6.893 Approximability of Optimization Problems November 24, 1999

Lecture 21

Lecturer: Madhu Sudan *Scribe: Ryan O'Donnell*

1 Current versions of the PCP theorem; parallel repetition and improved outer verifiers; Håstad's inner verifier

1.1 Introduction

In the last lecture we proved the PCP theorem ($\mathbf{NP} \subseteq \mathbf{PCP}_{1,1-\epsilon}[\log, 1]$), as well as Arora and Safra's verifier composition theorem. In this lecture, we try to create some verifiers, and get a PCP system for **MAX3SAT**.

1.2 Ideal verifiers for MAX3SAT

Let's consider the problem of creating a PCP system for **MAX3SAT**. Naturally, we are looking for the best possible outer verifier and inner verifier. Let's work on the outer verifier first. In general, one can think of an outer verifier as an MIP system. What are the ideal properties an outer verifier could have?

1. Few provers
2. Small answer sizes
3. Small soundness error
4. Perfect completeness
5. And more...

Let's discuss these properties a bit.

1. How few provers can we have? One is of course insufficient. Let's try for two.
2. This property is not so crucial. A large constant would be fine here.
3. This property however *is* crucial. We want to get arbitrarily close to 0 here.
4. Nothing to say here.

5. We have a somewhat technical desire here. A verifier as described so far will toss some coins, compute two queries q_1 and q_2, and make a circuit C. Then it will make its queries, getting a_1 and a_2 as responses, and then accept if and only if $C(a_1, a_2)$ accepts. Can we get a slightly nicer condition than this? What we will in fact insist on is that the verifier accepts if and only if $\Pi(a_2) = a_1$, were Π is some "projection" function. We call this property the "Π-property".

As we said, these are somehow the ideal properties we desire. If we could get these, this would be all we would want. In fact, we can actually get fairly close to this already.

1.3 A 2-prover MIP from the PCP theorem

Recall that the PCP theorem implies (indeed, is equivalent to) the **NP**-hardness of the gapped version of **MAX3SAT**; given a language $L \in \mathbf{NP}$, there is a reduction mapping strings x to sets of clauses $\phi_x = \{C_1, \ldots, C_m\}$ such that if x is in L then ϕ_x is satisfiable, and if x is not in L, then any assignment violates an ϵ_0 fraction of the clauses in ϕ_x.

Let's build an MIP system in which the provers try to convince the verifier that a collection of clauses $\phi_x = \{C_1, \ldots, C_m\}$ is satisfiable. There will be two provers, P_{clause} and $P_{variable}$. The verifier randomly chooses $j \in \{1, \ldots, m\}$ and $l \in \{1, 2, 3\}$. It then asks the provers one question each. The verifier asks P_{clause} for a truth assignment to the variables in C_j; its response a_2 is three bits indicating one of the 7 possible assignments. The verifier asks $P_{variable}$ for a truth assignment for the lth variable appearing in clause C_j; its response a_1 is one bit. Then the verifier checks that a_1 is consistent with a_2, and accepts if and only if it is. (Note that this is a projection.)

How does this MIP system do with respect to our desired properties? It has two provers, as desired. It's answer sizes are very small: three bits. It has perfect completeness. It has the Π-property. The only problem is that it has very bad soundness. If we assume that when ϕ_x is not satisfiable then any assignment violates an ϵ_0 fraction of the clauses in ϕ_x, then the soundness of our MIP system can be as large as $1 - \frac{\epsilon_0}{3}$. Although this is very high, at least this is bounded away from 1. Can we amplify away this error?

1.4 Error amplification

In order to get the soundness smaller, we try the natural idea; namely, we run the protocol many times.

Definition 1 *Given an MIP system with verifier V of the type we just constructed, consider the similar MIP system with verifier $\mathbf{V^{(k)}}$, which works as follows: $V^{(k)}$ picks k independent, random strings, of length equal to the number of random bits used by V. For the ith random string, it simulates V to get the questions $q_1^{(i)}$ and $q_2^{(i)}$ it would have asked, and also the projection function $\Pi^{(i)}$ it would have used. Then it forms $\bar{q}_1 := (q_1^{(1)}, \ldots, q_1^{(k)})$ and $\bar{q}_2 := (q_2^{(1)}, \ldots, q_2^{(k)})$, and asks these two questions of its provers, getting back the answers $\bar{a}_1 := (a_1^{(1)}, \ldots, a_1^{(k)})$ and $\bar{a}_2 := (a_2^{(1)}, \ldots, a_2^{(k)})$. $V^{(k)}$ then accepts if and only if $(\Pi^{(1)}, \ldots, \Pi^{(k)})(\bar{a}_2) = \bar{a}_1$.*

Applying this construction to the specific case of the **MAX3SAT** system we had in the previous section gives a very natural MIP system.

How does this new MIP system fare with respect to our desired properties? We still have two provers. The answer sizes are still small constants (they are blown up only by a factor of k). We still have perfect completeness. We still have the Π-property. How about soundness? If the original soundness was s, do we get that the new soundness, $s^{(k)}$, is s^k? Unfortunately, the answer is no. The following example due to Fortnow and Feige shows that this does not happen.

Consider a two-prover protocol in which the verifier flips two coins, and sends one result to one prover, the other result to the other prover. The provers' job is to agree on a guess for one of the coins' values. More rigorously, the verifier selects $c_1, c_2 \in \{0,1\}$ at random and sends c_1 to P_1 and c_2 to P_2. P_j returns (i_j, G_j), and the verifier accepts if and only if $i_1 = i_2$ and $G_1 = G_2 = c_{i_1}$. Clearly, the maximum acceptance probability for the system is $1/2$: one of the provers has to guess the value of a coin it doesn't see. Now what if we play the game twice? I.e., suppose the verifier now picks four random bits $c_1^{(1)}, c_2^{(1)}, c_1^{(2)}, c_2^{(2)}$, sends $c_j^{(1)}$ and $c_j^{(2)}$ to P_j, and now each prover has to guess a value for a $^{(1)}$ bit and a $^{(2)}$ bit. In this case, the provers can still win with probability $1/2$! P_1 guesses $(1, c_1^{(1)})$, $(2, c_1^{(1)})$, hoping that $c_2^{(2)}$ had the same value as $c_1^{(1)}$. P_2 does something similar; it guesses $(1, c_2^{(2)})$, $(2, c_2^{(2)})$ in the same hope that $c_1^{(1)} = c_2^{(2)}$. Now if this actually happens, then all the provers' guesses are correct and they win. But of course $c_1^{(1)} = c_2^{(2)}$ happens with probability $1/2$.

So we see that iterating the protocol twice did not decrease the error probability. Of course, this protocol wasn't exactly the sort of two-prover MIP system we were discussing previously, but the result transfers: $s^{(k)} \neq s^k$. But we're not out of luck; we have the following theorem due to Ran Raz which shows that iterating decreases soundness exponentially after all:

Theorem 2 *(Parallel Repetition Theorem) In a two-prover MIP system with perfect completeness, for all bounded answer sizes a and all soundness values $s < 1$, there exists some number $s' < 1$ (depending on a and s) such that the soundness $s^{(k)}$ of the k-iterated protocol is less than s'^k.*

We omit the proof as it is very difficult. It is in a sense elementary, though; one uses information theory to bound the amount of information leaked. Papers of Feige & Kilian and then Feige & Verbitsky showed that the dependence on a is necessary.

1.5 Håstad's verifier

So we got what we wanted: By applying the Parallel Repetition Theorem to the verifier we constructed in Section 1.3, we get a two-prover MIP system for **MAX3SAT** with constant (though large) answer sizes, perfect completeness, the Π-property, and arbitrarily small (though constant) soundness. Call the verifier in this system V^{Raz}. How can we tune this MIP system so that the verifier only needs to look at 3 bits of the proof? Right now V^{Raz} asks one question of each of its two provers and gets back two responses, a_1 and a_2, of length ℓ_1 and ℓ_2, say. We now insist that the provers encode their answers. Since ℓ_1 and ℓ_2 are constants, we can feel free to use a very long code. Indeed, we have our provers use the following code:

Definition 3 LONG *is the code mapping ℓ-bit strings to 2^{2^ℓ}-bit strings via $LONG(a) = \langle f(a) \rangle_{f:\{0,1\}^k \to \{0,1\}}$.*

So although our provers' encoded answers are *very* long, they are still of constant length. Now when V^{Raz} gets back the encoded answers A and B, it wants to check: (1) there exists a such that

$LONG(a) = A$; (2) there exists b such that $LONG(b) = B$; and, (3) $\Pi(b) = a$. And we want V^{Raz} to do this with completeness about 1, soundness about 1/2, all while looking at only 3 bits. What follows is Håstad's 1997 construction that accomplishes this.

We think of our answers A and B as tables of function values. Suppose we read 3 bits of B; specifically, $B(g_1)$, $B(g_2)$, and $B(g_1 + g_2)$, where g_1 and g_2 are some binary functions. Naturally we are interested in testing whether $B(g_1) + B(g_2) = B(g_1 + g_2)$. However, such a test doesn't really tell us much. Even if $B(g_1) + B(g_2) = B(g_1 + g_2)$ for all g_1 and g_2, then we can only conclude that $B(g) = \bigoplus_{i=1}^{m} g(b_i)$ for some bits $b_1, \ldots, b_m$.

What we do instead is consider $g_1 + g_2 + \eta$, for some "noise" function η. Specifically, say we pick g_1 and g_2 uniformly randomly from the set of functions from $\{0,1\}^{\ell_2}$ to $\{0,1\}$, but pick η from the same set according to the following skewed distribution: for each $y \in \{0,1\}^{\ell_2}$, we declare $\eta(y)$ to be 1 with probability ϵ, and 0 with probability $1-\epsilon$. Now what happens if we test $B(g_1) + B(g_2)$ against $B(g_1 + g_2 + \eta)$? Supposing our prover tries to cheat us by setting $B(g) = \bigoplus_{i=1}^{m} g(b_i)$ for some bits $b_1, \ldots, b_m$, what is the chance we accept? We accept when the following is 0:

$$
\begin{aligned}
& \bigoplus_{i=1}^{m} [g_1(b_i) + g_2(b_i) + (g_1(b_i) + g_2(b_i) + \eta(b_i))] \\
= \quad & \bigoplus_{i=1}^{m} \eta(b_i)
\end{aligned}
$$

And by the definition of η and an exercise from Lecture 13, this quantity is 0 with probability approaching 1/2 as m gets large. Note that with the noise function thrown in, our completeness drops to $1-\epsilon$. There are some details missing here; for one, it's unclear why the prover can only cheat us in the manner described. But with some Fourier analysis, one can essentially show that we can read and check from the table B in this way with completeness $1-\epsilon$ and soundness $1/2+\epsilon$.

Actually, we have glossed over another point: The prover could cheat by simply making B all zeroes (i.e., taking $m = 0$). To counter this, our verifier will simply assume that the table B satisfies $B(1-g) = 1 - B(g)$. I.e., the prover's table B will only be half as long as we said; the verifier will infer the other half.

Now how do we deal with table A? What the verifier wants is that $A = LONG(a)$, where $a = \Pi(b)$, and b is such that $LONG(b) = B$. So if $A(f) = x$, then that should mean $f(a) = x$, i.e., $f(\Pi(b)) = x$, i.e., $B(f \circ \Pi) = A(f)$. So our verifier can finish its verification by picking f at random and checking whether $A(f) = B(f \circ \Pi)$. In total, this entails looking at 5 bits. It turns out we can squeeze this down to 3 bits by combining the two tests: the verifier picks f, g_2, and η, and then checks if $B(g_2) \oplus B(f \circ \Pi + g_2 + \eta) = A(f)$.

This at last is Håstad's verifier, which reads 3 bits and gets completeness $1-\epsilon$ and soundness $1/2+\epsilon$. It also only uses $O(\log n)$ random bits. As an aside, we mention that in shrinking the soundess from $1-\epsilon$ down to $1/2+\epsilon$, the trade-off came in the proof size. Amusingly, if n was the classical proof size of $3SAT$, it grows to perhaps n^3 with the PCP Theorem, and then perhaps n^{10^6} with the Parallel Repetition Theorem, and then perhaps $2^{2^{10^6}} n^{10^6}$ with $LONG$ codes — and then we read just 3 bits!