CAS CS 535: Complexity Theory

Lecturer: Mark Bun — Fall 2023

Lecture Notes 23:

PCP Theorem, Hardness of Approximation

Reading.

- Arora-Barak § 11.1-11.3

Last time: $\mathbf{IP} = \mathbf{PSPACE}$

1 The PCP Theorem

In a traditional $\mathbf{NP}$ proof, the verifier generally needs to read the entire certificate to be convinced of the statement $x \in L$. For example, to verify that $\varphi \in \mathsf{SAT}$, one generally needs to check all of the bits of an alleged satisfying assignment. When can a (probabilistic) verifier get away with only spot-checking a few random bits of a certificate?

Theorem 1 (PCP Theorem (Informal))**.** *Every language in* $\mathbf{NP}$ *has a probabilistic verifier that reads only* $O(1)$ *bits of its certificate, and is convinced with high probability.*

Let us introduce some definitions to make this statement precise. We'll model a certificate π as an oracle, which can be nonadaptively queried by writing a sequence of indices $i_1, \dots, i_q$ to the oracle tape, and receiving the bit values $\pi[i_1], \dots, \pi[i_q]$ as answers.

Definition 2. For functions $r, q : \mathbb{N} \to \mathbb{N}$, we say a language $L \in \mathbf{PCP}(r(n), q(n))$ if there exists a probabilistic poly-time oracle TM V with the following properties:

Completeness: $x \in L \implies \exists \pi \Pr[V^\pi(x) = 1] = 1$

Soundness: $x \notin L \implies \forall \pi \Pr[V^\pi(x) = 1] \leq 1/2$

Efficiency: V uses $O(r(|x|))$ random coin tosses and makes $O(q(|x|))$ non-adaptive queries to π.

Comments on the Definition:

- We can assume $|\pi| \leq 2^{O(r(n))} \cdot q(n)$, since the verifier can only access this many distinct bits of the certificate.
- As usual, we can amplify soundness to 2^{-c} by repeating verification c times with fresh randomness.
- When q is small (constant), the restriction to non-adaptive queries doesn't make much of a difference, since q adaptive queries can be simulated by 2^q nonadaptive queries. Most positive results about PCP systems use nonadaptive verifiers.

Theorem 3 (PCP Theorem)**.** $\mathbf{NP} = \mathbf{PCP}(\log n, 1)$.

Why is the PCP Theorem nontrivial? Consider the following attempt to construct a probabilistic verifier for SAT. Given an instance φ and π representing a satisfying assignment:

1. Sample q random bits of π
2. Reject if the sampled bits violate some clause of φ, and accept otherwise.

This has perfect completeness, but it fails soundness badly. For example, if $\varphi(x_1, \ldots, x_n) = x_1 \wedge \overline{x_1}$, then the verifier rejects only if it happens to sample $\pi[1]$.

1.1 Proof that $\mathbf{PCP}(\log n, 1) \subseteq \mathbf{NP}$

We'll actually show the stronger statement that $\mathbf{PCP}(r, q) \subseteq \mathbf{NTIME}(2^{O(r)} \cdot q)$. To see this, let $L \in \mathbf{PCP}(r, q)$ with verifier V. Consider the following nondeterministic TM deciding L:

- Nondeterministically guess π of length $2^{O(r(n))q(n)}$
- Run $V^{\pi}(x)$ using all $2^{O(r(n))}$ possible choices of its coin tosses, and accept iff all runs accept.

The opposite containment $\mathbf{NP} \subseteq \mathbf{PCP}(\log n, 1)$ is a deep theorem that would take us a few weeks to cover. (See Chapter 22 of Arora-Barak.) We'll prove a weaker version on Tuesday that illustrates some of the main ideas.

1.2 Motivations for PCPs

The PCP Theorem and related results give us:

- New characterizations of $\mathbf{NP}$ and $\mathbf{NEXP}$
- Outsourcing computation: A server can convince a client of the outcome of a computation with a proof where the verifier needs to check only a few random bits.
- Cryptographic applications: PCPs can be combined with cryptographic tools to yield short non-interactive proofs (or "arguments") that can be stored on blockchains and such.
- Philosophy of math: Every mathematical theorem with an efficiently checkable proof also has one that can be probabilistically verified by checking only 3 lines.
- Many $\mathbf{NP}$-hard problems are similarly hard even to approximate.

2 Hardness of Approximation

Let us recall the notion of $\mathbf{NP}$ optimization problems (from HW2). Let $f(x, y)$ be a poly-time computable objective function. Given an input x, the goal is to compute $\operatorname{argmax}_y f(x, y)$.

Example 4. $\mathsf{MAX3SAT}$: Given 3CNF $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ with n variables and m clauses, output an assignment y that satisfies as many clauses as possible, i.e.,

$$\operatorname{argmax}_{y \in \{0,1\}^n} \sum_{i=1}^{m} C_i(y) =: \operatorname{val}(\varphi).$$

Example 5. MAXINDSET: Given a graph G, output a largest independent set.

Example 6. MAXqCSP: Generalizes MAX3SAT, but each "clause" C_i can instead be an arbitrary Boolean function on q input variables.

The decision versions of all of these problems are $\mathbf{NP}$-complete, so $\mathbf{P} \neq \mathbf{NP}$ implies no poly-time algorithms for (exactly) solving any of these problems.

2.1 Approximation Algorithms

Definition 7. A ρ-approximation algorithm for a maximization problem outputs $\hat{y}$ such that

$$f(x, \hat{y}) \geq \rho \cdot \mathrm{val}(x) := \rho \cdot \max_y f(x, y).$$

Example 8. MAX3SAT has an efficient $(7/8)$-approximation.

Randomized algorithm: A uniformly random assignment y satisfies:

$$\mathbb{E}_y\left[\sum_{i=1}^{m} C_i(y)\right] = \sum_{i=1}^{m} \Pr[C_i(y) = 1] = m \cdot \frac{7}{8} \geq \frac{7}{8} \cdot \mathrm{val}(\varphi).$$

This algorithm can be efficiently derandomized using the "method of conditional expectations."

Using advanced PCP technology, Håstad showed this is optimal:

Theorem 9. *For every $\varepsilon > 0$, if there is a poly-time $(7/8+\varepsilon)$-approximation to* MAX3SAT*, then* $\mathbf{P} = \mathbf{NP}$.

2.2 PCPs vs. Hardness of Approximation

The "standard" PCP theorem we stated has a completely equivalent interpretation in terms of hardness of approximation. To state this equivalence, let us define a decisional (promise) version of the MAXqCSP problem.

Recall that a qCSP instance φ is a collection of functions $C_1, \ldots, C_m$ (called constraints) such that each C_i depends on at most q variables. The value of of the instance is the maximum number of constraints that can be simultaneously satisfied:

$$\mathrm{val}(\varphi) = \max_{y \in \{0,1\}^n} \sum_{i=1}^{m} C_i(y).$$

Definition 10. For $q \in \mathbb{N}$ and $\rho \in (0, 1]$, define the promise problem $\mathsf{Gap}_\rho\mathsf{MAXqCSP}$ by

$$(\mathsf{Gap}_\rho\mathsf{MAXqCSP})_Y = \{\varphi \mid \mathrm{val}(\varphi) = m\}$$
$$(\mathsf{Gap}_\rho\mathsf{MAXqCSP})_N = \{\varphi \mid \mathrm{val}(\varphi) \leq \rho m\}.$$

Theorem 11. $\mathbf{NP} = \mathbf{PCP}(\log n, 1) \iff$ *There exist constants ρ, q such that* $\mathsf{Gap}_\rho\mathsf{MAXqCSP}$ *is* $\mathbf{NP}$*-hard.*

Proof. For the "only if" direction, suppose a language $L \in \mathbf{NP}$ has a PCP verifier V making q queries using $c \log n$ coin tosses. We'll reduce L to $\mathsf{Gap}_{1/2}\mathsf{MAXqCSP}$ as follows. Given an input x, construct a qCSP instance φ with the following correspondence:

Bits of the proof $\pi \mapsto$ input variables to φ

Sequences of random coin tosses $r \in \{0,1\}^{c \log n} \mapsto$ indices of constraints in φ

Number of possible random strings $m = 2^{c \log n} = n^c \mapsto$ number of constraints.

We set $\varphi = \{C_r\}_{r \in \{0,1\}^{c \log n}}$ where

$$C_r(\pi) = V^\pi(x; r).$$

That is, constraint C_r is satisfied iff the verifier accepts the proof π using randomness is r.

This is a qCSP because, for every r, the verifier reads at most q bits of the proof π. So each C_r can only depend on at most q bits of π.

Moreover, the reduction runs in polynomial time because each execution of V does, and because there are at most $m = n^c$ constraints that need to be generated.

Finally, to prove correctness, we check:

$$\begin{aligned} x \in L &\implies \exists \pi \Pr_{r \leftarrow \{0,1\}^{c \log n}}[V^\pi(x; r) = 1] = 1 \\ &\implies \exists \pi \sum_{r \in \{0,1\}^{c \log n}} C_r(\pi) = m \\ &\implies \exists \pi \, \mathrm{val}(\varphi) = m \implies \varphi \in (\mathsf{Gap}_{1/2}\mathsf{MAXqCSP})_Y. \end{aligned}$$

$$\begin{aligned} x \notin L &\implies \forall \pi \Pr_{r \leftarrow \{0,1\}^{c \log n}}[V^\pi(x; r) = 1] \le 1/2 \\ &\implies \forall \pi \, \mathrm{val}(\varphi) \le \frac{m}{2} \implies \varphi \in (\mathsf{Gap}_\rho\mathsf{MAXqCSP})_N. \end{aligned}$$

Now for the "if" direction, suppose $\mathsf{Gap}_\rho\mathsf{MAXqCSP}$ is $\mathbf{NP}$-hard. Let $L \in \mathbf{NP}$, with poly-time computable f computing the reduction. We give a PCP system for L as follows.

On input x, the verifier computes $f(x)$ to obtain a qCSP instance $\varphi = \{C_i\}_{i=1}^m$. It expects the proof π to be a satisfying assignment to φ. To verify the proof, $V^\pi(x)$ samples a random constraint i, and accepts iff $C_i(\pi) = 1$. This all takes polynomial time. We check:

<u>Completeness:</u> If $x \in L$, then $\mathrm{val}(\varphi) = 1$, so there exists π such that $\Pr[V^\pi(x) = 1] = 1$.

<u>Soundness:</u> If $x \notin L$, then $\mathrm{val}(\varphi) \le \rho$, so for every π, we have $\Pr[V^\pi(x) = 1] \le \rho$. (This can be amplified to the constant $1/2$ by repetition.)

<u>Queries:</u> The verifier makes $q = O(1)$ queries to π.

<u>Randomness.</u> The verifier samples a single random constraint, which takes $O(\log m) = O(\log n)$ bits. □