Probabilistically Checkable Proofs (PCP)

Charles Carlson

CS 579: Computational Complexity, Spring 2016

Outline

Outline

Recall NP

Definition (**NP**)

A language L is in **NP** if there is a poly-time Turing machine V ("verifier") that, given input x, checks certificates (or membership proofs) to the effect that $x \in L$. That is,

$$x \in L \Rightarrow \exists \pi \text{ s.t.} V^{\pi}(x) = 1$$
$$x \notin L \Rightarrow \forall \pi \ V^{\pi}(x) = 0$$

where V^{π} denotes "a verifier with access to certificate π."

PCP Verifiers

Definition (PCP *verifier*)

Let L be a language and $q, r : \mathbb{N} \to \mathbb{N}$. We say that L has an $(r(n), q(n))-$**PCP** *verifier* if there's a polynomial-time probabilistic algorithm V satisfying:

> *Efficiency:* On input a string $x \in \{0,1\}^n$ and given random access to a string $\pi \in \{0,1\}^*$ of length at most $q(n)2^{r(n)}$ (which we call the *proof*), V uses at most $r(n)$ random coins and makes at most $q(n)$ non-adaptive queries to locations of π. Then it outputs "1" (for "accept) or "0" (for "reject"). We let $V^\pi(x)$ denote the random variable representing V's output on input x and with random access to π.

PCP Verifiers. . .

Definition (PCP *verifier* . . .)

Completeness: If $x \in L$, then there exists a proof $\pi \in \{0,1\}^*$ such that $Pr[V^\pi(x) = 1] = 1$. We call this string π the *correct proof* for x.

Soundness: If $x \notin L$ then for every proof $\pi \in \{0,1\}^*$, $Pr[V^\pi(x) = 1] \leq 1/2$.

We say that a language L is in **PCP**$(r(n), q(n))$ if there are some constants $c, d > 0$ such that L has a $(C \cdot r(n), d \cdot q(n))-$**PCP** verifier.

Non-adaptive vs Adaptive Proof Reading

non-adaptive vs adaptive

Verifiers are *adaptive* or *non-adaptive*:

- *adaptive:* Can select bits to query based on already queried bits.
- *non adaptive:* Selects bits to query based only on input and random tape.

Soundness

Soundness

The $1/2$ constraint is arbitrary. Indeed, changing it to any other constant k $(0 < k < 1)$ will not change the class. Moreover, given a verifier with soundness of $1/2$ using r coins and making only q queries, we can construct a verifier that uses cr coins, makes cq queries and has soundness 2^{-c} by simply repetition.

proof size

The proof length restriction of at most $q2^r$ is inconsequential as such a verifier can look at most this number of locations with nonzero probability.

Outline

The PCP Theorem:

Theorem (The PCP Theorem (LT))

$\mathbf{NP} = \mathbf{PCP}(\log n, 1)$.

Outline

View: Locally testable proofs

locally testable proofs

New proof system with proofs that can be checked at arbitrary location instead of sequentially.

locally testable proofs example

Let $\mathcal{A}$ be an axiomatic system of mathematics for which proofs can be verified by a deterministic TM in time that is polynomial in the length of the proof. Then

$$L = \{\langle \varphi, 1^n \rangle : \ \varphi \text{ has a proof in } \mathcal{A} \text{ of length} \leq n\}$$

is in **NP**. However, the **PCP** Theorem says that L has probabilistically checkable certificates (alternative "proofs"!).

An example: Nonisomorphic Graphs (GNI)

$GNI \in \mathbf{PCP}(poly(n), 1)$

GNI is the language of nonisomorphic graphs. Given two graphs G_0 and G_1 with n vertices, a verifier expects π to contain, for each labeled graph H with n vertices, a bit $\pi[H] \in \{0, 1\}$ corresponding to whether $H \equiv G_0$ or $H \equiv G_1$ (arbitrary if neither case holds). Then the verifier can pick a random bit $b \in 0, 1$ and a random permutation of G_b, H. The verifier accepts iff the corresponding bit of $\pi[H]$ is b. If $G_0 \not\equiv G_1$ then the verifier accepts with probability 1 while if $G_0 \equiv G_1$, then the probability of accepting is at most $1/2$.

Approximation of MAX-3SAT

Definition (Approximation of MAX-3SAT)

For every $3CNF$ formula φ, the *value* of φ, denoted by $val(\varphi)$, is the maximum fraction of clauses that can be satisfied by any assignment to φ's variables. In particular, φ is satisfiable iff $val(\varphi) = 1$.

For every $\rho \leq 1$, an algorithm A is a ρ-approximation algorithm for MAX-3SAT if for every 3CNF formula φ with m clauses, $A(\varphi)$ outputs an assignment satisfying at least $\rho \cdot val(\varphi) m$ of φ's clauses.

View: Hardness of approximation

hardness of approximation

Can show that many **NP** optimization problems are **NP** hard to *approximate*.

Theorem (**PCP** Theorem: Harndess of approximation (HA))

There exists $\rho < 1$ such that for every $L \in NP$ there is a polynomial-time function f mapping strings to (representations of) $3CNF$ formulas such that

$$x \in L \Rightarrow val(f(x)) = 1$$
$$x \notin L \Rightarrow val(f(x)) < \rho.$$

Hardness restated

Corollary

There exists some constant $\rho < 1$ such that if there is a polynomial-time ρ-approximation algorithm for $MAX - 3SAT$ then $\mathbf{P} = \mathbf{NP}$.

The equivalence of both views

Theorem (Equivalence of views)

Hardness of approximation (HA) view and locally testable proofs (LT) view are equivalent.

Outline

Constraint satisfaction problems (CPS)

Definition (Constraint satisfaction problems (CPS))

If q is a natural number, then a *qCSP* instance φ is a collection of functions $\varphi_1, \ldots, \varphi_m$ (called *constraints*) from $\{0,1\}^n$ to $\{0,1\}$ such that each function φ_i depends on at most q of its input locations. That is, for every $i \in [m]$ there exists $j_1, \ldots, j_q \in [n]$ and $f : \{0,1\}^q \to \{0,1\}$ such that $\varphi_i(\mathbf{u}) = f(u_{j_1}, \ldots, u_{j_q})$ for every $\mathbf{u} \in \{0,1\}^n$.

We say that an *assignment* $\mathbf{u} \in \{0,1\}^n$ *satisfies* constraint φ_i if $\varphi(\mathbf{u}) = 1$. The fraction of constraints satisfied by $\mathbf{u}$ is $\frac{\sum_{i=1}^m \varphi_i(\mathbf{u})}{m}$, and we let $val(\varphi)$ denote the maximum of this value over all $u \in \{0,1\}^n$. We say that φ is *satisfiable* if $val(\varphi) = 1$. We call q the *arity* of φ.

CSP ...

3SAT

A generalization of $3SAT$ such that each clause has any form instead of just OR literals. Moreover, clauses can depend on more than just 3 variables. Indeed, $3SAT$ is a subclass such that $q = 3$ and all constraints are ORs.

Gap CSP

Definition (Gap CSP)

For every $q \in \mathbb{N}, \rho \leq 1$, define $\rho - GAPqCSP$ to be the problem of determining for a given $qCSP$-instance φ whether $val(\varphi) = 1$ (in which case we say φ is a YES instance of $\rho - GAPqCSP$) or $val(\varphi) < \rho$ (in which case we say φ is a NO instance of $\rho - GAPqCSP$).

We say that $\rho - GAPqCSP$ is **NP**-hard for every language **L** in **NP** if there is a polynomial-time function f mapping strings to (representations of) $qCSP$ instances satisfying:

Completeness: $x \in L \Rightarrow val(f(x)) = 1$.

Soundness: $x \notin L \Rightarrow val(f(x)) < \rho$.

The hardness of Gap CSP

Theorem (Gap CSP can be **NP**-hard (GAP))

There exist constants $q \in \mathbb{N}, \rho \in (0, 1)$ *such that* $\rho - GAPqCSP$ *is* **NP***-hard.*

Our first goal

We now show that both HA and LT are equivalent to GAP to prove equivalence.

Outline

LT $\Rightarrow$ GAP:

Proof.

Assume that **NP** $\subseteq$ **PCP**$(\log n, 1)$. It suffices to show that $3SAT$ can be reduced to $1/2 - GAPqCSP$ for some constant q. There must exist a $(c\log n, q)$-**PCP** Verifier V for $3SAT$. For every input x and $r \in \{0,1\}^{c\log n}$, let $V_{x,r}$ be the function that on input proof π outputs 1 if the verifier will accept the proof with input x and coins r. Then $V_{x,r}$ depends on at most q locations. Thus, for every $x \in \{0,1\}^n$, the collection $\varphi = \{V_{x,r}\}_{r\in\{0,1\}^{c\log n}}$ is a polynomial-sized $qCSP$ instance. Since V runs in poly-time we can construct φ in poly-time. Finally, by the completeness and soundness of the **PCP** system, if $x \in 3SAT$, then φ will be satisfiable, while if $x \notin 3SAT$, then φ will satisfy $val(\varphi) \leq 1/2$. $\square$

LT $\Leftarrow$ GAP:

Proof.

Suppose that $\rho - GAPqCSP$ is **NP**-hard for some constants $q, \rho < 1$. Consider a language **L** in **NP**. Given an input x, a verifier can run the reduction $f(x)$ to obtain a $qCSP$ instance $\varphi = \{\varphi_i\}_{i=1}^{m}$. It will expect the proof π to be an assignment to the variables of φ, which it will verify by choosing a random $i \in [m]$ and checking that φ_i is satisfied (by making q queries). Clearly, if $x \in L$, then the verifier will accept with probability 1, while if $x \notin L$, it will accept with probability at most ρ. The soundness can be boosted to $1/2$ at the expense of a constant factor in the randomness and number of queries. $\square$

HA $\Rightarrow$ GAP:

Proof.

Since $3SAT$ is a special case of $3CNF$ it follows that HA implies GAP. $\square$

HA $\Leftarrow$ GAP:

Proof.

Now suppose that *GAP* holds. Then let $\epsilon > 0$ and $q \in \mathbb{N}$ be such that $(1-\epsilon) - GAPqCSP$ is **NP**-hard. Let φ be a *qCSP* instance over n variables with m constraints. Each constraint φ_i of φ can be expressed as an *AND* of at most 2^q clauses, where each clause is the *OR* of at most q variables or their negations. Let φ' denote this collection of clauses. If φ is a YES instance then there exists an assignment satisfying all the clauses of φ'. If φ is a NO instance then every assignment violates at least an ϵ fraction of the constraints of φ. Thus, every assignment would violate at least a $\frac{\epsilon}{2^q}$ fraction of the constraints of φ'.

HA $\Leftarrow$ GAP:

Proof.

We can then transform φ' into $3CNF$ form φ'' of at most $qm2^q$ clauses. Observe, completeness holds since if φ is satisfiable then φ' is and hence φ'' is as well. Soundness also holds since if every assignment violates at least ϵ fraction the constraints of φ, then every assignment violate at least an $\frac{\epsilon}{2^q}$ fraction of the constraints in φ', and then every assignment violates at least an $\frac{\epsilon}{q2^q}$ fraction of the constraints of φ''. $\square$

Thus!

Proof view		**Hardness of approximation view**
PCP verifier (V)	$\longleftrightarrow$	CSP instance (φ)
PCP proof (π)	$\longleftrightarrow$	Assignment to variables ($\mathbf{u}$)
Length of proof	$\longleftrightarrow$	Number of variables (n)
Number of queries (q)	$\longleftrightarrow$	Arity of constraints (q)
Number of random bits (r)	$\longleftrightarrow$	Logarithm of number of constraints ($\log m$)
Soundness parameter (typically 1/2)	$\longleftrightarrow$	Maximum of val (φ) for a NO instance
Theorem 11.5 ($\mathbf{NP} \subseteq \mathbf{PCP}(\log n, 1)$)	$\longleftrightarrow$	Theorem 11.14 (ρ-GAPqCSP is **NP**-hard) , Theorem 11.9 (MAX-3SAT is **NP**-hard to ρ-approximate)

Hard to approximate independent set (IS)

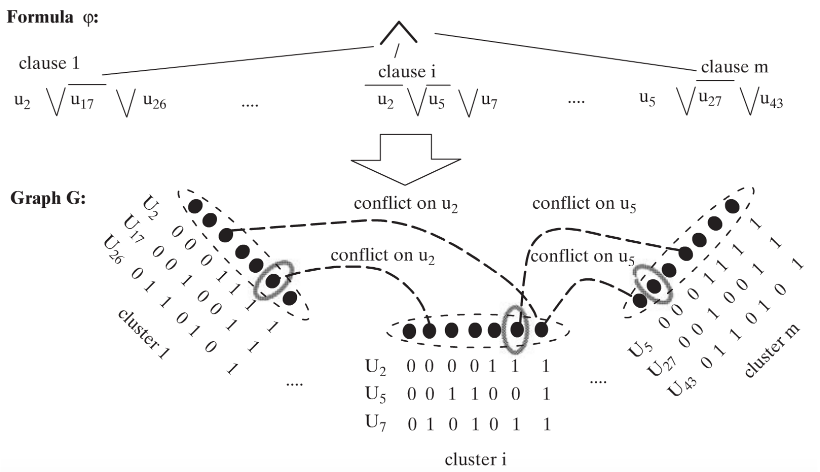
Theorem

There exists some $\rho < 1$ such that computing a ρ-approximation to the independent set problem (INDSET) is **NP***-hard.*

Proof sketch

1. Can transform $3CNF$ φ into n-vertex graph with largest independent set $val(\varphi)\frac{n}{7}$.
2. For any L in **NP**, HA implies that it can be reduced to approximating $MAX - 3SAT$.
3. Then there is a φ such that it is either satisfiable or $val(\varphi) < \rho$ for some $\rho < 1$.

Reduction recall



Hard to approximate independent set (IS)

Theorem

For all $\rho < 1$ computing a ρ-approximation to the independent set problem (INDSET) is **NP***-hard.*

Proof.

For any n-vertex graph G, define G^k to be a graph on $\binom{n}{k}$ vertices corresponding to all k-size subsets of vertices of G. Two subsets are adjacent if their union is an independent set in G. The largest independent set of G^k corresponds to all k-size subsets of the largest independent set of G, IS, and has size $\binom{|IS|}{k}$. If we take the graph produced by the reduction and its k-wise product, the ratio of largest independent sets is $\binom{|IS|}{k} / \binom{\rho|IS|}{k} \approx \rho^k$. Thus, there exists some constant k such that ρ^k is as small as desired. □

Let's not and say we did...

Both the original proof by Arora, Lund, Motwani, Sudan and Szegedy and the new proof by Dinur are long and involved. We will do a brief overview of the proof and its main lemmas. More details are in the book (Chapter 22).

Exponential-sized **PCP** system for **NP**

Theorem (Exponential-sized PCP system for NP)

NP $\subseteq$ **PCP**$(poly(n), 1)$.

Let's not and say we did...

The proof is similar to the original proof of the **PCP** theorem. Uses Hadamard code and language of satisfiable quadratic equations. The verifier will expect two functions (encoding of some assignment to the system) and test if they are *linear* to determine if they are actual encoding. Then accept if they are linear and reject if they are not. Again, full proof is in the book (Chapter 11).

Scaled-up **PCP** theorem

Theorem (Scaled-up PCP theorem)

$\mathbf{PCP}(poly(n), 1) = \mathbf{NEXP}$.

Let's not and say we did...

Again, similar to the original **PCP** theorem proof and using similar ideas as $\mathbf{IP} = \mathbf{PSPACE}$.

Outline

CSP again

Definition (Constraint satisfaction problems (CPS))

For integers $q, W \geq 1$, the $qCSP_W$ problem is defined analogously to the $qCSP$ problem, except the underlying alphabet is $[W] = \{1, 2, \ldots, W\}$ instead of $\{0, 1\}$. Thus constraints are functions mapping $[W]^q$ to $\{0, 1\}$. For $\rho < 1$ we define the promise problem $GAPqCSP_W\rho$ analogously to the definition of $\rho - GAPqCSP$ for binary alphabet.

3SAT

Again, $3SAT$ is the subcase of $qCSP_W$ where $q = 3$, $W = 2$, and the constraints are OR's of the involved literals.

Outline

Proof Sketch

Recall from HA that $\rho - GAPqCPS$ is **NP**-hard for some constants q and $1 - \epsilon = \rho < 1$. Consider the case where ϵ isn't a constant but a function on m (number of clauses). Then observe that if some CSP φ is unsatisfiable $val(\varphi) < 1 - \frac{1}{m}$. Hence, the $(1 - 1/m) - GAP3CSP$ is a generalization of $3SAT$ and is **NP**-hard. The proof grows the value of ϵ until it is as large as some absolute constant independent of m.

CL-reduction

Definition (Complete linear-blowup reduction)

Let f be a function mapping CSP instances to CSP instances. We say that f is a *CL-reduction* if it is polynomial-time computable and, for every CSP instance φ, satisfies:

1. *Completeness:* If φ is satisfiable then so is $f(\varphi)$.
2. *Linear blowup:* If m is the number of constraints in φ, then the new $qCSP$ instance $f(\varphi)$ has at most Cm constraints and alphabet W, where C and W can depend on the arity and the alphabet size of φ (but not on the number of constraints or variables).

Main theorem

Theorem (PCP main Lemma)

There exist constants $q_0 \geq 3$, $\epsilon_0 > 0$, and a CL-reduction f such that for every q_0CSP-instance φ with binary alphabet, and every $\epsilon < \epsilon_0$, the instance $\psi = f(\varphi)$ is a q_0CSP (over binary alphabet) satisfying

$$val(\varphi) \leq 1 - \epsilon \rightarrow val(\psi) \leq 1 - 2\epsilon.$$

Main theorem

	Arity	Alphabet	Constraints	Value
Original	q_0	binary	m	$1-\epsilon$
	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
Lemma 22.4	q_0	binary	Cm	$1-2\epsilon$

Thus!

Proof sketch of **PCP** Theorem

- Reduce from $q_0 CSP$ to $GAPq_0 CSP$ using gap of $1 - 1/m$.
- Amplify the gap using the main theorem and applying f to φ $\log m$ times.
- Results is new instance ψ such that if φ is satisfiable so is ψ and if φ is not satisfiable then

$$val(\psi) \leq 1 - \min\{2\epsilon_0, 1 - 2^{\log m}/m\} = 1 - 2\epsilon_0.$$

- Note, ψ has at most $C^{\log m}$ clauses, which is polynomial with respect to m.

Gap amplification lemma

Theorem

Gap amplification [Dinur] For every $l, q \in \mathbb{N}$ there exist numbers $W \in \mathbb{N}, \epsilon_0 > 0$ and a CL-reduction $g_{l,q}$ such that for every qCSP instance φ with binary alphabet, the instance $\psi = g_{l,q}(\varphi)$ has arity only 2, muses alphabet of size at most W and satisfies

$$val(\varphi) \leq 1 - \epsilon \rightarrow val(\psi) \leq 1 - l\epsilon$$

for every $\epsilon < \epsilon_0$.

Alphabet reduction lemma

Theorem (Alphabet reduction)

There exists a constant q_0 and a CL-reduction h such that for every CSP instance φ, if φ had arity two over a (possibly non-binary) alphabet $\{0, \ldots, W-1\}$ then $\psi = h(\varphi)$ has arity q_0 over a binary alphabet and satisfies

$$val(\varphi) \leq 1 - \epsilon \rightarrow val(h(\varphi)) \leq 1 - \epsilon/3.$$

Proof of main lemma

	Arity	Alphabet	Constraints	Value
Original	q_0	binary	m	$1-\epsilon$
	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
Lemma 22.5 ($\ell = 6$, $q = q_0$)	2	W	Cm	$1-6\epsilon$
	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
Lemma 22.6	q_0	binary	$C'Cm$	$1-2\epsilon$

Raz's **PCP** Theorem

Theorem (Parallel Repetition Theorem)

There is a $c > 1$ such that for every $t > 1$, $GAP2CSP_{W(\epsilon)}$ is **NP***-hard for $\epsilon = 2^{-t}$, $W = 2^{ct}$, and this is true also for 2CSP instances that are regular and have the projection property.*

proof ideas

proof ideas

We construct a new CSP instance ψ taking every original variable and converting it into a t-tuple. Moreover, for every t-tuple of original constraints we construct the constraint

$$\bigwedge_{i=1}^{t} \phi_i(y_i, z_i)$$

where $\phi_i(y_i, z_i)$ is a constraint of the original instance. It is clear that any satisfying assignment in φ is also a satisfying assignment of ψ. Moreover, it can be shown that if at most ρ constraints can be satisfied in φ then at most ρ^{ct} constraints can be satisfied in ψ.

Håstad's Three-Bit **PCP** Theorem

Theorem (Håstad's Three-Bit PCP Theorem)

*For every $\delta > 0$ and every language $L \in$ **NP**, there is a **PCP**-verifier V for L making three (binary) queries having completeness parameter $1 - \delta$ and soundness parameter at most $1/2 + \delta$. Moreover, the test used by V are linear. That is, given a proof $\phi \in \{0,1\}^m$, V chooses a triple $(i_1, i_2, i_3) \in [m]^3$ and $b \in \{0,1\}$ according to some distribution and accepts iff $\pi_{i_1} + \pi_{i_2} + \pi_{i_3} = b$ (mod 2).*

Unique Games Conjecture

Definition (UGC)

This conjecture concerns a special case of $2CSP_W$ in which the constraint function is a permutation on $[W]$. In other words, if the constraint φ_r involves variables i, j, the constraint function h is a bijective mapping from $[W]$ to $[W]$. Then assignment $u_1, u_2, \ldots, u_n$ to the variables satisfies this constraint iff $u_j = h(u_i)$. According to UGC, for every constants $\epsilon, \delta > 0$ there is a domain size $W = W(\epsilon, \delta)$ such that there is no polynomial-time algorithm that given such an instance of $2CSP_W$ with $val(\cdot) \geq 1 - \epsilon$ produces an assignment that satisfies δ fraction of constraints.

For Further Reading I

Sanjeev Arora; Boaz Barak
Computational Complexity: A Modern Approach.
Combridge University Press, 2009.

Luca Trevisan
Inapproximability of Combinatorial Optimization Problems.