

# Universal Algebra and Computational Complexity

## Lecture 2

Ross Willard

University of Waterloo, Canada

Třešť, September 2008

# Summary of Lecture 1

Recall from yesterday:

$$\begin{array}{ccccccc} L & \subseteq & P & \subseteq & PSPACE & \subseteq & EXPTIME \\ \Psi & & \Psi & & \Psi & & \Psi \\ & & PATH & & 3COL & & CLO \\ FVAL & & & & & & \end{array}$$

Topics for today:

- “Nondeterministic” complexity classes
- Reductions
- Complete problems

# “Nondeterministic polynomial time”: an example

Recall

Graph 3-Colorability problem ( $3COL$ )

INPUT: a finite graph  $G = (V, E)$ .

QUESTION: Does  $G$  have a 3-coloring?

Recall that we only know  $3COL \in EXPTIME$  (and  $PSPACE$ ).

Most complexity theorists conjecture that  $3COL$  is not tractable.

HOWEVER, if we are GIVEN a 3-coloring of  $G$ , it is easy (tractable) to VERIFY the correctness of the 3-coloring (and thus know that  $G$  is 3-colorable).

Informally,  $3COL$  is a **projection of a problem in  $P$** .

## 3COL as a projection of a problem in $P$

Identify 3COL with set

$$\{G : 3COL \text{ answers "YES" on input } G\}.$$

Similarly with other decision problems.

Define

$$3COL-TEST = \{(G, \chi) : \chi \text{ is a 3-coloring of } G\}.$$

Clearly 3COL-TEST is tractable (in  $TIME(N^2)$ , hence in  $P$ ).

And

$$G \in 3COL \iff \exists \chi [(G, \chi) \in 3COL-TEST].$$

If  $(G, \chi) \in 3COL-TEST$ , then we call  $\chi$  a **certificate** for “ $G \in 3COL$ .”

We say that:

- $3COL-TEST$  is a **polynomial-time certifier** for  $3COL$ .
- $3COL$  is **polynomial-time certifiable**.
- $3COL$  is in **Nondeterministic Polynomial Time** (or  $NP$ ).

# Nondeterministic Polynomial Time ( $NP$ )

More precisely,

## Definition

A decision problem  $D$  is *Polynomial-time certifiable* if there exists a decision problem  $E \in P$  such that

- $x \in D \Leftrightarrow \exists w[(x, w) \in E]$ .
- Technicality:  $\exists$  polynomial  $p(N)$  s.t.  $(x, w) \in E \Rightarrow |w| \leq p(|x|)$ .

## Definition

$NP$  is the class of polynomial-time certifiable problems.

$$L \subseteq P \subseteq NP \subseteq PSPACE \subseteq EXPTIME$$

$$\cup \\ 3COL$$

# More examples of $NP$ problems

The following problems are all in  $NP$  (and not known to be in  $P$ ).

①  $4COL$ ,  $5COL$ , etc.

②  $SAT$ :

- INPUT: a boolean formula  $\varphi$ .
- QUESTION: is  $\varphi$  satisfiable?
- Certificate: an assignment of values to the variables making  $\varphi$  true.
- Polynomial-time certifier: given  $(\varphi, \mathbf{c})$ , decide if  $\varphi(\mathbf{c}) = 1$  (i.e.,  $FVAL$ ).

③  $ISO$ :

- INPUT: two finite graphs  $G_1, G_2$ .
- QUESTION: are  $G_1$  and  $G_2$  isomorphic?
- Certificate: an isomorphism from  $G_1$  to  $G_2$ .
- Polynomial-time certifier: given  $(G_1, G_2, f)$ , decide if  $f : G_1 \cong G_2$ .

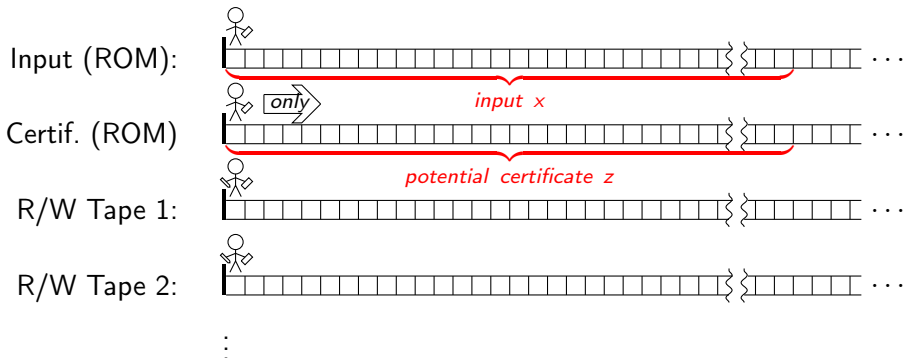
④  $HAMPATH$ :

- INPUT: a finite directed graph  $G$ .
- QUESTION: does  $G$  have a *Hamiltonian path*?

# Certifying Turing machines

In a similar way, we can “stick an  $N$ ” in front of any complexity class.  
To define it precisely, we need the notion of a **certifying Turing machine**:

- One additional input tape; holds the potential certificate.
  - Read-only
  - Grad student reader can only move RIGHT.





# Nondeterministic complexity classes

Roughly,

## Definition

If  $\square$  is a complexity class, then a decision problem  $D$  is in  $N\square$  iff there exists a decision problem  $E$  in two inputs  $(x, z)$ , and there exists a certifying Turing machine  $M$ , such that

- $x \in D \Leftrightarrow \exists w[(x, w) \in E]$ .
  - $M$  decides  $E$ .
  - Moreover,  $\forall (x, z)$ ,  $M$  decides whether  $(x, z) \in E$  with resource usage as defined by  $\square$ , measured as a function of  $N = \text{the length of } x$ .
- 
- Exercise: this defines  $NP$  equivalently.
  - $NL$  = “Nondeterministic  $LOGSPACE$ ”
  - $NSPACE$  = “Nondeterministic  $PSPACE$ ”
  - $NEXPTIME$  = “Nondeterministic  $EXPTIME$ ”

## Theorem

*PATH* is in NL.

Proof. We show that *PATH* is a projection of a problem that can be decided by a *LOGSPACE* certifying Turing machine.

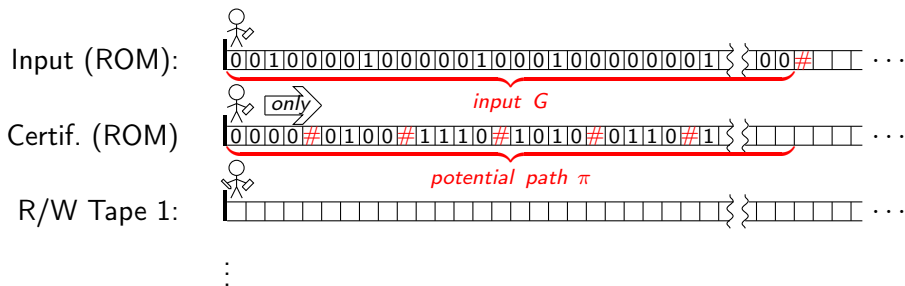
Define

$$\begin{aligned} \textit{PATH-TEST} = \{ (G, \pi) : G \text{ is a directed graph with } V = \{0, \dots, n-1\}, \\ \text{and } \pi = (v_0, v_1, \dots, v_k) \text{ is a path from } 0 \text{ to } 1 \text{ in } G \} \end{aligned}$$

Clearly *PATH* is a projection of *PATH-TEST*.

# Certifying *PATH-TEST*

We can build a certifying Turing machine which solves *PATH-TEST* ...



While the certifying student traverses  $\pi$ , the R/W Tape 1 student copies and remembers the last two vertices traversed, and checks the input tape to see if they form an edge.

Only LOGSPACE (as a function of the length of the input  $G$ ) is needed.

# Comparing deterministic and nondeterministic classes

Let  $f : \mathbb{N} \rightarrow \mathbb{N}$  be “nice” and such that  $f(N) \geq \log N$ .

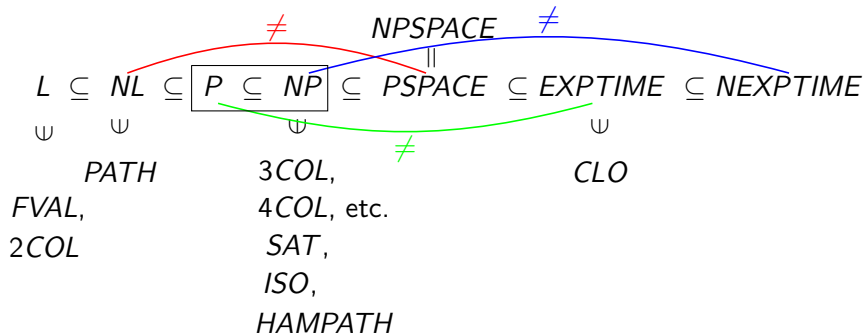
## Theorem

- 1  $TIME(f(N)) \subseteq NTIME(f(N))$  and similarly for  $SPACE$ .
- 2  $NTIME(f(N)) \subseteq SPACE(f(N))$ .
- 3  $NSPACE(f(N)) \subseteq TIME(2^{O(f(N))})$ .
- 4 (Savitch's Theorem):  $NSPACE(f(N)) \subseteq SPACE(f(N)^2)$ .

Since  $PATH \in NL$ , Savitch's theorem shows  $PATH \in SPACE((\log N)^2)$ .

(Our algorithm showed only that  $PATH \in SPACE(\sqrt{N})$ .)

# Summary of complexity classes



$10^6$  USD prize (Clay Mathematics Institute) for answering  $P \stackrel{?}{=} NP$ .

# Reductions

Suppose  $C, D$  are decision problems.

Suppose  $f : C_{inp} \rightarrow D_{inp}$  is a function.

We say that

$f$  reduces  $C$  to  $D$ ,

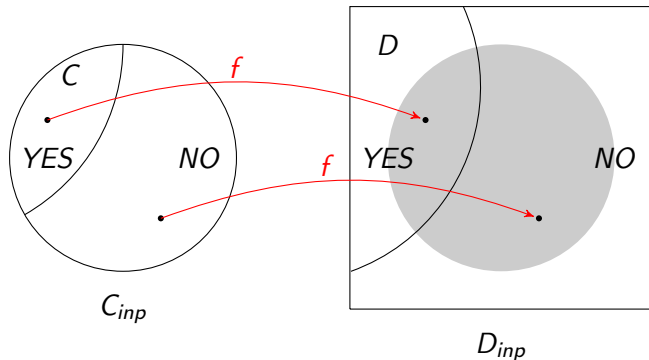
and write

$$C \leq_f D,$$

if for all  $x \in C_{inp}$ ,

$$x \in C \Leftrightarrow f(x) \in D.$$

# Picture of $C \leq_f D$



Intuition: if  $C \leq_f D$ , then

- Algorithms for  $D$  and  $f$  can be used to solve  $C$ .
- Hence  $D$  is **at least as hard** as  $C$  (modulo the cost of computing  $f$ ).

# Example

Recall the problems *3COL* and *SAT*:

## *3COL*

INPUT: a finite graph  $G = (V, E)$ .

QUESTION: is  $G$  3-colorable?

## *SAT*

INPUT: a boolean formula  $\varphi$ .

QUESTION: is  $\varphi$  satisfiable?

Let's find a function  $f$  which reduces *3COL* to *SAT*.



# A reduction of 3COL to SAT

Given a finite graph  $G = (V, E)$ , we want a boolean formula  $\varphi_G$  such that

$G$  is 3-colorable  $\Leftrightarrow \varphi_G$  is satisfiable.

- The variables of  $\varphi_G$  will be all  $x_v^c$  ( $v \in V$ ,  $c \in \{r, g, b\}$ ).
  - Think of  $x_v^c$  as representing the assertion “ $v$  is colored  $c$ .”
- For each  $v \in V$  let  $\alpha_v$  be the formula “ $v$  has exactly one color,” i.e.,

$$(x_v^r \vee x_v^g \vee x_v^b) \wedge \neg(x_v^r \wedge x_v^g) \wedge \neg(x_v^r \wedge x_v^b) \wedge \neg(x_v^g \wedge x_v^b).$$

- For  $v, w \in V$  let  $\beta_{v,w}$  be the formula “ $v$  and  $w$  have different colors,” i.e.,

$$\neg(x_v^r \wedge x_w^r) \wedge \neg(x_v^g \wedge x_w^g) \wedge \neg(x_v^b \wedge x_w^b).$$

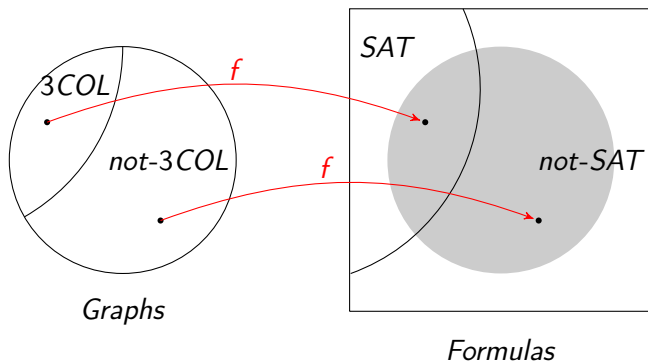
- Let

$$\varphi_G = \left( \bigwedge_{v \in V} \alpha_v \right) \wedge \left( \bigwedge_{(v,w) \in E} \beta_{v,w} \right).$$

This clearly works.

# Picture of $3COL \leq_f SAT$

Define  $f : G \mapsto \varphi_G$ . Then  $3COL \leq_f SAT$ .

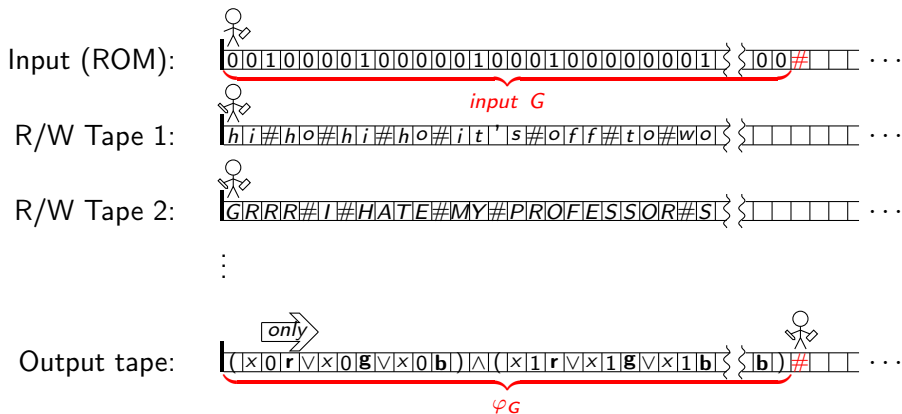


Thus *SAT* is **at least as hard as** *3COL*, modulo the cost of computing  $\varphi_G$ .

What is the cost of computing  $\varphi_G$ ?

# Computing $f$ with a functional Turing machine

Idea: replace the output bit with an output write-only tape.



At the end.

Exercise: Can compute  $\varphi_G$  from  $G$  in  $TIME(N^2)$  and  $SPACE(\log N)$ .

# Complexity of computing $f$

In general:

## Definition

a **functional Turing machine** is a Turing machine whose output *bit* is replaced by an output *tape* (write-only).

- Output tape grad student can only move RIGHT.

Let  $C, D$  be decision problems with appropriately encoded input sets  $C_{inp}, D_{inp}$  respectively.

## Definition

A function  $f : C_{inp} \rightarrow D_{inp}$  is *computed by a functional Turing Machine*  $M$  if whenever  $M$  is started with input  $x \in C_{inp}$ , it eventually halts with  $f(x)$  written on its output tape.

# $X$ -computable functions

Let  $X$  be a complexity class (such as  $P$ ,  $L$ , etc.).

## Definition

We say that a function  $f : C_{inp} \rightarrow D_{inp}$  is *computable in  $X$*  if there exists a functional Turing Machine which computes  $f$  and on input  $x$  requires no more resources than those permitted by the definition of  $X$ .

Example: the function  $f : G \mapsto \varphi_G$  in our example showing  $3COL \leq_f SAT$  is  $P$ -computable.

- (In fact, it is  $L$ -computable.)

## Lemma

For any decent complexity class  $X$ , if  $C \leq_f D \in X$  and  $f$  is  $X$ -computable, then  $C \in X$ .

# $X$ -reductions

Suppose  $X, Y$  are complexity classes with  $X \subseteq Y$ .  
Let  $C, D$  be decision problems with  $C, D \in Y$ .

## Definition

- ① We say that  $C$  reduces to  $D$  (mod  $X$ ) and write

$$C \leq_X D$$

if there exists an  $X$ -computable function  $f : C_{inp} \rightarrow D_{inp}$  which reduces  $C$  to  $D$ .

- ② We write  $C \equiv_X D$  if both  $C \leq_X D$  and  $D \leq_X C$ .

This turns the  $\equiv_X$ -classes of  $Y$  into a poset.

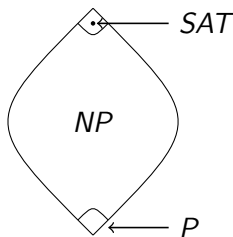
Most widely used when  $X = P$ .

# The picture of $NP \pmod{P}$

## Theorem

*The poset  $(NP / \equiv_P, \leq_P)$  has ...*

- 1 *a least element (consisting of all the elements of  $P$ ), and*
- 2 *(S. Cook, '71; L. Levin, '73) a greatest element, namely, the  $\equiv_P$ -class containing SAT.*



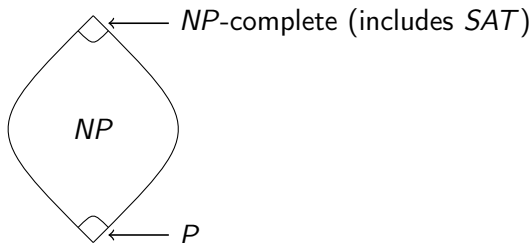
Jargon: SAT is **NP-complete** (for  $\leq_P$  reductions).

## Definition

A decision problem  $D$  is  **$NP$ -complete** if:

- $D \in NP$ , and
- $C \leq_P D$  for all  $C \in NP$ .

Equivalently (by Cook-Levin),  $D$  is  $NP$ -complete iff  $D \equiv_P SAT$ .





## Theorem (R. Karp, '72)

*Many problems are NP-complete.*

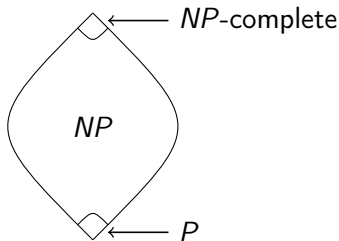
Examples:

- $3COL$ ,  $4COL$ , etc.
- $HAMPATH$
- $3SAT$  (the restriction of  $SAT$  to formulas in CNF, each conjunct being a disjunction of at most 3 literals)

(Exercise: check that our proof we gave for  $3COL \leq_P SAT$  also shows  $3COL \leq_P 3SAT$ .)

# Ladner's Theorem

Remark: the picture below of  $NP$  is accurate only if  $P \neq NP$ :



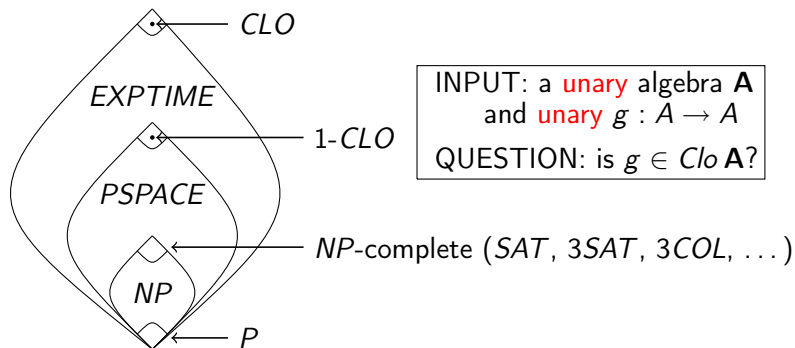
The picture if  $P = NP$ :  $\diamond P = NP = NP$ -complete

Theorem (R. Ladner, '75)

If  $P \neq NP$ , then  $|NP / \equiv_P| \geq 3$ .

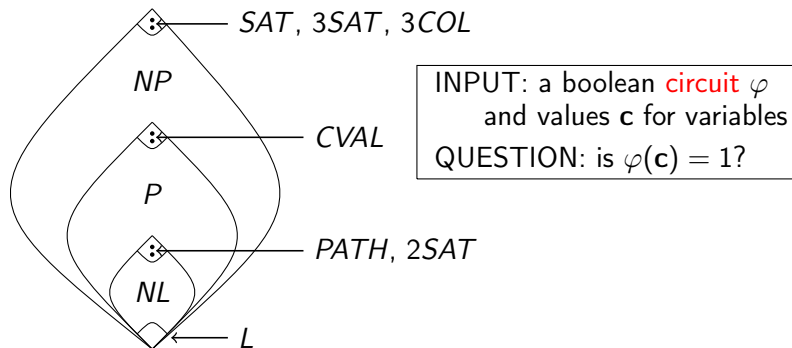
In fact, if  $P \neq NP$ , then  $NP / \equiv_P$  is order dense.

# The picture of $EXPTIME \pmod P$



- (H. Friedman '82, unpubl.; C. Bergman, D. Juedes & G. Slutzki, '99)  $CLO$  is  **$EXPTIME$ -complete** (for  $\leq_P$  reductions).
- (D. Kozen, '77)  $1-CLO$  is  **$PSPACE$ -complete** (for  $\leq_P$  reductions).

# The picture of $NP \pmod L$



INPUT: a boolean **circuit**  $\varphi$   
and values  $\mathbf{c}$  for variables  
QUESTION: is  $\varphi(\mathbf{c}) = 1$ ?

- $SAT$ ,  $3SAT$  and  $3COL$  are  $NP$ -complete (for  $\leq_L$  reductions).
- (W. Savitch, '70)  $PATH$ ,  $2SAT$  are  **$NL$ -complete** (for  $\leq_L$  reductions).
- (R. Ladner, '75)  $CVAL$  is ...  **$P$ -complete** (for  $\leq_L$  reductions).
- (???)  $HORN-SAT$  and  $HORN-3SAT$  are also  **$P$ -complete**.

# Summary

$L$	$\subseteq$	$NL$	$\subseteq$	$P$	$\subseteq$	$NP$	$\subseteq$	$PSPACE$	$\subseteq$	$EXPTIME \dots$
$\Psi$		$\Psi$		$\Psi$		$\Psi$		$\Psi$		$\Psi$
$FVAL,$ $2COL$		$PATH,$ $2SAT$		$CVAL,$ $HORN-$ $3SAT$		$SAT,$ $3SAT,$ $3COL,$ $4COL,$ etc. $HAMPATH$		$1-CLO$		$CLO$

Moreover, each problem listed above is “hardest in its class,” i.e., is complete with respect to either  $\leq_P$  or  $\leq_L$  reductions.

In Thursday’s lecture: some problems from universal algebra.