NP-completeness

Peter Mayr

Computability Theory, November 27, 2023

Recall

- P ... problems that can be decided in polynomial time
- NP ... problems that can be verified in polynomial time
- $P \subseteq NP \subseteq EXPTIME$

One of the Millenium Problems

Is $P = NP$?

Reductions

Definition

Let $A, B \in \Sigma^*$. A **polynomial time many-one reduction** from A to B is a function $f\colon \Sigma^* \to \Sigma^*$ that is computable by a DTM in polynomial time such that

$$\forall x \in \Sigma^* : \ x \in A \text{ iff } f(x) \in B.$$

If a polynomial time many-one reduction from A to B exists, we write $A \leq_m^p B$.

Note

Logspace reductions $\leq_m^{\log}$, etc., are defined analogously.
Since L$\subseteq$ P, also $\leq_m^{\log} \subseteq \leq_m^p$.

Hard problems don't reduce to easy ones

Lemma

Let $A \leq_m^p B$.

1. If $B \in \mathrm{P}$, then $A \in \mathrm{P}$.
2. If $B \in \mathrm{NP}$, then $A \in \mathrm{NP}$.

Proof.

- Let f be a reduction from A to B that is computable in n^k time for some $k \in \mathbb{N}$.
- Then $|f(x)| \leq |x|^k$.
- Assume $B \in \mathrm{DTIME}(n^\ell)$ for some $\ell \in \mathbb{N}$.
- Then $f(x) \in B$ can be decided in time $|f(x)|^\ell \leq |x|^{k\ell}$.
- Thus $x \in A$ is decidable in $O(n^{k\ell})$ time.

□

The hardest problems in NP

Definition

B is **NP-hard** (with respect to $\leq_m^p$) if for all $A \in$ NP: $A \leq_m^p B$
B is **NP-complete** if B is NP-hard and $B \in$ NP.

Note

1. If some NP-complete problem is in P, then P=NP.
2. If A is NP-complete and $A \leq_m^p B$ for some $B \in$ NP then B is NP-complete.

Question

How to define "complete in P"?

Satisfiability of Boolean formulas

Definition

- A **Boolean formula** Φ is formed from variables $x_1, x_2, \dots$ and logical connectives $\wedge, \vee, '$ (negation).
- Φ is **satisfiable** if Φ is true for some assignment of its variables to $0, 1$ (false, true).
- SAT $:= \{\sharp(\Phi) \ : \ \Phi$ is a satisfiable Boolean formula $\}$

Example

$\Phi(x_1, x_2, x_3) = (x_1' \vee x_2') \wedge (x_2 \vee x_3)$ is satisfiable by e.g.
$x_1 \mapsto 0, x_2 \mapsto 0, x_3 \mapsto 1$

Cook-Levin Theorem (1971)

SAT is NP-complete.

Proof.

SAT $\in$ NP: If a satisfying assignment for Φ exists, it can be verified in polynomial time in $|\Phi|$.

Idea for hardness: For each $A \in$ NP construct a polytime reduction to SAT realizing the following correspondences:

- NP machine N on w $\leftrightarrow$ Boolean formula Φ
- accepting computational path for w $\leftrightarrow$ satisfying assigment

Let $A \in \mathrm{NP}$ be decided by a nondeterministic TM N in time n^k for some $k \in \mathbb{N}$.
Wlog N deletes its tape and moves to position 0 before halting.

Represent a computational path of N for input w of length n by the following $n^k \times (n^k + 3)$ table T of configurations with entries in $C := Q \cup \Gamma \cup \{\sharp\}$ (state is left of the cell with the tape head):

$\sharp s\, w_1 \dots w_n ␣$	$\dots$	$␣\sharp$	start configuration
$\sharp$		$\sharp$	2nd configuration
$\vdots$			$\vdots$
$\sharp$		$\sharp$	n^kth configuration

- Describe T by a Boolean formula Φ in variables x_{iju} for $1 \leq i \leq n^k,\ 1 \leq j \leq n^k + 3,\ u \in C$.
- Interpret $x_{iju} = \begin{cases} 1 & \text{if } T_{i,j} = u, \\ 0 & \text{else.} \end{cases}$

Define

$$\Phi := \Phi_{\text{cell}} \wedge \Phi_{\text{start}} \wedge \Phi_{\text{move}} \wedge \Phi_{\text{accept}}$$

such that Φ is satisfiable iff it describes an accepting computational path.

1. Each cell of T contains exactly one symbol from C:

$$\Phi_{\text{cell}} := \bigwedge_{i,j} \left((\bigvee_{u \in C} x_{iju}) \wedge \bigwedge_{u \neq v} (x_{iju} \wedge x_{ijv})' \right)$$

2. The first row contains the start configuration:

$$\Phi_{\text{start}} := x_{11\sharp} \wedge x_{12s} \wedge x_{13w_1} \wedge \dots$$

3. The accept state t of N occurs in T:

$$\Phi_{\text{accept}} := x_{n^k 2t}$$

4. Φ_{move} expresses that each row encodes the successor configuration of the previous.

To define Φ_{move} say a 2×3 subblock of T is **legal** if it is consistent with the transition function Δ of N (or copying a halting configuration).

E.g. if $\Delta(q, a) = \{(q', b, -1), \dots\}$, the following are legal:

c	q	a
q'	c	b

q	a	d
c	b	d

a	b	c
a	b	c

$\sharp$	t	$\sqcup$
$\sharp$	t	$\sqcup$

These are illegal:

a	b	b
a	a	b

$*$	$*$	$*$
q	$*$	q'

c	q	a
a	c	b

Let

Φ_{move} := all 2×3 subblocks of T are legal

$$= \bigwedge_{i,j} \bigvee_{\begin{array}{|c|c|c|}\hline c_1 & c_2 & c_3 \\ \hline c_4 & c_5 & c_6 \\ \hline \end{array} \text{ legal}} \left(\begin{array}{l} x_{i,j,c_1} \wedge x_{i,j+1,c_2} \wedge x_{i,j+2,c_3} \wedge \\ x_{i+1,j,c_4} \wedge x_{i+1,j+1,c_5} \wedge x_{i+1,j+2,c_6} \end{array} \right)$$

Claim.

If the top row of T represents the starting configuration of N and each 2×3 subblock is legal, then each row is the successor configuration of the previous (or the copy of the previous halting configuration)

Proof by induction on the rows of T.

- If a cell of T contains some $a \in \Gamma$ but is not next to a state, it is the center top of some legal 2×3 subblock

$*$	a	$*$
$*$	a	$*$

  and remains unchanged.

- Cells next to some state $q \in Q \setminus \{r, t\}$ occur in legal blocks

$*$	q	a
$*$	$*$	$*$

$\sharp$	r	$\llcorner\!\lrcorner$
$\sharp$	r	$\llcorner\!\lrcorner$

$\sharp$	t	$\llcorner\!\lrcorner$
$\sharp$	t	$\llcorner\!\lrcorner$

  and change according to the transition function Δ. □

This completes the proof that

$$w \in L(N) \text{ iff } \Phi = \Phi_{\text{cell}} \wedge \Phi_{\text{start}} \wedge \Phi_{\text{move}} \wedge \Phi_{\text{accept}} \text{ is satisfiable.}$$

Complexity of the reduction.

- Each variable is represented by its index in $O(\log n)$ space.
- Φ_{cell} is a conjunction of $O(n^{2k})$ formulas of fixed length.
- Φ_{start} is a conjunction of $O(n^k)$ variables.
- Φ_{move} is a conjunction of $O(n^{2k})$ formulas of fixed length.
- Φ_{accept} is a variable.

Since every part of Φ can be written down in polynomial time in n, we have $L(N) \leq_m^p$ SAT. □

kSAT

A Boolean formula Φ is in kCNF if Φ is in conjunctive normal form and each clause has k literals (arguments or their negations), e.g. $\Phi = (x_1 \vee x_2' \vee x_3') \wedge (x_2 \vee x_3' \vee x_4)$ is in 3CNF.

$$k\text{SAT} := \{\Phi \text{ in } k\text{CNF} \ : \ \Phi \text{ is satisfiable}\}$$

Corollary

3SAT is NP-complete.

Proof.

Φ in the proof for SAT is a conjunction of Boolean formulas $\varphi(y_1, \ldots, y_\ell)$ for a constant k.

1. Any $\varphi(y_1, \ldots, y_k)$ can be written in kCNF with $\leq 2^k$-clauses.
2. Any formula in kCNF can be written in 3CNF, e.g.
   $y_1 \vee y_2' \vee y_3' \vee y_4$ is satisfiable iff
   $(y_1 \vee y_2' \vee z) \wedge (y_3' \vee y_4 \vee z')$ is satisfiable.

□