U.C. Berkeley
CS294: PCP and Hardness of Approximation
Professor Luca Trevisan

Handout N5
February 1, 2006
Scribe: Luca Trevisan

Notes for Lecture 5

These notes are based on my survey paper [1]. L.T.

More on the Inapproximability of Independent Set

In the last lecture we gave the following reduction from PCP to the Independent Set problem.

Theorem 1 (FGLSS Reduction) *If there is a ρ-approximate algorithm for the independent set problem, then every problem in $\mathbf{PCP}_{c,s}[r(n), q(n)]$ can be solved in time poly$(n, 2^{r(n)+q(n)})$, provided $c/s < \rho$.*

Theorem 1 and the PCP Theorem immediately implies that there is no ρ-approximate algorithm for the independent set problem with $\rho < 2$ unless $\mathbf{P} = \mathbf{NP}$. In this lecture we will substantially strengthen this inapproximability result.

1 Graph Product

Consider the following graph product operation. If $G = (V, E)$ is an undirected graph, then define the *k-th power of* G, denoted as $G^k(V', E')$ as the graph whose set of vertices is $V' := V^k$ (the set of k-tuples of vertices of G) and whose set of edges is

$$E' := \{((u_1, \dots, u_k), (v_1, \dots, v_k)) \text{ such that } \exists i.(u_i, v_i) \in E\}$$

Let use denote by $\alpha(G)$ the size of a largest independent set in G. We have the following simple result.

Lemma 2 (Independence Number of Product Graph) *For every graph $G = (V, E)$ and every positive integer k,*

$$\alpha(G^k) = (\alpha(G))^k \tag{1}$$

Proof: Let $S \subseteq V$ be a largest independent set in G. Then $|S| = \alpha(G)$ and S^k is an independent set in G^k, proving

$$\alpha(G^k) \geq |S^k| = (\alpha(G))^k \tag{2}$$

Let now $T \subseteq V^k$ be a largest independent set in G^k and, for $i = 1, \dots, k$, let

$$T_i := \{v \text{ such that } \exists (v_1, \dots, v_k) \in T \ : \ v_i = v\}$$

be the “projection of T on the i-th coordinate.”

Note that each set $T_i \subseteq V$ must be an independent set in G, and that $T \subseteq T_1 \times \cdots \times T_k$, so

$$\alpha(G^k) = |T| \leq |T_1 \times \cdots \times T_k| \leq (\alpha(G))^k \tag{3}$$

□

From the PCP Theorem and the reduction of Theorem 1 we deduced that there is no polynomial time algorithm for the independent set problem achieving an approximation ratio better than two unless $\mathbf{P} = \mathbf{NP}$. From Lemma 2 we can strengthen this result and rule out any constant factor approximation.

Indeed, suppose that, for some constant r, we have an r-approximate algorithm for the maximum independent set problem. On input a graph G, apply the algorithm to the graph G^k. The resulting algorithm will be $r^{1/k}$-approximate for G, and we can make $r^{1/k} < 2$ by choosing k large enough.

2 Error-Reduction

We now consider an alternative approach to prove that the maximum independent set problem has no constant approximation algorithm unless $\mathbf{P} = \mathbf{NP}$.

Suppose that V is a $(O(\log n), O(1))$-restricted verifier for an $\mathbf{NP}$-complete problem, and that V has soundness $1/2$ and completeness 1, as promised by the PCP Theorem. Define a new verifier V' that performs two independent repetitions of the computation of V, and that accepts if and only if both repetitions accept. Then V' has clearly soundness $1/4$ and completeness 1, and it is still $(O(\log n), O(1))$-restricted, thus showing that even an approximation better than 4 is infeasible. If we repeat V a constant number of times, rather than twice, we can rule out any constant factor approximation for the independent set problem.[1]

In general, a verifier that makes $k(n)$ repetitions shows that $L \in \mathbf{PCP}_{1,1/2^{k(n)}}[O(k(n) \cdot \log n), O(k(n))]$, and the reduction to Independent Set produces graphs that have $2^{O(k(n) \cdot \log n)}$ vertices and for which $2^{k(n)}$-approximate algorithms are infeasible. If we let $k(n) = \log n$, then the graph has size $N = 2^{O((\log n)^2)}$ and the infeasible ratio is n, which is $2^{\Omega(\sqrt{\log N})}$. So, if we have an algorithm that on graphs with N vertices runs in polynomial time and has an approximation ratio $2^{o(\sqrt{\log N})}$, then we have an $O(n^{O(\log n)})$ algorithm to solve 3SAT, and $\mathbf{NP} \subseteq \mathbf{QP}$. More generally, by setting $k(n) = (\log n)^{O(1)}$, we can show that if there is an $\epsilon > 0$ such that Independent Set can be approximated within a factor $2^{O((\log n)^{1-\epsilon})}$ then $\mathbf{NP} \subseteq \mathbf{QP}$.

[1] It is instructive to note that the graph obtained by applying the reduction of Theorem 1 to the verifier of the PCP Theorem and then applying a k-fold graph product it is *identical* to the graph obtained by a k-fold sequential repetition of the verifier of the PCP Theorem followed by an application of the reduction of Theorem 1.

3 Error-Reduction Using Random Walks

To prove an even stronger hardness of approximation result, we want to reduce the error probability of the verifier of the PCP Theorem without increasing too much the *randomness complexity* of the verifier, which is the main parameter affecting the size of the final graph.

Towards this goal we need a result about *random walks on expander graphs.* If $G = (V, E)$ is a regular graph, then a *length-k random walk in G* is a $k+1$-tuple of vertices $(v_0, \ldots, v_k)$ in G that is selected according to the following distribution: we select v_0 uniformly at random in V, then we select v_1 as a uniformly chosen neighbor of v_0, and, for $i = 2, \ldots, k$ we select v_k as a uniformly chosen neighbor of v_k.

In future lectures we will prove the following result.

Theorem 3 (Random Walks in Expanders) *There is a constant d such that for every n there is a d-regular graph $G_n = (V_n, E_n)$ with n vertices such that for every set $B \subseteq V_n$, $|B| \leq n/2$, and for every k, the probability that all the vertices of a random walk $(v_0, \ldots, v_k)$ are contained in B is at most $(2/3)^k$.*

Furthermore, there is an algorithm that, on input n, runs in time polynomial in n and outputs G_n.

From the PCP Theorem and the above result we have

Theorem 4 (PCP Theorem With Low Soundness) *For every $k(n)$,*

$$\mathbf{NP} = \mathbf{PCP}_{1,\left(\frac{2}{3}\right)^{k(n)}}[O(k(n) + \log n), O(k(n))] \tag{4}$$

Proof: Let L be a language in $\mathbf{NP}$. From the PCP Theorem we have the existence of a verifier V witnessing $L \in \mathbf{PCP}_{1,1/2}[O(\log n), O(1)]$. Define the verifier V' as follows: on input an instance x of length n and given oracle access to a proof w, let $r(n) = O(\log n)$ be the number of random bits used by V given x and w.

The verifier V' first constructs the graph G_R, where $R := 2^{r(n)} = \text{poly}(n)$ is the number of possible random strings used by V given x and w we identify the set of vertices of G_R with the set $\{0,1\}^{r(n)}$. Then V' selects a random walk of length $k(n)$ in G_R, a task that requires $\log R + O(k(n)) = O(\log n + k(n))$ random bits. Finally, let $z_0, \ldots, z_k$ be the vertices of the random walk: V' simulates $V^w(x)$ on each of the $k+1$ random strings $z_0, \ldots, z_k$. Finally, V' accepts if and only if all the simulated computations accept.

It is clear that V' makes at most $O(k(n))$ oracles accesses into w and that it uses at most $O(\log n + k(n))$ random bits. If $x \in L$ and w is a witness that makes V accept with probability 1, then $V'^w(x)$ also accepts with probability 1. Finally, consider the case $x \notin L$ and let w be an arbitrary oracle. Then $V^w(x)$ accepts with probability at most 1/2. Let $B \subseteq \{0,1\}^{r(n)}$ be the set of at most $R/2$ random strings that make $V^w(x)$ accept. Then the probability that $V'^w(x)$ accepts is the probability that all the vertices of a length-k random walk in G_R are contained in B, an event that (by Theorem 3) happens with probability at most $(2/3)^k$. $\Box$

If we choose $k(n) = \log n$ and then apply the reduction of Theorem 1 then we have a graph of size $2^{O(k(n)+\log n)} = n^{O(1)}$ for which an approximation ratio of n is infeasible. This shows the following result.

Theorem 5 *There is a constant $c > 1$ such that if there is a polynomial time n^c-approximate algorithm for Independent Set then* $\mathbf{P} = \mathbf{NP}$.

References

[1] Luca Trevisan. Inapproximability of combinatorial optimization problems. Technical Report TR04-065, Electronic Colloquium on Computational Complexity, 2004. 1