Lecture 19

CS 170

- Search problems
- P and NP
- Reductions

Sanjam Garg.

Various search problems

- Shortest path
- minimum spanning trees
- maximum increasing subsequence.
- max flows.

⋮

Can we always find efficient algorithms for any optimization task?

Satisfiability or SAT

Input: a formula ϕ over variables $x_1 \dots x_n$

Solution: find a satisfying assignment – an assignment (T/F) to $x_1 \dots x_n$ such that ϕ is satisfied.
or output no satisfying assignment exists.

Example:

$$\phi = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \bar{x}_2) \wedge (x_2 \vee \bar{x}_3) \wedge (x_3 \vee \bar{x}_1) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3)$$

↳ try all possible assignments $\longrightarrow 2^n$

$$\text{VERIFY}\Big(\overset{\text{Instance}}{\phi},\ \overset{\text{Solution}}{f:\{x_1 \dots x_n\} \to \{T, F\}}\Big) \to ① \ \alpha = \phi\,(f(x_1) \dots f(x_n))$$

② output α.

Search problems

A **search problem** is specified by an algorithm VERIFY that takes two inputs, an instance I and a proposed solution S and runs in time polynomial in $|I|$.

We say that S is a solution to I if $\text{VERIFY}(I, S) = 1$

- HornSAT
- 2-SAT

P : Class of search problems where we can **find** a solution in polynomial in $|I|$ time.

Horn SAT $\in$ P & 2SAT $\in$ P

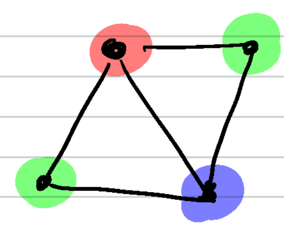
NP : Class of all search problems; i.e. the class of all problems where we can **verify** a solution in polynomial time

SAT $\in$ NP

Horn SAT $\in$ NP 2SAT $\in$ NP

$P \subseteq NP$

Graph 3-coloring



Input : $G = (V, E)$

Solution : Coloring function $c: V \to \{R, G, B\}$ so that $\forall (u,v) \in E \quad c(u) \neq c(v)$

or output no coloring function exists

Lemma: Graph three coloring $\in$ NP.

Proof: VERIFY(Instance $G=(V,E)$, Solution $c: V \to \{R,G,B\}$)

① output 1 if $\forall (u,v) \in E \quad c(u) \neq c(v)$ & $\forall v \in V \quad c(v) \in \{R,G,B\}$

② else output 0.

Vertex Cover (VC)

Input: $G = (V, E)$, b

Solution: $A \subseteq V$ s.t. $|A| \leq b$ s.t. $\forall (u,v) \in E$ either $u \in A$ or $v \in A$

or output no solution exists.

Lemma: $VC \in NP$

Proof: VERIFY (Instance $G=(V,E), b$, Solution Set A)

① Output 0 if $|A| > b$

② output 0 if $\exists (u,v) \in E$ s.t. $u \notin A$ & $v \notin A$

③ output 1 otherwise.

Factoring

Input: $N = pq$ (where p & q are large primes)

Solution: p, q

Lemma: Factoring $\in NP$

Proof: VERIFY (Instance N , Solution p, q)

output 1 if $N = pq$

& 0 otherwise.

Travelling Salesman Problem (TSP)

Instance: n vertices and distances d_{ij} between them & a bound B

Solution: a permutation $\tau : \{1 \dots n\} \to \{1 .. n\}$ s.t.

$$d_{\tau(1)\tau(2)} + d_{\tau(2)\tau(3)} \dots d_{\tau(n)\tau(1)} \leq B$$

or output that no solution exists.

Lemma: $TSP \in NP$

Proof: $\text{VERIFY} \Big(\underset{n, \{d_{ij}\}, B}{\text{Instance}} , \underset{\tau : \{1..n\} \to \{1...n\}}{\text{Solution}} \Big)$

output 1 if (1) $d_{\tau(1)\tau(2)} + d_{\tau(2)\tau(3)} \dots d_{\tau(n)\tau(1)} \leq B$ $\wedge$ (2) $\forall i, j \in \{1 ... n\}$ $\tau(i) \neq \tau(j)$

and 0 otherwise.

Rudrata Cycle / Hamiltonian Cycle.

Instance: $G = (V, E)$ $\quad |V| = n$

Solution: permutation $\tau : \{1 \dots n\} \to V$

s.t. $(\tau(1), \tau(2)), \dots (\tau(n), \tau(1)) \in E$

or that no solution exists

Lemma: $RC/HC \in NP$

Proof: $\text{VERIFY} \Big(\underset{G = (V, E)}{\text{Instance}} , \underset{\tau : \{1..n\} \to V}{\text{Solution}} \Big)$

output 1 if (i) $\forall i, j$ $\tau(i) \neq \tau(j)$

(ii) $(\tau(1), \tau(2)) \dots (\tau(n), \tau(1)) \in E$

and 0 otherwise.

Reductions

$A \longrightarrow_p B$

A reduces to B in polynomial time.

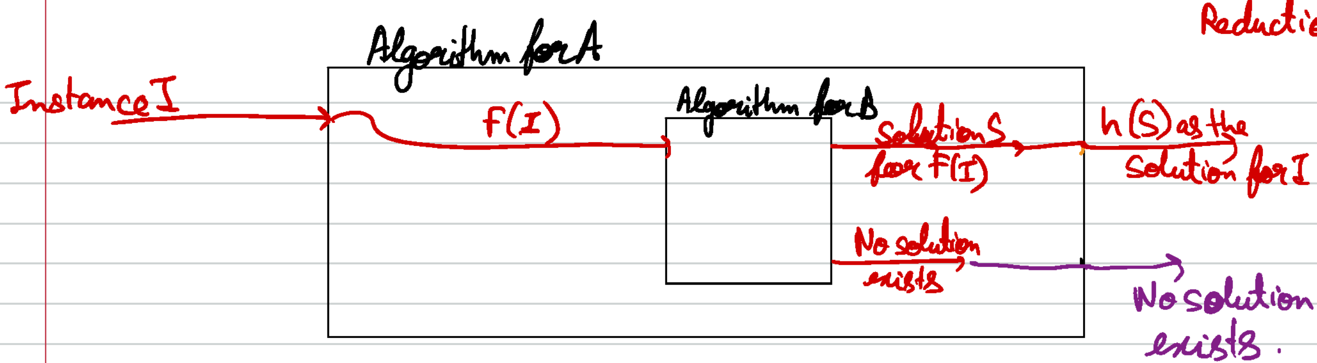
an algorithm for B yields an algorithm for A

B is at least as hard as A.

$A \leq_p B$

Reduction = (f, h)

two poly time procedures



1) Running time: f & h run in time polynomial in $|I|$

2) If B outputs S as a solution to $f(I)$ (under problem B)

then $h(S)$ is a solution to I (under problem A)

3) If B outputs "no solution exists" then no solution to I exists either.

↳ if I has a solution then $f(I)$ also has a solution.

Rudrata Cycle $\longrightarrow$ Rudrata Half Cycle

Rudrata Cycle

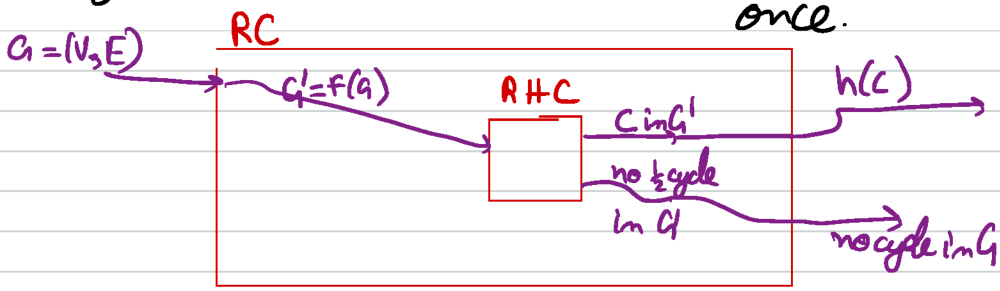
Input: $G = (V, E)$

Solution: cycle visiting each vertex exactly once

Rudrata Half Cycle

Input: $G' = (V', E')$

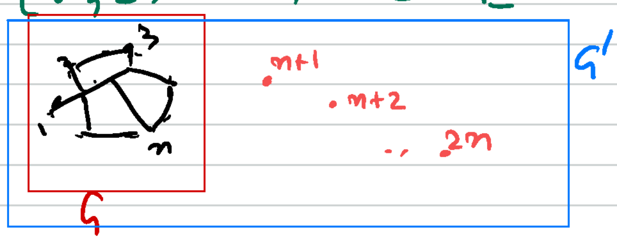
Solution: cycle visiting $|V|/2$ of the vertices exactly once.



$F(G)$:

$|V| = n$

$G' = (V', E') \longrightarrow E' = E \qquad V' = V \cup \{n+1 \dots 2n\}$



Lemma: f & h are polynomial time in $|I|$

Proof: $\square$

Lemma: If C is a RHC in G' then $\underbrace{h(C)}_{=C}$ is also RC in G.

Proof: (i) C does not contain $n+1 \dots\dots 2n$

(ii) Number of vertices in C is $n = |V|$ $\quad (|V'| = 2|V|)$

$\Downarrow$

C contains all of the vertices $1 \dots n$

$\Downarrow$

C is a RC in G

Lemma: If G has a RC then G' has a HRC.

Proof: Let C be the HC in G then (C has $|V|$ vertices)

C is also the HRC in G'.

$\downarrow$ $2|V|$ vertices.

SAT

Input: Formula ϕ

Solution: Assignment $S : \{x_1 \dots x_n\} \to \{T, F\}$

3-SAT

Input: Formula ϕ — each clause has 3 variables.

Solution: Assignment $S : \{x_1 \dots x_n\} \to \{T, F\}$

$w = f(\phi)$

$\bar{x}_4$

$(a_1 \vee a_2 \dots\dots a_k)$

$y_1 \dots y_{k-3}$

$$(a_1 \vee a_2 \vee y_1) \wedge (\bar{y}_1 \vee a_3 \vee y_2) \dots (\bar{y}_{k-3} \vee a_{k-1} \vee a_k)$$

f:

$h(S)$: recover a solution to ϕ

S: $a_1 \to$, $a_2 \to$, $\vdots$, $a_k \to$ $\quad$ $\cancel{y_1 \to \;\vdots\; y_{k-3} \to}$ $\quad \xrightarrow{h} \quad$ $a_1 \to$, $\vdots$, $a_k \to$

Lemma: if $w = f(\phi)$ has a satisfying assignment S then $h(S)$ is a satisfying assignment for ϕ.

Proof: $\exists\, i$ s.t. $a_i = T$

Lemma: if ϕ has a satisfying assignment then w also has a satisfying assignment.

Proof: say $a_i = T$

$y_1 \cdots y_{i-2}$ to be True and rest to false.

$\Downarrow$

w be satisfied.