**CS294-4: Fourier Transforms & Theoretical Computer Science** **Spring 1999**

Lecture 7: March 12, 1999

*Lecturer: Madhu Sudan* *Scribe: Anupam Gupta*

This lecture studies Probabilistically Checkable Proofs using techniques developed in the previous lecture.

## 7.1 Probabilistically Checkable Proofs

The notion of a Probabilistically Checkable Proof (henceforth called PCP) is an extension of the conventional idea of a proof, one in which the verifier of the proof wants to read very few bits of a (purported) proof given to him and then decide to accept or reject the proof based on the values of those bits. However, he is allowed to flip some coins in the process, and it is this randomness that allows him to get away without looking at all (or most of) the bits of the proof.

More formally, a $(r, q)$-restricted verifier is one, who when trying to decide whether $\phi \in \mathcal{L}$ (with $|\phi| = n$), tosses at most $r(n)$ coins, and then looks at only $q(n)$ bits of the proof before accepting or rejecting it. Now given a language $\mathcal{L}$, a $\mathrm{PCP}_{c,s}(r, q)$ proof system for $\mathcal{L}$ is defined as follows: there exists a $(r, q)$-restricted verifier $V$ which, on every input $\phi$, satisfies the following conditions:

- **(Completeness)** If $\phi \in \mathcal{L}$, $\exists$ a proof $P$ such that $\Pr[V \text{ accepts } P] \geq c$.
- **(Soundness)** If $\phi \notin \mathcal{L}$, $\forall$ proofs $P'$, $\Pr[V \text{ accepts } P'] \leq s$.

The main theorem we shall see today is the following:

**Theorem 7.1 (Håstad [Hås97])**

$$NP \subseteq PCP_{1-\epsilon,\frac{1}{2}}(\log, 3) \tag{7.1}$$

*where $\epsilon > 0$ is an arbitrarily small constant.*

We will not prove this from scratch, but assume that we can prove the following weaker version of the PCP theorem:

**Theorem 7.2** *There exists a constant $\delta > 0$ such that*

$$NP \subseteq PCP_{1,1-\delta}(\log, 3). \tag{7.2}$$

In fact, we will not really use the fact that the PCP referred to in the above theorem reads only 3 bits; any constant number of query bits would be sufficient for our purposes. Given this theorem, we will now show that the existence of such a verifier for a language in NP (referred to as $V_{PCP}$) will imply the existence of a verifier that satisfies the conditions of Theorem (7.1). The main idea of the entire proof will be that of "Composition of Proof Systems".

### 7.1.1 Multiparty Interactive Proofs

Since the proof has a few conceptual steps, let us first identify an intermediate goal. We would like to exhibit a verifier for NP with the following properties: It asks just two questions from the prover, even though the responses the prover makes to the questions may be quite long; and it errs with very small probability. More formally, we want to prove the following fact:

$$\mathrm{PCP}_{1,1-\delta}(\log, 3) \subseteq \mathrm{MIP}_{1,\gamma}(k\log, 3k) \tag{7.3}$$

where $\gamma > 0$ is an arbitrarily small constant, and $k$ depends only on $\gamma$ and $\delta$.

The $\mathrm{MIP}_{c,s}(q,r)$ in equation (7.3) is a model slightly different from the PCP model, the acronym standing for "Multiparty Interactive Proofs". In this model, the verifier can ask questions to several provers, each of whom is all powerful. On receiving a question from the verifier, the prover may adapt his strategy according to the question; however it cannot see the questions the verifier asks to the other provers. We shall be concentrating on the case where there are two provers and there is only one round of communication. In this case, the verifier performs the following actions:

- Toss some random coins \$.
- Generate questions $q_1$, $q_2$ for the two provers, and a "consistency" function $\pi$.
- On getting the answers $a_1$ and $a_2$, accepts iff $a_1 = \pi(a_2)$.

To specify such a system, we just have to specify what the questions and the function $\pi$ are. The parameters $c$ and $s$ again give the completeness and soundness, $r(n)$ gives the number of random bits and $q(n)$ gives the maximum number of bits received.

We first show a $\mathrm{MIP}_{1,1-\delta/3}(\log, 3)$ for any language in NP. For this, we take the verifier $V_{PCP}$ implied by Theorem (7.2). The verifier for the MIP system, $V_{MIP}$ does the following:

- Tosses coins and generates the 3 locations that $V_{PCP}$ would have queried.
- Asks the values at these locations from the second prover $P_{II}$. Hence $q_2$ is just 3 locations in the proof.
- Chooses one of these locations at random and asks $P_I$ for the value at that location.
- The consistency function is the obvious one: for the location both provers are asked for, check if the values are the same.

It is easy to see that the completeness is 1 as before, and the fact that the soundness is at most $1 - \delta/3$ is left as an exercise.

By this protocol, we have managed to reduce the number of questions and answers to the desired level. However, the probability of error is still very large. One way to take care of this problem would be to independently repeat the above test $k$ times. This would cause the error probability to drop as $(1-\delta/3)^k$; unfortunately, the number of questions would become too large.

A simple-minded fix to reduce the number of questions is to flip all the coins and ask all the $k$ questions in one round itself, instead of flipping the $(i+1)$-th set of coins after the provers have given the $i$-th answers. The hope is that even though the provers know the random choices of the future rounds, this will not help them and we can get the soundness to be $(1-\delta/3)^k$ again. Sadly, this hope is not true; there are several beautiful counterexamples by Fortnow and Feige. However, on the bright side, it does not perform too badly either: in a breakthrough result, Raz [Raz96] proved that the soundness is at most $(1-\delta')^k$, where $\delta'$ is a parameter that is non-zero whenever $\delta$ is non-zero (and is smaller than $\delta$). So even though we cannot get the amplification we would have really liked, we do get a drop of the error probability which is exponential in the number of coin-flips performed, and this is good enough for our purposes.

This shows that we can get a verifier $V_{INTER}$ with the following properties: it asks the provers $P_I$ and $P_{II}$ questions $q_1$ and $q_2$ respectively, and gets replies $a$ (of length $k$ bits) and $b$ (of length $3k$ bits), with the property that the completeness is 1, and the soundness is an arbitrarily small constant $\gamma$ (which depends on $k$). But since there is only one round of communication, we could replace both the provers by tables which list the responses of these provers for each possible question of the verifier, which brings us back to the PCP world.

### 7.1.2 The Håstad Verifier

The above verifier reads the values $a$ and $b$ completely, and then applies $\pi$ to perform the verification. We would like to reduce the actual number of bits read from the tables. For this purpose, we replace each string in the tables by their log codes. Let $A$ and $B$ be the $2^{2^k}$ and $2^{2^{3k}}$ bit strings at the positions corresponding to the two questions. We want to do the following:

- Check that $A$ and $B$ are the long codes of some strings $a$ and $b$ respectively.
- Verify that $a = \pi(b)$.

For this, we define the verifier $V_{COMP}$, which does the following: It picks a function $f$ at random and looks at $A(f)$. Then it picks $g$ and $\eta$ at random (as in the previous lecture), and checks that $A(f)\cdot B(g) = B((f\circ\pi)\cdot g\cdot\eta)$. We will prove that this has completeness $(1-\epsilon)$, and soundness $\frac{1}{2}+f(\gamma,\epsilon)$, where $\epsilon$ is the parameter associated with the noise function $\eta$.

**Exercise 7.1** *Show that the soundness is at most* $\frac{1}{2}$.

Why should this work? Note that at this point, a 5 bit protocol is simple to see: $B$ fixes $A$ (which is just a projection of $B$), so checking if $B$ is a valid long code is enough to verify $A$ as well (this takes 3 bits in all, as described in the previous lecture). After this, we have to check that $A(f) = B(f\circ\pi)$ for a random function $f$, and this can be done by looking at two more places.

But we can do better: Note that ideally $A(f) = f(a)$, which is in turn equal to $f(\pi(b)) = (f\circ\pi)(b)$. Thus we have "essentially" read one bit from $B$, and we can make use of this fact. Showing the claimed completeness is exactly as before. To show the soundness, we claim $V_{COMP}$ accepts with probability $\frac{1}{2}+\delta$, then the (randomized) decoding strategy mentioned in the previous lecture would cause the verifier $V_{INTER}$ to accept with probability $\gamma(\delta)$. Choosing the value of $\delta$ suitably can then give the desired contradiction. Remember that the decoding strategy was the following: given $A$, pick set $S$ with probability $\hat{A}_S^2$, and return a random element $a \in S$.

Suppose $V_{COMP}$ accepts with probability $\frac{1}{2} + \delta$. Then, by a simple counting argument, there are many (a $\delta/2$ fraction of) "good" random strings for $V_{INTER}$ such that the further choices of $f, g$ and $\eta$ (by $V_{COMP}$) are accepting with probability $\frac{1}{2} + \delta/2$. Translating this into the language of expectations, we get that for any good string,

$$\mathbf{E}\left[A(f)B(g)B((f \circ \pi) \cdot g \cdot \eta)\right] \geq \delta. \tag{7.4}$$

However, letting $\chi_S$ denote the codeword $\text{L}\textsc{ong}_S$ corresponding to $S$, we get that the left hand side is just

$$\begin{aligned} & \sum_{S,T_1,T_2} \hat{A}_S \hat{B}_{T_1} \hat{B}_{T_2}\, \mathbf{E}_f\left[\chi_S^k(f)\chi_{T_1}^{3k}(f \circ \pi)\right] \mathbf{E}_g\left[\chi_{T_1}^{3k}(g)\chi_{T_2}^{3k}(g)\right] \mathbf{E}_\eta\left[\chi_{T_2}^{3k}(\eta)\right] \\ = & \sum_{S,T} \hat{A}_S \hat{B}_T^2 (1-2\epsilon)^{|T|} \mathbf{E}_f\left[\chi_S^k(f)\chi_{T_1}^{3k}(f \circ \pi)\right]. \end{aligned}$$

(In the above expressions, the superscripts $k$ and $3k$ indicate which table the codeword was read from.) To handle the $\mathbf{E}_f[\cdot]$ term above, let us define a new projection operator thus: $\Pi_2(T) = \{\pi(b)\}$, is $T = \{b\}$, and $\Pi_2(T_1 \uplus T_2) = \Pi_2(T_1)\Delta\Pi_2(T_2)$. Then it can be shown that the expectation is 1 if $\Pi_2(T) = S$, and is 0 otherwise. Thus the inequality (7.4) reduces to

$$\sum_T \hat{A}_{\Pi_2(T)} \hat{B}_T^2 (1-2\epsilon)^{|T|} \geq \delta.$$

We want to show that, under this assumption, the decoding strategy does well. I.e., there are many "small" sets on which most of the weight lies, and there are some sets where both $\hat{A}_{\Pi_2(T)}$ and $\hat{B}_T$ are large. To formalize the first idea, let the "large" sets $T$ be those with $|T| \geq l$ (where $l$ is to be decided).

$$\sum_{|T| \geq l} \hat{A}_{\Pi_2(T)} \hat{B}_T^2 (1-2\epsilon)^{|T|} \leq \sum_T \hat{B}_T^2 (1-2\epsilon)^l \leq \delta/2 \tag{7.5}$$

if we take $l$ to be about $1/\delta \log(1/\epsilon)$. So the sum over sets $T$ having at most $l$ elements is at least $\delta/2$. Further, we want to ignore $T$ when $\hat{A}_{\Pi_2(T)}$ is small. Note that

$$\sum_{T:\hat{A}_{\Pi_2(T)} < \delta/4} \hat{A}_{\Pi_2(T)} \hat{B}_T^2 \leq \delta/4 \sum_T \hat{B}_T^2 = \delta/4.$$

Using this and inequality (7.5), we get

$$\sum_{T:\hat{A}_{\Pi_2(T)} \geq \delta/4} \hat{A}_{\Pi_2(T)} \hat{B}_T^2 \geq \delta/4 \tag{7.6}$$

and thus

$$\sum_{T:\hat{A}_{\Pi_2(T)} \geq \delta/4} \hat{A}_{\Pi_2(T)}^2 \hat{B}_T^2 \geq \delta^2/16 \tag{7.7}$$

Now we will lower bound the success probability of the decoder. Note that if $S \cap \Pi(T) \neq \phi$, then the probability of success is at least $1/|S||T|$ (which is the probability of picking one of the common elements). Thus the decoder succeeds with probability at least

$$\sum_{S,T:S\cap\Pi(T)\neq\phi} \hat{A}_S^2 \hat{B}_T^2 \frac{1}{|S|}\frac{1}{|T|} \geq (1/l^2) \sum_{S,T:\mathcal{P}(S,T)} \hat{A}_S^2 \hat{B}_T^2 \tag{7.8}$$

where the condition $\mathcal{P}$ is satisfied when $S$ and $T$ have less than $l$ elements, and are not disjoint. But then equation (7.7) gives us that this quantity is at least $\delta^2/(16l^2)$. Now choosing $k$ large enough will cause $\gamma$ to be smaller than this value, which will give the contradiction.

## 7.2 The Hardness of Approximation

There are several other parameters of interest in PCPs, which play an important part in the proving the hardness of approximations. The fact that we use only 3 query bits in the above proof was used by Håstad [Hås97] to show that MAX-3SAT is hard to approximate within $7/8 + \epsilon$ (for any $\epsilon$). Here we briefly look at some of the other important parameters.

### 7.2.1 Low query complexity

If the allowed error probability is $\frac{1}{2}$, then the minimum number of bits we can read is 3. However, what if the error probability is only $2^{-k}$? Clearly, we can get this by doing $k$ independent trials and reading $3k$ bits. Trevisan [Tre98] has shown that the correct value is at most $(\frac{3}{2}k + 2)$, and it is possible that a $(1 + \epsilon)k$ result be shown in the near future.

### 7.2.2 Free bits

In the proof above, we read two bits, and then we had only one value for the third bit that would cause us to accept. Hence the first two bits we read were "free bits". We would like to find the minimum number of free bits to get a certain error probability $p$.

Currently, for $p = \frac{1}{2}$, the best result known is 2, but it is conjectured that the answer is 1. Proving this is equivalent to showing that approximating Vertex Cover to better than 2 is hard. For $p = 2^{-k}$, the correct answer is smaller than $\epsilon k$ (for any $\epsilon$), which is equivalent to showing that clique is hard to approximate within $n^{1-\epsilon}$ (and was shown by Håstad [Hås96]).

### 7.2.3 An open question

Suppose we are given two long codes $A$ and $B$. We fix $f$ and $g$, but do not read $A(f)$ and $B(g)$ just yet. Now we define $h$ thus:

$$h(y) = \begin{cases} f(\pi(y)) & \text{with prob. } p \\ g(y) & \text{with prob. } 1-p \end{cases}$$

and read $B(h)$. With probability $q$, we check if $B(h) = A(f)$, and otherwise we check if $B(h) = B(g)$. For $p = q = \frac{1}{4}$, we can show that the completeness is $3/4$, and the soundness is at most $\frac{11}{16} + \epsilon$. Can we find an analysis of this process using Fourier analysis? The hope is that the proof will also extend to arbitrary values of $p$ and $q$. This problem has implications to showing the Hardness of MaxCut.

## References

[Hås96] Johan Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *FOCS 1996*, pp. 627–636.

[Hås97] Johan Håstad. Some optimal inapproximability results. *STOC 1997*, pp 1–10.

[Raz96] Ran Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, June 1998.

[Tre98] Luca Trevisan. Recycling queries in PCPs and in linearity tests. *STOC 1998*, pp. 299–308.