Probabilistically Checkable Proofs

20/2/08

Def A verifier is a poly-time deterministic algorithm, that recieves an input x and a proof π and accepts/rejects.

The language defined by the verifier V is $L(V)=\{x \mid \exists \pi\ V(x,\pi)=\text{accept}\}$.

Completeness $x \in L \rightarrow \exists \pi \quad V(x,\pi)=\text{acc}$

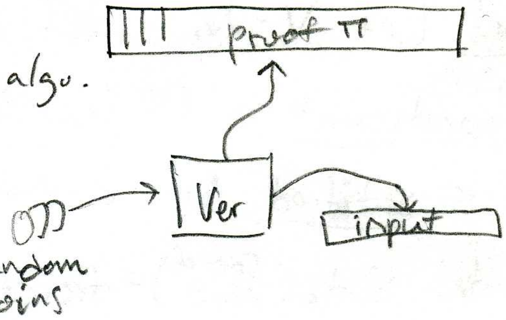
Soundness $x \notin L \rightarrow \forall \pi \quad V(x,\pi)=\text{rej}$

NP - The class of all languages L(V) for some V.

Remark asymmetry between complexity of proof and complexity of verification.

Q What happens if we add randomness to the verifier?

Def An $(r,q)_\Sigma$-restricted verifier (if $\Sigma=\{0,1\}$ we sometimes omit Σ) is a poly-time algo.



1. reads x and r.
2. queries q locations in π.
3. acc/rej

The lang. defined by the verifier V is L, s.t.

Completeness $x \in L \rightarrow \exists \pi \ P_r(V^\pi(x,r)=\text{acc})=1 \quad (c)$

Soundness $x \notin L \rightarrow \forall \pi \ P_r(V^\pi(x,r)=\text{acc})<\frac{1}{2} \quad (s)$

$PCP_{c,s}[r,q]_\Sigma$ = class of all lang. with $(r,q)_\Sigma$-restricted verifier w/ comp. c and soundness s.

- $\bigcup_c PCP[0,n^c] = NP = \bigcup_c PCP[O(\log n), n^c]$
- $PCP[0,0] = P = PCP[O(\log n), 0]$

Easy $PCP[O(\log n), O(1)] \subseteq NP$

$$\boxed{\text{Thm}\quad PCP[O(\log n), O(1)] = NP}$$

We'll start by showing $PCP[\text{polylog}\, n, \text{polylog}\, n] \supseteq NP$.

Hardness of Approximation

An optimization problem maps a set of solutions for each input. Each solution has a value.

Goal: Find a solution with optimal value.

- Max-Clique: instance = graph   sol's: cliques   value: size
- Min-Set-Cover: instance = $U, S_1, \dots, S_m$ Sol's: covers   value: size
- Max-CSP (constraint satisfaction problem)

↳ **Def** Let $V = \{v_1, \dots, v_n\}$ be variables over alphabet Σ, $q \in \mathbb{N}$.

A q-ary constraint is $(\varphi, i_1, \dots, i_q)$ s.t. $i_1, \dots, i_q \in [n]$, $\varphi : \Sigma^q \to \{0,1\}$

It is satisfied by $a : V \to \Sigma$ iff $\varphi(a(v_{i_1}), \dots, a(v_{i_q})) = 1$

Denote $\text{Sat}_a(\text{set of const.})$ = fraction of const. sat. by a.

$\text{Max-CSP}(q)_\Sigma$: instance = var's $V = \{v_1, \dots, v_n\}$, q-ary constraints $C = \{c_1, \dots, c_m\}$ over Σ.

sol's = assign. $a : V \to \Sigma$ value: $\text{sat}_a(C)$

Generalizes Max-3SAT, Max-Cut, Max-3COL.

Def An poly-time algo A α-approximates problem O if given input x, it outputs a solution whose value $A(x)$ satisfies

$\alpha < 1$ $\alpha OPT(x) \leq A(x) \leq OPT(x)$ for maximization

$\alpha > 1$ $OPT(x) \leq A(x) \leq \alpha \cdot OPT(x)$ for minimization

Rem α may be a function of $|x|$.

Known eff. approx

- Max-Clique $\frac{n}{\log^2 n}$
- Min-Set-Cover $\ln n$
- Knapsack $1+\varepsilon \; \forall \varepsilon>0$
- Max-3SAT $\frac{7}{8}$

Def $\text{gap-CSP}(q)_{\Sigma}^{c>s}$ with subscript c,s is the following problem:
Given an instance of $\text{Max-CSP}(q)_{\Sigma}$ (V,C) decide between:

YES: $OPT(V,C) \geq c$

NO: $OPT(V,C) < s$

An algo. is said to solve a gap problem if says YES on YES inst, NO on NO inst.

Claim If there is a reduction from NPC language to gap $\text{CSP}_{c,s}(q)_{\Sigma}$ mapping x to (V_x, C_x) s.t.

$x \in L \rightarrow OPT(V_x, C_x) \geq c$

$x \notin L \rightarrow OPT(V_x, C_x) < s$

Then it is NP-hard to approximate Max-CSP to within $\frac{c}{s}$.

Proof Assume that A is a $\frac{c}{s}$-approx. algo. for Max-CSP.
Eff. algo for L: Given x, run reduction to get (V_x, C_x).
Run A on (V_x, C_x), denote value by v.

If $v < s$, output NO (because if $x \in L$, $OPT(V_x, C_x) \geq c$ so $v \geq \frac{s}{c} \cdot c = s$)

If $v \geq s$, output YES (because if $x \notin L$, $OPT(V_x, C_x) < s$, so $v < s$)

Lemma The following two statements are equivalent: ($1 \geq c > s > 0$ are constants)

(i) $NP = PCP_{c,s}[O(\log n), O(1)]_{\Sigma}$

(ii) There exists a constant q, s.t. gap-$CSP_{c,s}(q)_{\Sigma}$ is NP-hard.

Proof (i) $\Rightarrow$ (ii). Let $L \in NPC$. We will show a reduction

$x \mapsto (V_x, C_x)$

- $x \in L \rightarrow OPT(V_x, C_x) \geq c$
- $x \notin L \rightarrow OPT(V_x, C_x) < s$

Let Ver be an (r,q)-verifier for L ($q = O(1)$, $r = O(\log n)$) given by (i).
Define V_x to be the $\leq 2^r \cdot q$ var's corr. to the proof locations accessible by Ver.

Define C_x as follows: for each $p_1 - p_r \in \{0,1\}^r$ Ver decides on $i_1^{(p)} - i_q^{(p)}$ and on a predicate $\varphi^{(p)}: \Sigma^q \rightarrow \{0,1\}$
Denote $C_p = (\varphi^{(p)}, i_1^{(p)}, \ldots, i_q^{(p)})$ the corr. const.
$C_x = \{C_p \mid p \in \{0,1\}^r\}$. This reduction clearly works.

(ii) $\Rightarrow$ (i) Assume we have a reduction from $L \in NPC$ to gap $CSP_{c,s}(q)_{\Sigma}$
s.t. $x \in L \rightarrow OPT(V_x, C_x) \geq c$
$x \notin L \rightarrow OPT(V_x, C_x) < s$
we need to prove $NP \subseteq PCP_{c,s}[O(\log n), q]_{\Sigma}$.
Enough to show $L \in PCP_{c,s}[O(\log n), q]_{\Sigma}$.
Let Ver work as follows: On input x, run the reduction, getting (V_x, C_x). Interpret the proof π as assign. $a: V_x \rightarrow \Sigma$. Read $\log|C_x|$ random bits to select a random constraint $c \in C_x$. $c = (\varphi, i_1, \ldots, i_q)$. Read proof locations $\pi_{i_1}, \ldots, \pi_{i_q}$ and check if they sat. φ.