

4-11 P and NP

Hengfeng Wei

hfwei@nju.edu.cn

May 20, 2019



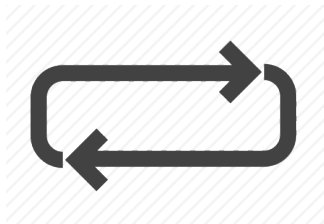


“对于数学问题，自己想出解答，
和判断别人说的解答是否正确，何者比较简单？”

CONCEPT

decide.

ACCEPT



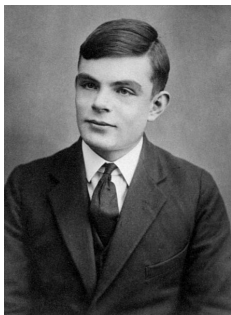
Always terminate.

May loop forever for “NO”
instance.

Definition (Halting Problem)

Input: An arbitrary program and input

Output: Will the program eventually halt?



Alan designed the perfect computer

Undecidable

But Acceptable (Semi-decidable)

$$P = \left\{ L : L \text{ is } \textcolor{red}{\text{decided}} \text{ by a poly. time algorithm} \right\}$$

Theorem (Theorem 34.2)

$$P = \left\{ L : L \text{ is } \textcolor{red}{\text{accepted}} \text{ by a poly. time algorithm} \right\}$$

*You can safely forget “semi-decidable”
in computational complexity theory.*

Definition (NP)

$$L \in \text{NP}$$

$$\iff$$

\exists **poly.** time *verifier* $V(x, c)$ such that

$$\forall x \in \{0, 1\}^* : x \in L \iff \exists c \text{ with } |c| = O(|x|^k), V(x, c) = 1.$$

NP-problems has short **certificates** that are easy to verify.

$\exists L : L \notin \text{NP?}$



Alan designed the perfect computer

$\exists L : L \notin \text{NP} \wedge L \text{ is decidable?}$

Theorem (Deterministic Time Hierarchy Theorem (1965))

$$f(n) \log f(n) = o(g(n)) \implies \text{DTIME}(f(n)) \subsetneq \text{DTIME}(g(n))$$

$$\text{P} \subsetneq \text{EXP}$$

Theorem (Non-deterministic Time Hierarchy Theorem (Cook, 1972))

$$f(n+1) = o(g(n)) \implies \text{NTIME}(f(n)) \subsetneq \text{NTIME}(g(n))$$

$$\text{NP} \subsetneq \text{NEXP}$$

$\exists L : L \notin \text{NP} \wedge L \text{ is decidable?}$

“Equivalence of Regular Expressions with Squaring” is
NEXP-complete:

$$e_1 \cup e_2, \quad e_1 \cdot e_2, \quad e^2$$

Closure of NP (CLRS 34.2-4)

NP is closed under \cup, \cap, \cdot, \star .

$$L_1 \in \text{NP}, L_2 \in \text{NP} \implies L = L_1 \circ L_2 \in \text{NP}$$

$$L_1 \in \text{NP}, L_2 \in \text{NP} \implies L = L_1 \cup L_2 \in \text{NP}$$

```
1: procedure V( $x, c$ )  
2:   if  $c \neq c_1 \# c_2$  then  
3:     return 0  
  
4:   return  $V(x, c_1) \vee V(x, c_2)$ 
```

$$x \in L_1 \cup L_2 \iff \exists c, V(x, c) = 1$$

$$L_1 \in \text{NP}, L_2 \in \text{NP} \implies L = L_1 \cap L_2 \in \text{NP}$$

```
1: procedure V( $x, c$ )  
2:   if  $c \neq c_1 \# c_2$  then  
3:     return 0  
  
4:   return  $V(x, c_1) \wedge V(x, c_2)$ 
```

$$x \in L_1 \cap L_2 \iff \exists c, V(x, c) = 1$$

$$L_1 \in \text{NP}, L_2 \in \text{NP} \implies L = L_1 \cdot L_2 \in \text{NP}$$

```

1: procedure V( $x, c$ )
2:   if  $c \neq c_1 \# c_2 \& m$  then
3:     return 0

4:   return  $V(x_{1\dots m}, c_1) \wedge V(x_{m+1\dots|x|}, c_2)$ 

```

$$x \in L_1 \cdot L_2 \iff \exists c, V(x, c) = 1$$

$$L \in \text{NP} \implies L^* \in \text{NP}$$

```

1: procedure V( $x, c$ )
2:   for  $k \leftarrow 1$  to  $|x|$  do
3:      $m_0 \leftarrow 0, m_k \leftarrow |x|$ 
4:     if  $c = c_1 \# c_2 \# \cdots \# c_k \& m_1 \& m_2 \& \cdots \& m_{k-1}$  then
5:       return  $\bigwedge_{i=1}^{i=k} V(x_{m_{i-1}+1 \dots m_i}, c_i)$ 

```

$$x \in L^* \iff \exists c, A(x, c) = 1$$

Definition (Polynomial-time Reduction)

$L_1 \leq_p L_2$ if \exists **poly.** time function f such that

$$\forall x : x \in L_1 \iff f(x) \in L_2.$$

$\forall L \in \text{NP}, L \leq_p L' \implies L'$ is NP-hard

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

$$\text{UNSAT} = \{\varphi : \varphi \text{ is unsatisfiable.}\}$$

Q : Is UNSAT NP-hard?

Proof.

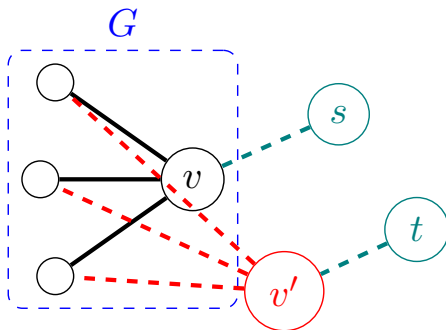
$$\text{SAT} \leq_p \text{UNSAT}$$

$$x \in \text{SAT} \iff x \notin \text{UNSAT}$$



$$\forall x : x \in L_1 \iff f(x) \in L_2$$

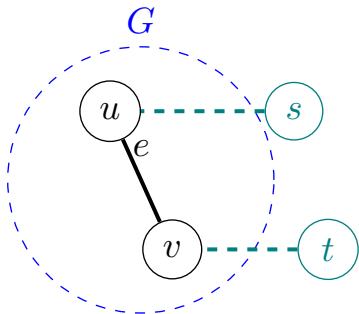
HAM-PATH is NP-complete

 $\text{HAM-CYCLE} \leq_p \text{HAM-PATH}$ 

$$G \in \text{HAM-CYCLE} \iff G' \in \text{HAM-PATH}$$

HAM-CYCLE \leq_p HAM-PATH

$\forall e \in G : \text{Construct } G_e$



Definition (Polynomial-time Reduction)

$L_1 \leq_p L_2$ if \exists **poly.** time function f such that

$$\forall x : x \in L_1 \iff f(x) \in L_2.$$

x for $L_1 \mapsto x' = f(x)$ for L_2

Call the oracle O_2 for L_2 once

Answer whatever O_2 returns

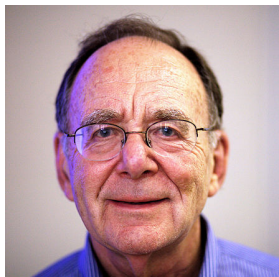


Definition (Polynomial-time Reduction)

$L_1 \leq_p L_2$ if \exists **poly.** time function f such that

$$\forall x : x \in L_1 \iff f(x) \in L_2.$$

Karp Reduction



REDUCIBILITY AMONG COMBINATORIAL PROBLEMS

Richard M. Karp

University of California at Berkeley

(1972)

Richard M. Karp (1935 ~)

Cook Reduction



The Complexity of Theorem-Proving Procedures

Stephen A. Cook

University of Toronto

(1971)

Stephen Cook (1939 ~)





Office 302

Mailbox: H016

hfwei@nju.edu.cn