15-854: Approximations Algorithms **Lecturer:** Anupam Gupta
Topic: Inapproximability **Date:** 12/05/2005
Scribe: Karl Wimmer

24.1 Inapproximability

In this course, we have seen approximation algorithms applied to a wide variety of NP-hard problems. When we obtain a performance guarantee for such an algorithm applied to a problem, we must wonder whether or not we could do any better. In lecture 2, we saw the idea of inapproximability applied to the K-center problem. It was shown that for any $\epsilon > 0$, the K-center problem is not (2 - ϵ)-approximable unless P = NP. In this lecture, we will focus other problems and prove lower bounds on the performance guarantee of any approximation algorithm for these problems.

24.2 Reductions with gaps

In order to show that problem Π is NP-hard, we reduce a different NP-hard problem Π_2 to problem Π. For example, take Π to be the decision version of the Independent Set problem: Given $G = (V, E)$ and an integer k, does there exist an independent set of size at least k in G?. To show that this problem is NP-hard, we can perform a polynomial time reduction f from some problem known to be NP-hard. Here we will take Π_2 to be SAT.

For the reduction to be correct, the reduction must map an instance I of SAT into some graph G such that ϕ is satisfiable iff G has an independent set of size at least k_1. If ϕ is not satisfiable, then G has no independent set of size greater than k_2.

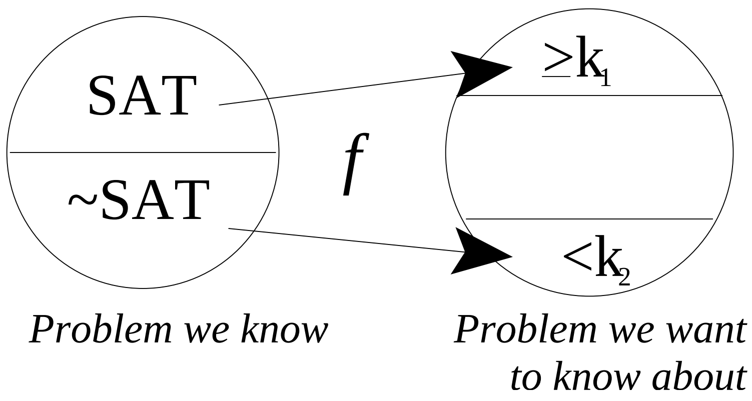

Figure 24.2.1: The kind of mapping we want.

The critical observation is that k_1 need not equal k_2, hence the subscripts. This is the idea of a "gap"; there may exist values of α such that f maps no instances of SAT into a graph G whose largest independent set has size α. We can use this idea to prove inapproximability results. In the claim, we replace the independent set problem with an arbitrary problem Π.

Claim 24.2.1 *If there exists a function f as described above, and Π has an approximation algorithm A with performance guarantee $\rho < \frac{k_1}{k_2}$, then P=NP.*

Proof: Take any instance I of SAT and compute $f(I)$. Since $f(I)$ is an instance of Π, we can apply A to $f(I)$. Let $A(f(I))$ be the value returned by A on instance $f(I)$. If I is satisfiable, then the optimal value of $f(I)$ is at least k_1, by definition of f. It follows that $A(f(I)) \geq \frac{k_1}{\rho} > k_2$. If I is not satisfiable, then $A(f(I)) \leq k_2$. Thus the value of $A(f(I))$ completely determines whether or not I is satisfiable. Since all computations are done in polynomial time, the above is a polynomial time algorithm for SAT. Thus P=NP. ■

As in lecture 2, we can also take Π_2 to be an inapproximable problem, and find a mapping f that exhibits a "gap".

24.3 The PCP Theorem

PCP stands for "Probabilistically Checkable Proofs." We want to show that we can bound the probability of making a mistake, while doing very little work.

To apply the PCP theorem, it is helpful to cast the notion of NP in terms of languages.

Definition 24.3.1 *We say that a language L is in NP if there exists a polynomial p and an algorithm A such that if an instance $x \in L$ then there exists a proof y such that $|y| \leq p(|x|)$ and $A(x, y)$ returns YES in time $p(|x|)$. Also, if $x \notin L$, then for all y, $A(x, y)$ returns NO.*

So in the above definition, we could say that L is the language of satisifiable instances of 3SAT. Then x is an instance of 3SAT, y is a proof that x really is in 3SAT, and A is an algorithm that verifies that y is a valid proof that x belongs to L.

We also need the notion of a probabilistic proof system.

Definition 24.3.2 *A probabilistic proof system is described by a verifier. The verifier uses $r(n)$ random bits and reads $q(n)$ bits of a proof, where n is the length of the instance. If a statement is true, the verifier accepts the statement with at probability at least $c(n)$ (completeness). If the statement is false, the verifier accepts the statement with at most probability $s(n)$ (soundness).*

Fact 24.3.3 *Any language in NP has a probabilistic proof system with $r(n) = 0$, $q(n) = poly(n)$, and $c(n) = 1$, and $s(n) = 0$.*

The above fact simply states that by using no randomness and reading the entire proof, we can with probability 1 accept all correct proofs and reject all incorrect proofs.

However, by increasing the amount of randomness, perhaps we can decrease the number of bits that we need to check while maintaining acceptable levels of completeness and soundness. For PCP applications, we usually require the completeness to be 1 and the soundness to be $\frac{1}{2}$.

We now state the PCP theorem.

Theorem 24.3.4 *SAT has a probabilistic proof system that uses $O(\log n)$ random bits and reads $O(1)$ bits of the proof. If $x \in L$, there exists a proof such that the probability that the verifier accepts is 1. If $x \notin L$, for all proofs, the verifier accepts with probabity at most $\frac{1}{2}$. The probabilities are* *taken over the $O(\log n)$ random bits.*

The PCP theorem says that it is sufficient to use a small amount of randomness and read a even smaller amount of the proof.

The following extension of the PCP theorem is due to Håstad. We will not prove the theorem here. This extension of the PCP theorem will allow us to prove the inapproximability of some problems.

Theorem 24.3.5 *3SAT has a probabilistic proof system that uses $O(\log n)$ random bits and reads 3 bits of the proof, where n is the length of the proof. For any $\epsilon > 0$, if $x \in L$, there exists a proof such that the probability that the verifier accepts is at least $1-\epsilon$. If $x \notin L$, for all proofs, the verifier accepts with probabity at most $\frac{1}{2}+\epsilon$. The probabilities are taken over the $O(\log n)$ random bits.*

At first glance, it seems that Håstad's version of the PCP theorem is weaker than the original PCP theorem, as the probabilities go in "bad" directions by an additive constant ϵ. However, what we gain is information in how the random bits are used to determine whether or not to accept the proof. It is this knowledge that will allow us to prove inapproximability results.

The $O(\log n)$ random bits determined in the theorem are not random bits of the proof, but simply coin flips. From these $O(\log n)$ coin flips, we get 3 positions i, j, k to read in the proof y, as well as some bit b. These bits are functions of x. We accept y if and only if we satisfy a linear test, defined next.

Definition 24.3.6 *A linear test $T(y, i, j, k, b)$ is one that reads the PCP y in 3 positions i, j, and k, and outputs YES iff $y_i \oplus y_j \oplus y_k = b$.*

So we perform this linear test and decide whether or not to accept y. However, since the size of the proof is bounded by a polynomial in n, there are a polynomial number of outcomes for the $O(\log n)$ random bits. So using only logarithmic randomness allows us to enumerate all possible outcomes in polynomial time. This is formalized below:

Corollary 24.3.7 *Given any instance ϕ of 3SAT and $\epsilon > 0$, one can write down $N = 2^{O(\log n)} =$ poly(n) 4-tuples (i_t, j_t, k_t, b_t) in polynomial time such that the following statements hold::*

(i) If ϕ is in 3SAT, then there exists y such that the fraction of the linear tests $\{T_t(y) = T(y, i_t, j_t, k_t, b_t)\}_{t=1}^{N}$ that outputs YES is at least $(1-\epsilon)$.

(ii) If ϕ is not in 3SAT, then for all y, the fraction of the linear tests $\{T_t(y)\}$ that outputs YES is at most $(\frac{1}{2}-\epsilon)$.

Proof: Consider each possible random sequence R_t of coin tosses. Each gives us a linear test $T_t(y)$. The result is immediate by Håstad's extension of the PCP theorem. ■

Fact 24.3.8 *Notice that if we could set $\epsilon = 0$ above, then P = NP. For correct proofs, we would solve the system of all linear tests using Gaussian elimination over $\mathbb{Z}_2$.*

This lays for groundwork for constructing a "gap". In our reductions, we will make use of the fact that satisfiable instances always yield a high fraction of satisfied linear tests, where unsatisfiable instances always yield a bounded fraction of satisfied linear tests.

24.4 Using the PCP theorem to establish inapproximability

The linear tests from the previous sections look a lot like 3SAT clauses. A natural first problem that we will attack using Håstad's extension of the PCP theorem is MAX3SAT. We will prove that we can not get a constant factor approximation to MAX3SAT better than the random assignment algorithm.

Claim 24.4.1 *MAX3SAT can not be approximated to within $(\frac{8}{7} - \epsilon)$ for any $\epsilon > 0$, unless P=NP.*

Proof: Take any instance ϕ of 3SAT. Consider all N possible linear tests generated by each R_t in Håstad's extension of the PCP theorem. We will construct a new instance $f(\phi)$ of MAX3SAT. For the linear test $y_i \oplus y_j \oplus y_k = 0$, include 4 clauses $(y_i \vee y_j \vee \neg y_k) \wedge (y_i \vee \neq y_j \vee y_k) \wedge (\neg y_i \vee y_j \vee y_k) \wedge (\neg y_i \vee \neg y_j \vee \neg y_k)$ in $f(\phi)$. For the linear test $y_i \oplus y_j \oplus y_k = 1$, include the other 4 possible clauses not in the first case. So $f(\phi)$ will have $4N$ clauses.

Notice that if $y_i \oplus y_j \oplus y_k = b$ is satisfied, the 4 clauses generated by that linear test are all satisfied. Further, if $y_i \oplus y_j \oplus y_k = b$ is not satisfied, then exactly 3 of the 4 clauses generated by that linear test are satisfied.

Now we can prove inapproximability. If $\phi \in$ 3SAT, then by Håstad, at least $((1-\epsilon)4 + 3\epsilon)N = (4-\epsilon)N$ clauses will be satisfiable by some assignment in $f(\phi)$. If $f(\phi) \notin$ 3SAT, then at most $(4(\frac{1}{2}+\epsilon) + 3(\frac{1}{2}-\epsilon))N = (\frac{7}{2}+\epsilon)N$ clauses will be satisfiable in any assignment in $f(\phi)$.So if we had any approximation algorithm for MAX3SAT with performance guarantee better than $\frac{8}{7}$, we could choose a small enough ϵ so that the above process distinguishes satisfiable instances of 3SAT from nonsatisfiable instances. Thus MAX3SAT is hard to approximate strictly within $\frac{8}{7}$ unless P=NP. ■

We can use this theorem to show an inapproximability result for Vertex Cover.

Claim 24.4.2 *Vertex Cover can not be approximated to within $(\frac{7}{6} - \epsilon)$ for any $\epsilon > 0$, unless $P = NP$.*

Proof: Take any instance ϕ of 3SAT. We will construct a graph $f(\phi)$ with $4N$ vertices. Consider all N possible linear tests generated by each R_t in Håstad's extension of the PCP theorem. Let $G = (V, E) = f(\phi)$. For each linear test, there are 4 possible satisfying assignments to the 3 y variables. For each linear test, add a complete graph on 4 vertices to V, and label each vertex with one of the satisfying assignments. The case for $b = 0$ is shown in the figure below. At this point, G is the disjoint union of N copies of K_4. Now add an edge between two vertices if the assignments on their labels conflict, that is, if $u \in V$ has y_i set to 0 and $v \in V$ has y_i set to 1, include the edge (u, v) in E.

Now we are set to show inapproximability. If ϕ is in 3SAT, then at least $(1-\epsilon)N$ of the linear tests are satisfied. Since all of these linear tests are simultaneously satisfied, it follows that the assignments do not conflict, so we can take the $(1-\epsilon)N$ vertices corresponding to these assignments in each linear tests as an independent set in G. So G has an independent set of size $(1-\epsilon)N$. This means that there exists a vertex cover of size less than $|V| - (1-\epsilon)N = 4N - (1-\epsilon)N = (3+\epsilon)N$. If ϕ is not in 3SAT, then at most $(\frac{1}{2}+\epsilon)N$ of the linear tests are satisfied. In the same way as above, this corresponds to an independent set of size at most $(\frac{1}{2}+\epsilon)N$. But note that since each linear test generates a complete graph on 4 vertices, all independent sets in G are generated in this way. Thus the maximum size of an independent set in G is $(\frac{1}{2}+\epsilon)N$, and thus every vertex cover

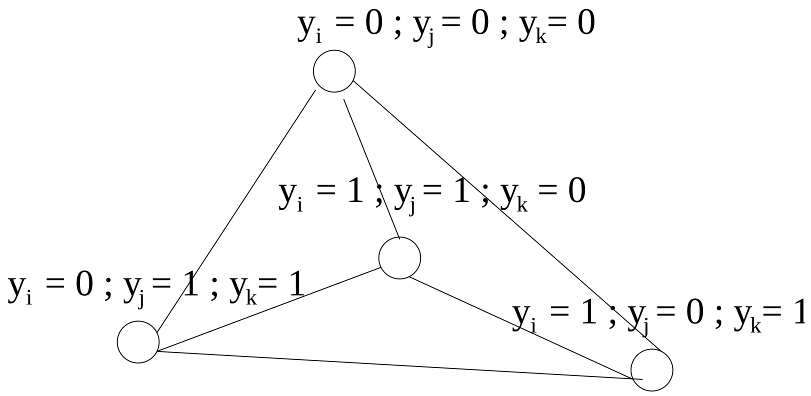

Figure 24.4.2: The resulting complete graph with vertex labels

has size at least $|V| - (\frac{1}{2} + \epsilon)N = 4N - (\frac{1}{2} + \epsilon)N = (\frac{7}{2} - \epsilon)N$. If we had an approximation algorithm for Vertex Cover with a performance guarantee better than $\frac{7}{6}$, we could use the above process with a sufficiently small ϵ to solve 3SAT. Thus, unless P=NP, Vertex Cover can not be approximated to strictly within $\frac{7}{6}$. ■

24.5 The Parallel Repetition Theorem

Suppose that we have a probabilistic proof system as in the PCP theorem. The completeness is 1, so we can repeatedly ask the verifier if our proof is correct. This drives down the probability that we accept a false proof. However, this increases the amount of randomness we need. One idea is to encode the instance and the proofs in such a way such that the probability of accepting a false proof decreases. In essence, we are performing our repetitions in parallel, hence the name of the Parallel Repetition Theorem. The idea behind the parallel repetition theorem involves a two-prover proof system; for details, see [2].

24.6 Conclusions

The PCP theorem itself was not immediately useful for showing inapproximability, but Håstad's extension allowed us to quite easily show inapproximability results for MAX3SAT and Vertex Cover. Variants of the PCP theorem have been applied to many NP-hard problems. See [1] for one development of the PCP theorem and its variants along with some inapproximability results not shown here.

References

[1] Håstad, J. *Some optimal inapproximability results,* In Journal of ACM, Vol. 48, 2001, pp. 798-859.

[2] Raz, R. *A parallel repetition theorem,* In SIAM Journal on Computing, Vol. 27, No. 3, 1998, pp. 768-803.