# Last class

1. **Nondeterministic TM**

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

2. $NP = \cup_{c\geq 1} NTIME(n^c)$

- An equivalent definitions of NP
    - $L \in NP \Leftrightarrow \exists$ *polynomial* $p : \mathbb{N} \to \mathbb{N}$ and $\exists$ *poly-time TM* $M$ s.t.
        1. $x \in L$ if $\exists w \in \{0,1\}^{P(|x|)}$ and $M(x, w) = 1$
        2. $x \notin L$ if $\forall w \in \{0,1\}^{P(|x|)}$ and $M(x, w) = 0$

3. **Karp Reduction** $L \leq_P K$: $\exists$ *poly-time TM* $M$ s.t. $\forall x \in \{0,1\}^*, x \in L \Leftrightarrow M(x) \in K$

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

4. **NP-hard**

- $L \in$**NP-hard** $\Leftrightarrow \forall K \in NP, K \leq_p L$

5. **NP-complete**

---

# Cook-Levin Theorem

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

## SAT (Boolean Satisfiability Problem)

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

$SAT = \{\langle\phi\rangle | \phi \ is \ satisfiable\}$

eg. $\phi = (\neg x \vee y) \wedge (x \vee \neg y) \wedge (x \vee y) \wedge (\neg x \vee \neg y)$ is not satisfiable.

---

*Proof:* To prove SAT is NP-complete, we need to show $\forall L \in NP, L \leq_P SAT$.

So, it suffices to show, for any poly-time NTM $M$, and for any input $x \in \{0,1\}^*$, we can algorithmically construct a formula $\phi_{M,x}$ (in polynomial time) s.t. $M$ accepts $x$ ($x \in L$) if and only if $\phi_{M,x} \in SAT$.

$$\overset{x}{\to} efficent \ TM \overset{\phi_{M,x}}{\to} SAT \ solver \to Yes/No$$

- Snapshot

| **0** | **1** | **1** | **0** | **1** | **0** | **0** | **...** |
|---|---|---|---|---|---|---|---|
| | | | $\uparrow Q_1$ | | | | |

| 0 | 1 | 1 | 1 | 1 | 0 | 0 | ... |
|---|---|---|---|---|---|---|---|
| | | | | $\uparrow Q_2$ | | | |

| 0 | 1 | 1 | 1 | 1 | 0 | 0 | ... |
|---|---|---|---|---|---|---|---|
| | | | | | $\uparrow Q_3$ | | |

- $T_{i,j,k} : i \in [1, T(n)], j \in \Gamma, k \in [1, T(n)]$: Cell $i$ contains symbol $j$ at step $k$.

  eg. in the snapshots above, $T_{1,0,3} = true, T_{1,1,3} = false, T_{1,\sqcup,3} = false$
- $H_{i,k} : i, k \in [1, T(n)]$: Head is on cell $i$ at step $k$.

  eg. in the snapshots above, $H_{4,3} = true, H_{1,3} = H_{2,3} = H_{3,3} = H_{5,3} = \ldots = false$
- $Q_{q,k}$: The head is on state $q$ at step $k$.
- In total, we have $T(n)^2 \times |\Gamma| + T(n)^2 + |Q| \times T(n) = O_M(T(n)^2)$ variables.

---

## Initialization

| $x_1$ | $x_2$ | $x_3$ | ... | $x_n$ | $\sqcup$ | $\sqcup$ | ... |
|---|---|---|---|---|---|---|---|
| $\uparrow$ | | | | | | | |

Input $x \in \{0,1\}^*$ is of length $n$.

$T_{i,x_i,0} = true$ for $i = 1, 2, \ldots, n$, $T_{i,\sqcup,0} = true$ for $i = n+1, n+2, \ldots, T(n)$. Otherwise $T_{i,j,0} = false$

$H_{1,0} = true$, $H_{i,0} = false$ for $i = 2, 3, \ldots, T(n)$

At step 0, the head is on the leftmost position.

The initial state is $q_0$. i.e. $Q_{q_0,0} = true$, $Q_{q,0} = false$ for every $q \in Q \setminus \{q_0\}$.

## Restrictions

1. At most one symbol per cell.

$$(\forall i \in [1, T(n)], \forall k \in [1, T(n)], \forall j \neq j' \in \Gamma)(\neg T_{i,j,k} \vee \neg T_{i,j',k})$$

2. At least one symbol per cell.

$$(\forall i, k \in [1, T(n)])(\vee_{j \in \Gamma} T_{i,j,k})$$

3. At most one state at a time.

$$(\forall k \in [1, T(n)], \forall q \neq q' \in Q)(\neg Q_{q,k} \vee \neg Q_{q',k})$$

4. At least one state at a time.

$$(\forall k \in [1, T(n)])(\vee_{q \in Q} Q_{q,k})$$

5. At most one head position at a time.

$$(\forall k \in [1, T(n)], \forall i \neq i' \in [1, T(n)])(\neg H_{i,k} \vee \neg H_{i',k})$$

6. At least one head position at a time.

$$(\forall k \in [1, T(n)])(\vee_{i\in[1,T(n)]} H_{i,k})$$

### Transition Rule

- $A \rightarrow B$ is equivalent to $\neg A \vee B$.

1. Cell remains unchanged unless written.

$$(\forall i \in [1, T(n)], \forall k \in [1, T(n) - 1], \forall j \neq j' \in \Gamma)(T_{i,j,k} \wedge T_{i,j',k+1} \rightarrow H_{i,k})$$

2. Transition function $\delta$.

$$(\forall i \in [1, T(n)], \forall k \in [1, T(n) - 1], \forall q \in Q, \forall j \in \Gamma)$$
$$(H_{i,k} \wedge Q_{q,k} \wedge T_{i,j,k} \rightarrow \vee_{(q',j',d)\in\delta(q,j), d\in\{-1,0,1\}}(H_{i+d,k+1} \wedge Q_{q',k+1} \wedge T_{i,j',k+1}))$$

### Halt in an accept state

Without loss of generality , there is a self loop in the accept state for $\forall x \in \Gamma$.

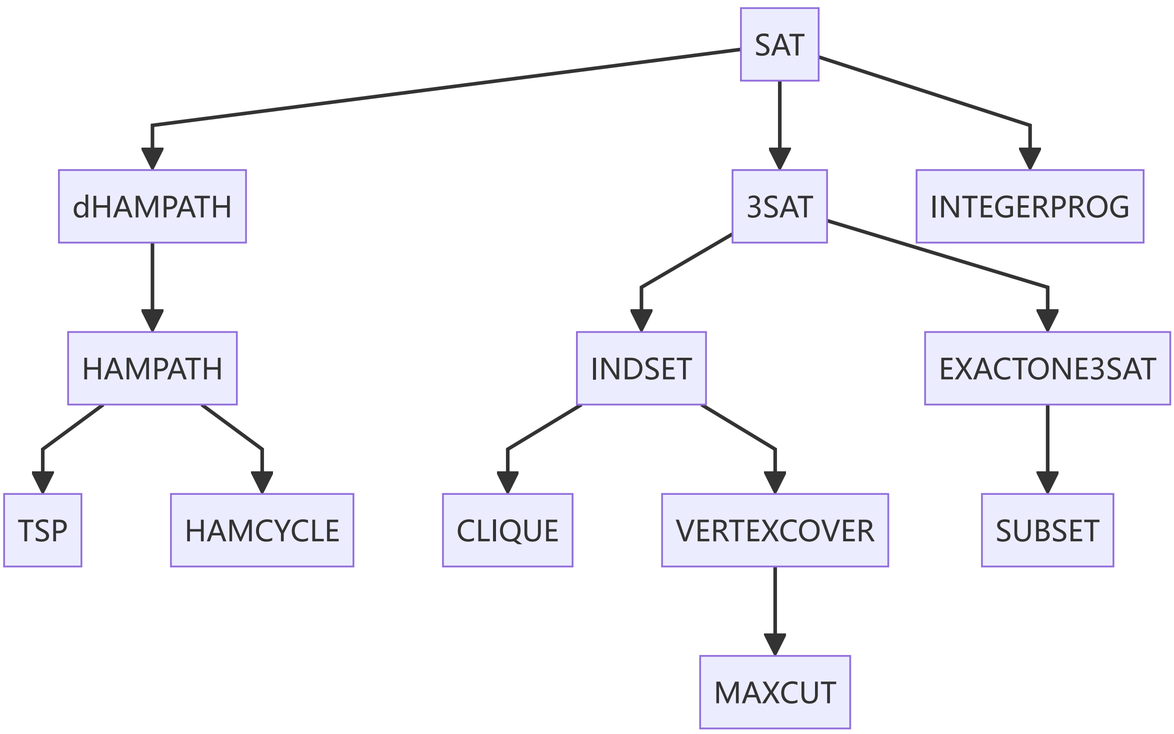
$$(\vee_{k\in[1,T(n)]} Q_{q_{accept},k})$$

Take the AND of all the above clauses. The number of clauses is $n^{O(1)}$.

### Goal

NTM $M$ accepts $x \Leftrightarrow \phi_{M,x} \in SAT$.

---




- In **3SAT**, each clause has (at most) 3 literals.
- eg. $\phi = (x \vee y \vee \neg z) \wedge (\neg x \vee \neg y \vee z) \wedge (x \vee \neg z)$

*Thm 5.30* $SAT \leq_P 3SAT$.

*Proof:*

- A **conjunctive normal form (CNF)** is the AND of several clauses. eg. $(x \vee y \vee \neg z) \wedge (\neg x \vee \neg y \vee z) \wedge (x \vee \neg z)$.
- A **3CNF** is a **CNF** where each clause has 3 literals.

Let $\phi$ be a CNF with $m$ clauses.

If each clause has $\leq 3$ literals, then we are done.

If some clause has $\geq 3$ literals, we replace it by an equivalent 3CNF.

Let $C = l_1 \vee l_2 \vee \ldots \vee l_k, k \geq 4$, where $l_i \in \{x_1, \bar{x_1}, \ldots, x_n, \bar{x_n}\}$.

Replace $C$ by

$$C' = (l_1 \vee l_2 \vee z_1) \wedge (\bar{z_1} \vee l_3 \vee z_2) \wedge (\bar{z_2} \vee l_4 \vee z_3) \wedge \ldots \wedge (\bar{z_{k-2}} \vee l_k \vee z_{k-1})$$

($z_1, z_2, \ldots, z_{k-2}$ are new variables)

Claim $C$ is satisfiable if and only if $C'$ is satisfiable. *Q.E.D.*

eg. $x_1 \vee x_2 \vee \bar{x_3} \vee x_5$ is satisfiable if and only if $(x_1 \vee x_2 \vee z) \wedge (\bar{z} \vee \bar{x_3} \vee x_5)$.

---

*Thm 5.31* $SAT \leq_P INTEGERPROG$.

- **INTEGERPROG**: $x_1, x_2, \ldots, x_n \in \{0, 1\}, a_{i,1}x_1 + a_{i,2}x_2 + \ldots + a_{i,n}x_n \geq a_{i,0}$

*Proof:* Let $\phi$ be a CNF on $n$ variables $x_1, \ldots, x_n$ with $m$ clauses.

For each variable $x_i$, introduce a variable $y_i$ in $INTEGERPROG$. Add constraints $o \leq y_i \leq 1$ for all $i$.

For each clause in $\phi$, sums up all literals (replacing $x_i$ by $y_i$, and replacing $\bar{x_i}$ by $1 - y_i$), and asserts the sum is at least 1.

eg. $x_1 \vee \bar{x_2} \vee \bar{x_3} \rightarrow y_1 + (1 - y_2) + (1 - y_3) \geq 1$. *Q.E.D.*

---

- $INTEGERPROG \in NPC$
- $01PROG \leq_P INTEGERPROG$
- $01PROG \in NPC$

---

- $INDSET = \{\langle G, k \rangle | G\ has\ an\ independent\ set\ of\ size\ k\}$.