \( i-1 \)
CIRCUIT-SAT is NP-complete

This circuit $C_{M,w}$ has inputs $w_1 w_2 \ldots w_n$ and $C(w) = 1$ iff $M_w$ accepts input $w$. Size = $O(|w|^{2c})$.

3SAT is NP-complete

Theorem: 3SAT is NP-complete

3SAT = \{ $\phi$ : $\phi$ is a 3-CNF formula for which there exists a satisfying truth assignment \}

Proof:

- Part 1: need to show 3-SAT $\in$ NP
  - already done
- Part 2: need to show 3-SAT is NP-hard
  - we will give a poly-time reduction from CIRCUIT-SAT to 3-SAT

3SAT is NP-complete

- given a circuit $C$
  - variables $x_1, x_2, \ldots, x_n$
  - AND ($\land$), OR ($\lor$), NOT ($\neg$) gates $g_1, g_2, \ldots, g_m$
- reduction $f(C)$ produces these clauses for $\phi$ on variables $x_1, x_2, \ldots, x_n, g_1, g_2, \ldots, g_m$:

  $\neg g_i$

  $\{ (g_i \lor z) \}$

  $\{ (\neg z \lor \neg g_i) \}$

  $(z \iff \neg g_i)$
3SAT is NP-complete

– given a circuit C
  • variables $x_1, x_2, \ldots, x_n$
  • AND ($\land$), OR ($\lor$), NOT ($\neg$) gates $g_1, g_2, \ldots, g_m$
– reduction $f(C)$ produces these clauses for $\varphi$ on variables $x_1, x_2, \ldots, x_n, g_1, g_2, \ldots, g_m$:
  $$\land g \left\{ \begin{array}{l}
  \neg g \lor z_1 \\
  \neg g \lor z_2 \\
  \neg z_1 \lor \neg z_2 \lor g_i \\
  \end{array} \right\} \iff g_i$$

Search vs. Decision

• We want to talk about languages (or decision problems)
• Most search problems have a natural, related decision problem by adding a bound “$k$”; for example:
  – search problem: given $G$, find the largest independent set
  – decision problem: given ($G, k$), is there an independent set of size at least $k$