CSC 2421: Approximation Algorithms	**Winter 2003**

Lecture 12: April 1st

Lecturer: Charles Rackoff *Scribes: Stephanie Horn and Tobi Kral*

In today's lecture, we will look at

- a nearly linear lower bound for Clique
- an overview of some early PCP results
- a proof that $P^{\#P} \subseteq IP$.

1 A Nearly Linear Lower Bound for Clique

Brief review of steps covered in the last lecture

In the previous lecture, we took $L \in \mathrm{PCP}_{1,s}(d \lg n, b)$ via verifier V_0, and applied the following FGLSS construction to obtain a bound on Clique.

Suppose we are given an instance y, such that $|y| = n$. Let $(r_1, \ldots, r_{n^d})$ be the sequence of binary strings of length $d \lg n$ that represent the possible choices of random bits by V_0.

From y we construct a graph G whose vertices are pairs (i, σ), where $i \in \{1, \ldots, n^d\}$ and σ is an accepting transcript for V_0 on y using random bits r_i. There is an edge between vertices (i, σ) and (i', σ') if σ and σ' do not assign different truth values to any bit of the proof.

Under this construction we obtain a graph with $\leq n^d 2^b$ vertices, and we used it to prove that, if $P \neq NP$, no polytime algorithm approximates Clique within a factor of $\frac{1}{s}$, giving us an approximation gap of $\frac{1}{s}$.

This is a nice result, but we're not quite happy. Ideally, we would like our gap to be close to the size of the graph. We're aiming for a gap that is close to the number of vertices in the graph raised to a constant power.

To accomplish this, we develop a new PCP.

Completion of a nearly linear lower bound for CLIQUE

Suppose that $L \in \mathrm{PCP}_{1,s}(d \lg n, b)$ via verifier V_0, as we had above. We may assume, without loss of generality, that the size of a proof is n^d. (This will be important later.)

For some k (k to be specified later), let V be the probabilistic polytime verifier that

- runs $k \lg n$ simulations of V_0 using independently chosen random bits for each and
- accepts if and only if all of the simulations accept.

Then $L \in \mathrm{PCP}_{1,s^{k \lg n}}(dk(\lg n)^2, bk \lg n)$ via V, and the proof size remains n^d. We can now use a probabilistic reduction to accomplish our original goal of a nearly linear lower bound. We will probabilistically choose a verifier V', which will be related to V but will use fewer random bits.

First, randomly choose $R = (r_1, \ldots, r_m)$, where each r_i is a binary string of length $dk(\lg n)^2$ (m, like k, will be specified later). Since R is chosen as part of the reduction, the $mdk(\lg n)^2$ random bits involved will not be included in the number of random bits used by any verifier.

Now V' will randomly choose $i \in \{1, \ldots, m\}$, then run V on a given instance y and proof π, using r_i. V' will accept if and only if V accepts using random bits r_i.

Note that V' uses a mere $\lg m$ random bits.

Fix $\epsilon > 0$. We want a value for m such that

- if $y \in L$ then, for all choices of R, V' accepts all π with probability 1. (This is independent of the choice of m.)
- if $y \notin L$ then, with probability at least $\frac{1}{2}$ over all choices of R, for all π, V' accepts π with probability less than $s^{k \lg n}(1 + \epsilon)$.

We will now deduce how to choose m.

Consider, for now, a fixed proof π. We want our verifier V' to accept with low probability if $y \notin L$. Say we have a coin that, when flipped, will come up heads with probability $p = s^{k \lg n}$. We want to define m such that if we flip our coin m times, with very high likelihood the fraction of heads is less than $p(1+\epsilon)$. (Specifically, we would like the probability of flipping at least $p(1+\epsilon)$ heads to be less than $\frac{1}{4 \cdot 2^{n^d}}$, where 2^{n^d} is the number of proofs.) Using a well-known Chernoff bound, we obtain

$$Pr(number\ of\ heads\ \geq pm(1+\epsilon)) < 2^{-\Omega(\epsilon^2 pm)}$$

We can now define $m = \frac{\gamma n^d}{\epsilon^2 s^{k \lg n}}$, where γ is such that the probability guaranteed by the Chernoff bound is at most $\frac{1}{4 \cdot 2^{n^d}}$.

Say $y \notin L$. Then for each π, with probability greater than $(1 - \frac{1}{4 \cdot 2^{n^d}})$ over the choice of R, V' accepts with probability less than $(1+\epsilon)s^{k \lg n}$.

Call the pair (R, π) "bad" if V', using R, accepts π with probability at least $(1+\epsilon)s^{k \lg n}$. Then $Pr_{R,\pi}((R,\pi)\ is\ bad) < \frac{1}{4 \cdot 2^{n^d}}$.

Claim: For at least half of the possible sequences of R, (R, π) is good for all π.

Proof. Suppose not. Since there are only 2^{n^d} values for π, $Pr_{R,\pi}((R,\pi)\ is\ bad) > (\frac{1}{2})(\frac{1}{2^{n^d}})$, contradicting the above inequality. □

The number of nodes in the graph of our new construction is at most $N = m 2^{bk \lg n}$ (recalling that V' is only flipping $\lg m$ bits, and that $2^{bk \lg n}$ is an upper bound on the number of accepting transcripts for any particular choice of random bits by V').

The approximation gap for this reduction will be $\frac{1}{s^{k \lg n}(1+\epsilon)}$. Choose k such that $n^{\epsilon k(-\lg s)} > n^d$.

Then we claim that $\frac{1}{s^{k \lg n}(1+\epsilon)} > N^{\overline{\frac{1}{1+(\frac{b}{-\lg s})+2\epsilon}}}$, where we can make $\epsilon > 0$ as small as possible.

(Vague intuition: Imagine that d and ϵ weren't there. By deleting ϵ and setting $n^d = \gamma = 1$, we would have $s^{-k \lg n} = N^{\overline{\frac{1}{1+(\frac{b}{-\lg s})}}}$.)

Theorem 1. *If there is a polytime algorithm for approximating* CLIQUE *on an N node graph with approximation ratio better than $\frac{1}{N^{1+\frac{b}{-\lg s}+\epsilon}}$, then $\overline{NP} \subseteq RP$.*

Proof. Assume otherwise. Say L is NP-complete. Consider the following NP algorithm for $\overline{L}$:

- Given y, choose a sequence $R = (r_1, \ldots, r_m)$.
- Create graph G by the above reduction.
- Approximate the maximum clique size of G (using the assumed algorithm).
- Accept if the approximation of clique size is "small" (i.e. within the lower bound).

This algorithm accepts exactly $\overline{L}$. Also, if $y \in \overline{L}$ then for at least half of the choices for R, it is the case that for all π, V' will accept with low probability and so all cliques will be small.

Hence $\overline{L} \in RP$. $\square$

(Note that $\overline{NP} \subseteq RP$ implies that $NP = ZPP$.)

Improving $\frac{b}{-\lg s}$

We know that $NP = PCP_{1,(\frac{1}{2}+\epsilon)}(O(\log n), 3)$, so we know that we can get a lower bound of $N^{\frac{1}{4+\epsilon}}$. But can we do better?

The immediate answer is yes. In the construction, the significant parameter was not b, but the maximum over all choices of random bits of $\lg(\textit{number of accepting transcripts})$. However, if we are examining b bits, there are at most 2^{b-1} accepting transcripts. (If, after looking at $b-1$ bits, the verifier will accept (or reject) for both values of the b^{th} bit, then there is no need to look at the b^{th} bit. Hence at the last bit there can be only one choice that will be accepted.) So we really get a lower bound of $N^{\overline{\frac{1}{1+\frac{b-1}{-\lg s}+\epsilon}}}$, or $N^{\frac{1}{3+\epsilon}}$.

Def: For a PCP system

- f is the number of free bits, where the number of free bits is the maximum over all random choices of the verifier of $\lg(\textit{number of accepting transcripts})$.
- $\frac{f}{-\lg s}$ is the number of amortized free bits.

Result (Håstad, 96): For all $\epsilon > 0$, there exists a PCP system with completeness $c = 1$, $O(\log n)$ bits of randomness and at most ϵ amortized free bits.

So if $NP \neq ZPP$ we get a lower bound on the approximation ratio for CLIQUE by polynomial time algorithms of $N^{\frac{1}{1+\epsilon}}$.

2 PCP Results

Theorem 2. $P^{\#P} \in IP$

1990: Lund, Fortnow, Karloff, Nisan
Title : Algebraic methods for interactive proof systems
Proceedings of the 31st IEEE Symposium on Foundations of Computer Science, 1-10.

Given a propositional formula F and a number α, there exists a probabilistic verifier, V, that runs in polynomial time and interacts with a prover, such that if α is the number of satisfying assignments for F, then some prover with unlimited computational power will convince V to accept with probability 1. Otherwise, every prover will convince V to accept with probability less than 1/2.

Corollary 3. $TAUT \in IP$

Theorem 4. $PSPACE = IP$

1990: Shamir, A.
Title : $IP = PSPACE$
Proceedings of the 31st IEEE Symposium on Foundations of Computer Science, 11-15.

The challenging part of the proof is to show that $PSPACE \subseteq IP$.

Theorem 5. $NEXP = PCP_{1,1/2}(n^{O(1)}, n^{O(1)})$

1991: Babai, Fortnow, Lund
Title : Nondeterministic exponential time has two-prover interactive protocols
Computational Complexity **1**, 3-40.

This paper shows that when using a two-prover interactive proof system, two all-powerful, noncommunicating provers can convince a randomizing polynomial time verifier that the input x belongs to the language L, in polynomial time. This is equivalent to the stated PCP result.

$NEXP \subseteq PCP_{1,1/2}(n^{O(1)}, n^{O(1)})$ is the harder direction for this proof, and uses Theorem 2.

Theorem 6. $NP \subseteq PCP_{1,1/2}(O(lgn), (lgn)^{O(1)})$

Theorem 7. $NP \subseteq PCP_{1,1/2}(O(lgn), (lg(lgn))^{O(1)})$

Theorem 8. $NP \subseteq PCP_{1,1/2}(n^{O(1)}, O(1))$

Theorem 9. $NP \subseteq PCP_{1,1/2}(O(lgn), O(1))$

1992: Arora, Lund, Motwani, Sudan, Szegedy
Title : Proof verification and hardness of approximation problems
Proceedings of the 33st IEEE Symposium on Foundations of Computer Science, 13-22.

1998:
Title : Proof verification and the hardness of approximation problems
Journal of the ACM, 45(3): 501–555.

Theorem 7 is the result of composing a strong version of Theorem 6 with itself. Theorem 8, on the other hand, is shown independently of Theorems 6 and 7. Theorem 9 combines the results from both Theorems 7 and 8.

3 Proof of $P^{\#P} \subseteq IP$

Given a propositional calculus formula F and a number α, we will show that there is an interactive proof system to prove that α is the number of satisfying truth assignments for F.

Let $F(x_1, ..., x_n)$ be a propositional formula using only $\neg$ and $\wedge$. Assume that the size of F is polynomial in n. We wish to create an arithmetic expression $E(x_1, ..., x_n)$ with variables, integer constants, and operators $+$, $-$ and $*$, such that for all $a_1, ...a_n \in \{0, 1\}$:

$E(a_1, ..., a_n) = 1$ if $\vec{a}$ satisfies F
$E(a_1, ..., a_n) = 0$ if $\vec{a}$ falsifies F

$E(x_1, ..., x_n)$ can be constructed from $F(x_1, ..., x_n)$ using the following simple transformation:

- x_i transforms to x_i
- if F_1 transforms to E_1, and F_2 transforms to E_2 then
  $F_1 \wedge F_2$ transforms to $E_1 * E_2$ and
  $\neg F_1$ transforms to $(1 - E_1)$

The size of E is polynomial in the size of F, so the degree of E is at most n^c, where c is some constant.

Let $F = GF(q)$, where q is prime. F is the prime field of order q, with elements $0, 1, ..., q-1$ and operators $+$ and $*$ modulo q. Pick F large, say $|F| > 2^n$. Note that F should not be too large, for example $|F| < 2^{n+1}$, to allow the algorithm to process elements of F in reasonable time.

View $E(x_1, ..., x_n)$ as an arithmetic expression over F, and $\alpha \in F$.

The goal for the verifier is to interact with a prover that is supposedly proving that

$$\sum_{x_1,...,x_n \in \{0,1\}} E(x_1, ..., x_n) = \alpha.$$

Define $p(x_n) = \sum_{x_1,...,x_{n-1} \in \{0,1\}} E(x_1, ..., x_n)$, a polynomial of degree at most n^c.

The prover produces a polynomial $p'_n(x_n)$ which is supposedly equal to $p(x_n)$. The verifier checks that $p'_n(0) + p'_n(1) = \alpha$. If the verifier finds that the equality does not hold, it rejects. If the verifier finds that the equality holds, it chooses random integer $r_n \in F$, and sets $\alpha_{n-1} = p'_n(r_n)$. The new goal is to show that:

$$\sum_{x_1,...,x_{n-1} \in \{0,1\}} E(x_1, ..., x_{n-1}, r_n) = \alpha_{n-1}$$

This process continues until the problem is reduced to proving $E(r_1, ..., r_n) = \alpha_0$. At this point, the verifier evaluates $E(r_1, ..., r_n)$ by itself and checks that the value is α_0.

Consider the behaviour of the prover and the verifier when trying to prove the goal
$\sum_{x_1,...,x_k \in \{0,1\}} E(x_1, ..., x_k, r_{k+1} ..., r_n) = \alpha_k$:

- If $LHS = \alpha_k$, then there exists a prover that produces the correct polynomial.
- If $LHS \neq \alpha_k$, then $p_k(0) + p_k(1) \neq \alpha_k$. If the verifier does not reject, then $p'_k(0) + p'_k(1) = \alpha_k$. In this case, $p_k \neq p'_k$. Since $p'_k - p_k$ is a non-zero polynomial of degree $\leq n^c$ (and therefore has at most n^c roots), $p'_k(r_k) \neq p_k(r_k)$ with probability at least $1 - \frac{n^c}{|F|}$. As a result, the LHS in the following step will probably not be equal to α_{k-1}.

Thus, if the original statement was true, then there exists a prover that convinces the verifier with probability 1. If the statement is false, the prover will convince the verifier not to reject if at one of the n stages, the false statement is converted to a true statement. This occurs at stage k if a root of $p'_k - p_k$ is randomly selected.

Thus the verifier will accept with probability less than $n\frac{deg(E(x_1,\ldots,x_n))}{|F|} < \frac{n^{c+1}}{2^n}$. $\square$