CS 221 LECTURE 10

2/22/2018

TODAY:

- SAT $\leq_{RP}$ Unique-SAT
- #P, Counting Problems
- Permanent.

Unique-SAT: Inspired by one-way permutations

$$f: \{0,1\}^n \to \{0,1\}^n \qquad 1\text{-}1$$

- "One way: given x computing $f(x)$ is in P.
  given y computing x s.t. $f(x)=y$ is (hopefully) hard"
- $NP = P \Rightarrow$ Oneway permutations do not exist.
  $\Leftarrow$ open.

Hurdle to proving $\Leftarrow$.

Natural Candidate: Given y produce ϕ s.t. $\phi(x)=1$ if $f(x)=y$.

But such a ϕ has unique sat. assignment!

Most NP-completeness reductions will produce ϕ s.t. ϕ has exp( ) many sat. assgmts.

So challenge: Show even ϕ's ~~...~~ with unique sat. assgmt hard to satisfy.

Definitional Challenge: How to define Unique SAT.

Def 1 UniqueSAT = $\{ \phi \mid \phi$ has exactly one sat. assgmt $\}$

Not right! UniqueSAT $\notin$ NP [at least not known]

Correct Defn: Promise Problem

$USAT_Y = \{ \phi \mid \phi$ has exactly 1 sat. assignment $\}$

$USAT_N = \{ \phi \mid \phi$ has no sat. assgmt $\}$.

Now $USAT \leq SAT$!

Can we show converse : SAT $\leq$ USAT

- No known deterministic algorithm !
- (Do not trust this statement ... its a vague recollection)
  " if SAT $\leq_k$ USAT (with Karp reduction) then hierarchy collapses ? "
- [Valiant- Vazirani] SAT $\leq_{RP}$ U-SAT

<u>Randomized Reductions</u> ~~Pr~~ from problem $A = (A_Y, A_N)$ to $B = (B_Y, B_N)$

$A \leq_{BP} B$ if $\exists$ randomized alg f in prob. poly time

s.t. $x \in A_Y \implies \Pr[f(x) \in B_Y] \geq c$

$x \in A_N \implies \Pr[f(x) \notin B_N] \leq s$

[with $c - s \geq \frac{1}{poly(n)}$]

if $s = 0$ then $A \leq_{RP} B$.

Proposition:
- $A \leq_{BP} B$ & $B \in BPP \Rightarrow A \in BPP$
- $A \leq_{RP} B$ & $B \in RP \Rightarrow A \in RP$
- $A \leq_{RP} B$ & $B \in coRP \Rightarrow$?

Theorem [VV]: $SAT \leq_{RP} USAT$.

Proof Idea:
- Will start with formula ϕ & tack on ψ s.t. that # sat assignments reduce
- Pick ψ at random so that $\psi(x)$ satisfies ψ w.p. 2^{-k}. (ind for every

Clealy

$\phi \notin SAT \Rightarrow \phi \wedge \psi \notin SAT$

ϕ has $\sim 2^k$ SAT assgmts $\Rightarrow$ $\phi \wedge \psi$ has exactly 1 sat. assgmt w.p. $\Omega(1)$.

- Need to check idea - does it work?
- But what is k?
- How to "construct efficient ψ"?

Random ψ works [if k is known & $\# \{x \mid \phi(x)=1\} \in [2^{k-2}, 2^{k-1}]$]

Fix x s.t. $\phi(x)=1$

$\Pr[x \text{ satisfies } \psi] = 2^{-k}$

Fix $x' \neq x$ s.t. $\phi(x')=1$

$\Pr[x' \text{ does not satisfy } \psi] = 1-2^{-k}$

$\Pr[x \text{ uniquely satisfies } \psi] = 2^{-k} \cdot (1-2^{-k})^{\#\phi}$

$\geq \frac{1}{10} \cdot 2^{-k}$

$\Pr[\exists x \in \phi^{-1}(1) \text{ s.t. } x \text{ uniquely satisfies } \psi]$

reverse union bound for mutually exclusive events

$\geq \underbrace{2^{k-2}}_{①} \cdot \frac{1}{10} \cdot 2^{-k} \geq \frac{1}{40}!$

① $\#x$ s.t. $\phi(x)=1$

② x uniquely satisfies ψ
$\Rightarrow x'$ does not uniquely satisfy ψ

Efficient ψ?

suffices that $\Pr_{\psi}^{\forall x \neq y} \left[\psi(x) = 1 \ \& \ \psi(y) = 1 \right] = 4^{-k}$

& $\Pr_{\psi} \left[\psi(x) = 1 \right] = 2^{-k}$ $\Bigg\}$ (A)

- Pairwise Independent hash families

$\mathcal{H} = \{ h : \{0,1\}^n \to \{0,1\}^k \}$ is p.w.i. if

$\forall x \neq y \quad \forall a, b \qquad \Pr_{h \in \mathcal{H}} \left[h(x) = a \ \& \ h(y) = b \right] = 4^{-k}.$

- Example $h = \left(\begin{array}{c} \\ k \end{array} \overset{n}{\left[\quad M \quad \right]}, \ \left[b \right] \right)$ $\qquad h(x) = M \cdot x + b$ [over $GF(2)$]

Exercise: Verify.

- $\left[\psi(x) = \text{"} h(x) = 0 \cdots 0 \text{"} \right] \Rightarrow$ (A)

(A) suffices? $\Pr_{\psi} \left[\phi \wedge \psi \text{ uniquely sat. by } x \right]$

$\geq \Pr_{\psi} \left[\psi \text{ sat. by } x \right] - \Pr \left[\exists x' \neq x \text{ s.t. } \psi(x) = \psi(x') = 1 \right]$

$\geq 2^{-k} - (2^{k-1}) \cdot 4^{-k} \geq \frac{2^{-k}}{4}.$

Last hurdle: k unknown!

Solution: Guess!! $k \in_U \{1 \ldots n+1\}$

Final Reduction

$$\phi \longrightarrow (k, \phi \wedge \psi)_{k \in_U [n+1],\ \psi \in \mathcal{H}_{n,k}}.$$

Soundness ϕ UNSAT $\Rightarrow$ $\phi \wedge \psi \in \mathrm{USAT}_N$ (w.p. 1)

Completeness ϕ SAT $\Rightarrow$ $\phi \wedge \psi \in \mathrm{USAT}_Y$ w.p. $\geq \frac{1}{40n}$

$\frac{1}{\mathrm{poly}(n)}$!

OPEN: Amplify Error Probability!

Counting Problems

$f : \{0,1\}^* \to \mathbb{Z}^{\geq 0}$ is a #P function if

$\exists$ ~~Nondet.~~ $M(\cdot,\cdot)$ that is in P such that

$$\forall x \quad f(x) = \left| \cancel{M} \{ y \mid M(x,y) = 1 \} \right| .$$

NP $\subseteq$ #P $\subseteq$ PSPACE

coNP $\subseteq$

BPP

Introduced by Valiant. Why? → Algebraic interest → inherent interest.

Algebra	$\exists$ (OR)	$\to$	$\sum$
	AND $\forall$	$\to$	$\prod$

$$\exists x \ \forall j \quad C_j(x) \text{ satisfied} \Rightarrow \underbrace{\sum_x \prod_j (C_j(x) = 1)}_{\#P!}$$

#P- Complete Problems

- #SAT
- # Vertex Cover
- # Hamiltonian Circuit

$\Big\}$ Karp reduction were "parsimonious"

$\vdots$

$\phi \in SAT? \implies T(\phi) \in V.C.?$

(ϕ, y) satisfied $\implies (T(\phi), \underbrace{T'(y)})$ satisfied

$\uparrow$ for a known number of T'

N sat. assgmts $\implies \underbrace{c(n) \cdot N}$ sat. assignments.

$\Uparrow$ Need this to be efficiently invertible.

- #DNF [# sat. assgmts of DNF formula]

$$\phi \implies \bar{\phi}$$

$$N \text{ sat} \implies 2^n - N \text{ sat.}$$

- Network Reliabity
- Partition functions in Physics
- Bayesian Inference

Permanent

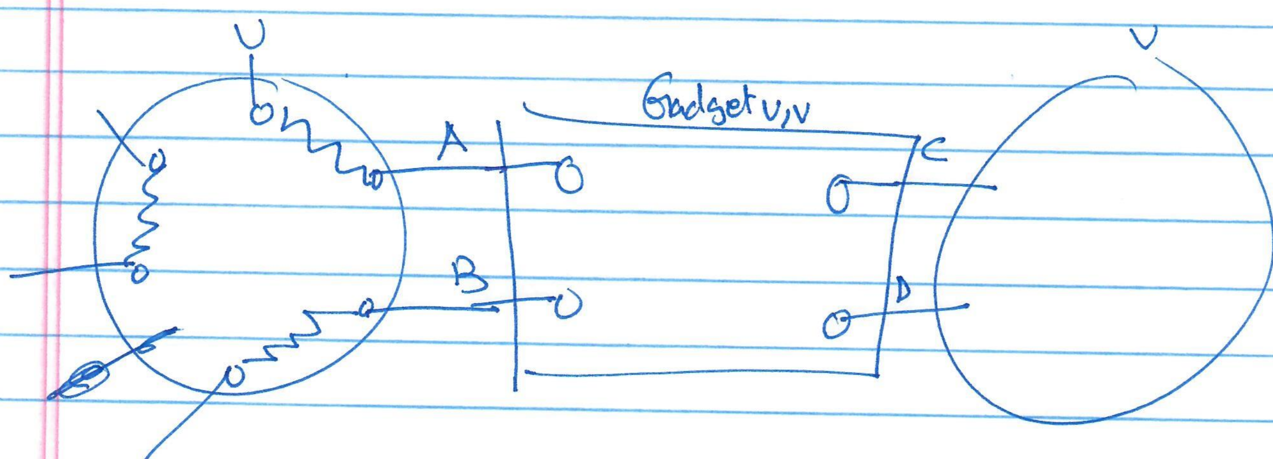
$$M = \left[\; M_{ij} \;\right]_{n \times n} \qquad \mathrm{Perm}(M) = \sum_{\pi \in S_n} \prod_{j=1}^{n} M_{j\,\pi(j)}$$

Combinatorial Interp.: Perm(M) = # perfect matchings in Bip. graph with adjacency matrix M

Valiant: #VC ≤ Perm.

Recall: VC = Subset of vertices that touch every edge.

Key Step 1: Gadgets for vertices + edges



Want: A, B, C, D chosen ⇒ 1 matching in Gadget

A, B chosen ⇒ "

C, D chosen ⇒ "

none chosen ⇒ 1 - - -

only possibilities!

Unfortunately contradictory! No such gadgets exist **if** edges have positive weight.

Valiant's Brilliant Idea: 1 Use negative weight edges.

Permanent ~~still well-defined~~ ~~but~~ no comb. interpretation

2nd Brilliant Idea:

Compute Perm(M) mod p.

Now all #'s positive & allows us to compute

Perm(M) [if p large enough, or by using many p's ....]

Many nice prop. of #P come from permanent.