Universal Algebra and Computational Complexity Lecture 2

Ross Willard

University of Waterloo, Canada

Třešt', September 2008

Summary of Lecture 1

Recall from yesterday:

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

Topics for today:

- "Nondeterministic" complexity classes
- Reductions
- Complete problems

Certificates for $3COL$

Identify $3COL$ with $\{G : G \text{ is 3-colorable}\}$. Similarly with other decision problems.

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

Define

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

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

And

$$G \in 3COL \quad\Leftrightarrow\quad \exists\chi[(G,\chi) \in 3COL\text{-}TEST].$$

Certificates for $3COL$

If $(G, \chi) \in 3COL\text{-}TEST$, then we call χ a certificate for "$G \in 3COL$."

We say that:

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

NP

More generally,

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: there exists a polynomial bound to the length of w as a function of the length of x.

NP is the class of polynomial-time certifiable problems.

$$\begin{array}{ccccccccc} L & \subseteq & P & \subseteq & NP & \subseteq & PSPACE & \subseteq & EXPTIME \\ & & & & \cup\!\!\!\!| & & & & \\ & & & & 3COL & & & & \end{array}$$

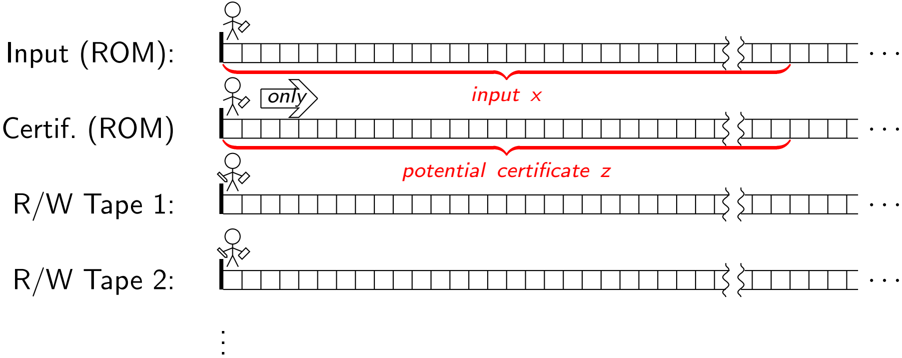
More examples of NP problems

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

1. $4COL$, $5COL$, etc.
2. SAT:
   - INPUT: a boolean formula φ.
   - QUESTION: is φ satisfiable?
   - Certificate: an assignment of values to the variables making φ 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 *Hamiltonion path*?

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,

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 $N =$ the length of x.

- $NL =$ "Nondeterministic $LOGSPACE$"
- $NSPACE =$ "Nondeterministic $PSPACE$"
- $NEXPTIME =$ "Nondeterministic $EXPTIME$"

Example

PATH is in NL.

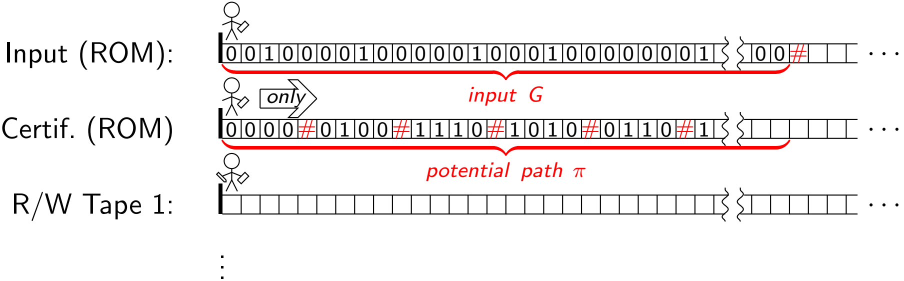
Proof. We show that *PATH* is a projection of a problem that can be decided by a *LOGSPACE* certifying Turing machine.

Define

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

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

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



. . . and using only LOGSPACE as a function of the length of G.

Summary of complexity classes

$$\begin{array}{ccccccccc} & & & & & & NPSPACE & & \\ & & & & & & \| & & \\ L & \subseteq & NL & \subseteq & \boxed{P \subseteq NP} & \subseteq & PSPACE & \subseteq EXPTIME & \subseteq NEXPTIME \\ \Cup & & \Cup & & \Cup & & & & \\ FVAL, & & PATH & & 3COL, & & & & \\ 2COL & & & & 4COL, \text{ etc.} & & & & \\ & & & & SAT, & & & & \\ & & & & ISO, & & & & \\ & & & & HAMPATH & & & & \end{array}$$

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

Reductions

Suppose C, D are decision problems.

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

We say that

$$f \text{ reduces } C \text{ to } D,$$

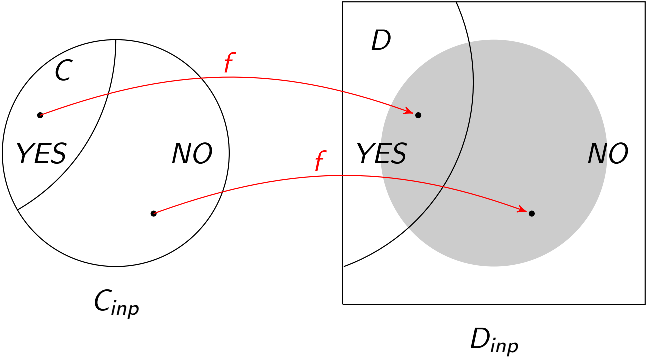
and write

$$C \leq_f D,$$

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 φ.
QUESTION: is φ 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 φ_G such that

$$G \text{ is 3-colorable} \iff \varphi_G \text{ is satisfiable.}$$

- The variables of φ_G are $x_v^{\mathbf{c}}$ ($v \in V$, $\mathbf{c} \in \{\mathbf{r}, \mathbf{g}, \mathbf{b}\}$).
  - Think of $x_v^{\mathbf{c}}$ as representing the assertion "v is colored $\mathbf{c}$."
- For each $v \in V$ let α_v be the formula "v has exactly one color," i.e.,

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

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

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

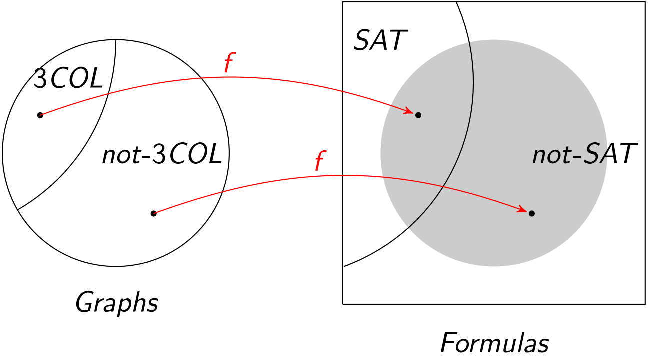
- Let

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

This clearly works.

Picture of $3COL \leq_f SAT$

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

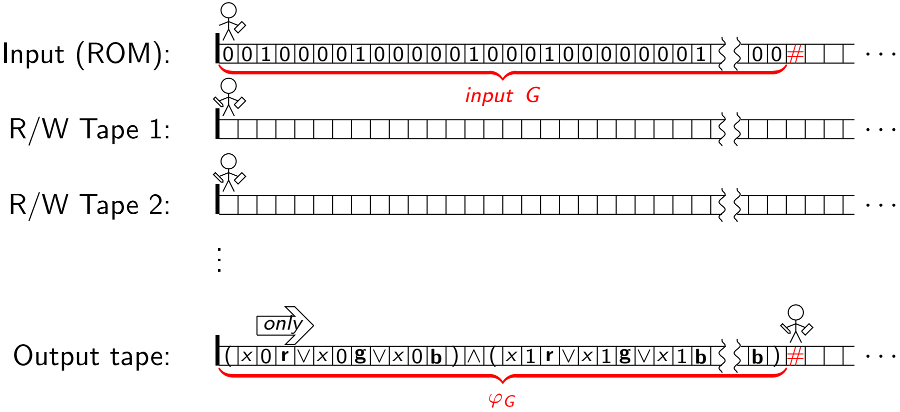


SAT is at least as hard as $3COL$, modulo the cost of computing φ_G.

What is the cost of computing φ_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 φ_G from G in $TIME(N^2)$ and $SPACE(\log N)$.

Picture of a functional Turing machine

In general:

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.

A function $f : C_{inp} \to D_{inp}$ 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.).

We say that a function $f : C_{inp} \to D_{inp}$ is *computable in X* if there exists a 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 $C, D \in Y$.

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

$$C \leq_X D$$

   if there exists an X-computable function $f : C_{inp} \to D_{inp}$ which reduces C to D.
2. We write $C \equiv_X D$ if both $C \leq_X D$ and $D \leq_X C$.

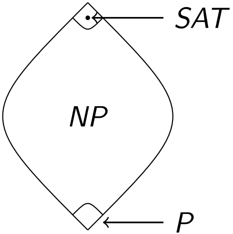
This turns the $\equiv_X$-classes of Y into a poset.

Most widely used when $X = P$.

The picture of NP (mod P)

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*.*

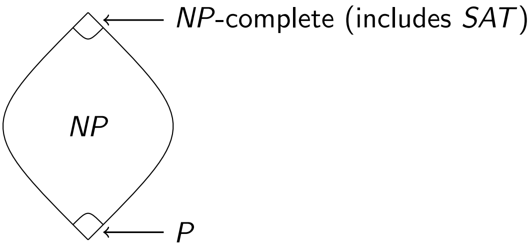


Jargon: SAT is NP-complete (for $\leq_P$ reductions).

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 $D \equiv_P SAT$.



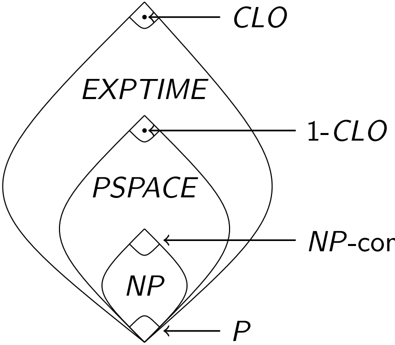
Many problems in NP are known to be NP-complete.

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 $3COL \leq_P SAT$ is actually a proof of $3COL \leq_P 3SAT$.)

The picture of *EXPTIME* (mod *P*)

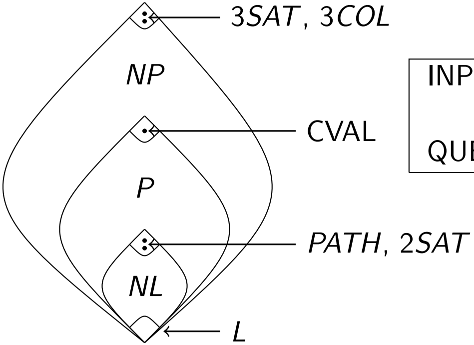


INPUT: a unary algebra $\mathbf{A}$
and unary $g : A \to A$

QUESTION: is $g \in Clo\,\mathbf{A}$?

- (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 (mod L)



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

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

Summary

$$
\begin{array}{ccccccccccccc}
L & \subseteq & NL & \subseteq & P & \subseteq & NP & \subseteq & PSPACE & \subseteq & EXPTIME & \cdots \\
\rotatebox{90}{$\in$} & & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\in$} & \\
FVAL, & & PATH, & & CVAL & & SAT, & & 1\text{-}CLO & & CLO & \\
2COL & & 2SAT & & & & 3SAT, & & & & & \\
 & & & & & & 3COL, & & & & & \\
 & & & & & & 4COL, \text{ etc.} & & & & & \\
 & & & & & & HAMPATH & & & & &
\end{array}
$$

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