Lecturer: Hung Q. Ngo
Last update: November 25, 2006

Hardness of Approximation

1 Overview

To date, thousands of natural optimization problems have been shown to be $\mathbf{NP}$-hard [11,18]. Designing approximation algorithms [5, 27, 37] has become a standard path to attack these problems. For some problems, however, it is even $\mathbf{NP}$-hard to approximate the optimal solution to within a certain ratio. Approximating the VERTEX COVER to within 1.3606 is $\mathbf{NP}$-hard. In an extreme case, the TSP problem cannot even be approximated at all, since deciding if there is a TSP tour of length 0 is already $\mathbf{NP}$-hard (the HAMILTONIAN CIRCUIT problem).

Until 1990, few inapproximability results were known. To prove a typical inapproximability result such as MAX-CLIQUE is not approximable to within some ratio r unless $\mathbf{P} = \mathbf{NP}$, a natural direction is to find a reduction from some $\mathbf{NP}$-complete problem, say 3SAT, to MAX-CLIQUE which satisfies the following properties:

- given a 3CNF formula ϕ, the reduction constructs in polynomial-time a graph G_ϕ
- there is some polynomial-time computable *threshold* t such that
  - if ϕ is satisfiable, then G_ϕ has a clique of size at least t
  - if ϕ is not satisfiable, then G_ϕ does not have any clique of size t/r or more.

If MAX-CLIQUE is r-approximable, then one can use this r-approximation algorithm, along with the reduction above, to decide if a 3CNF formula ϕ is satisfiable. We simply run the approximation algorithm on G_ϕ. If the answer is t/r or more, then ϕ is satisfiable; otherwise ϕ is not.

Current techniques for proving $\mathbf{NP}$-hardness seem inadequate for this kind of *gap-producing* reductions. Intuitively, the reason is that non-deterministic Turing Machines are sensitive to small changes: the accepting computations and rejecting computations are not very far from one another (no gap). In 1990, the landmark work by Feige, Goldwasser, Lovász, Safra, and Szegedy [15] connects probabilistic proof systems and the inapproximability of $\mathbf{NP}$-hard problems. This has become known as **the** PCP connection. A year later, the PCP theorem - a very strong characterization of $\mathbf{NP}$ - was proved with the works of Arora and Safra [4], Arora, Lund, Motwani, Sudan, and Szegedy [3]. A plethora of inapproximability results using the PCP connection follow, some of them are optimal. See the historical notes section for more related references.

In the rest of this note, we will present some basic techniques of proving hardness of approximation. It will require a good deal of familiarity with the area to go beyond the basics presented here and to prove more sophisticated inapproximability results.

2 Basic notions

2.1 NPO Problems, Approximation Algorithms, and Approximation Ratios

We begin with several definitions laying out a framework to analyze inapproximability results.

Definition 2.1 (NPO problems). Following Johnson [], an *NP optimization problem* Π has five components:

1. A set I_Π of instances which are polynomial time recognizable, i.e. we can tell in polynomial time if an input x is an instance of Π or not.
2. For each instance $x \in I_\Pi$, there is a polynomial time recognizable set $S(x)$ of feasible solutions to x, i.e. we can tell in polynomial time if a solution y is a feasible solution of instance x or not.
3. A polynomial-time computable cost function cost_Π which assigns a positive integer to each (instance, feasible solution) pair.
4. An indication of whether Π is a *maximization* or a *minimization* problem.
5. A polynomial time algorithm which gives some feasible solution to each instance x. This is to ensure that the problem has *some* polynomial time approximation algorithm.

For each instance x, let $\Pi(x)$ denote the optimal objective value for x.

As an example, for $\Pi = $ MAX-CLIQUE, x is some graph G, $S(G)$ is the set of all cliques of G, c gives the size of the clique, the problem is a maximization problem, and we can give a single vertex as a feasible solution to the problem.

Definition 2.2 (Problem size). We will assume that the encodings of instances are *reasonable* in the sense described in Garey and Johnson []. More often than not, instead of the actual instance size $|x|$, we use a parameter n within a polynomial of $|x|$ to describe the size. For instance, we often use the number n of vertices to denote the size of a graph, even though in most cases the encoding requires $O(n^2)$ bits. In most cases, n is the size of the underlying set describing the instance x.

Definition 2.3 (Approximation algorithm). An *approximation algorithm* A is a polynomial time algorithm that computes the objective value of some feasible solution to an instance x of Π. Let $A(x)$ denote the value that A returns.

Definition 2.4 (Approximation ratio). An approximation algorithm A for Π has **approximation ratio** $\mu_A(n)$ if, for all instances x of Π,

$$\frac{\Pi(x)}{\mu_A(n)} \leq A(x) \leq \Pi(x), \quad \text{for maximization problems,}$$

and

$$\Pi(x) \leq A(x) \leq \mu_A(n)\Pi(x), \quad \text{for minimization problems.}$$

3 Probabilistically checkable proofs

A language L is in $\mathbf{NP}$ if there is a Turing machine which can verify in polynomial time if a string x is in L, given that the Turing machine is provided with a good "witness string" w. Hence, $\mathbf{NP}$ can be viewed as the class of languages which have an interactive proof system in which there is a polynomial time verifier V (Turing machine) and an infinitely powerful prover P (an oracle) who, on in put x, provides V with a proof π to convince V that x belongs to L. Henceforth, verifiers implicitly are polynomial time Turing machines. If x is in L, then there will be a proof π for which V accepts. This means that the proof system is *complete*. When x is not in L, the prover will not be able to fool V into accepting x. This means that the proof system is *sound*.

A good example is a proof system for the satisfiability problem. Given a boolean formula ϕ, the prover can give V a truth assignment a. Then, V only needs to verify that ϕ is evaluated to be TRUE under a. In this scheme, V has to read the entire proof to be convinced that ϕ is satisfiable. If verifiers are only allowed to read a small piece of the proof, is it possible to design a proof system so that we can be fairly confident (with a certain threshold on error probability) that the verification is correct?

To be more precise, by "fairly confident" we mean that, when x is in L the verifier accepts with high probability (completeness); and if x is not in L, then the probability that the verifier accepts x is upperbounded by some small soundness probability $s < 1$. The probability is taken over the random choices V made, assuming there is some internal random source. Hence, V now becomes a *probabilistic Turing machine.*

These ideas can be formalized with a *probabilistically checkable proof* (PCP) system. A *proof* is a binary string. An (r,q)*-restricted verifier* V is a probabilistic polynomial time verifier which uses at most r random bits and query the proof in at most q places (q bits). Both r and q are functions of the input size.

Given a proof π, a random string R, and an input x, let

$$V^\pi(x;R) = \begin{cases} 1 & \text{if } V \text{ accepts } x \\ 0 & \text{if } V \text{ rejects } x. \end{cases}$$

Define the acceptance probability of V given the proof π as follows.

$$\text{ACC}[V^\pi(x)] := \underset{R}{\text{Prob}}[V^\pi(x;R) = 1].$$

The maximum acceptance probability is an important measure, which is defined as

$$\text{ACC}[V(x)] := \max_\pi\{\text{ACC}[V^\pi(x)]\}.$$

Let $c, s : \mathbb{N}^+ \to [0,1]$ such that $0 < s(n) < c(n) \leq 1,\ \forall n \in \mathbb{N}^+$. The class $\mathbf{PCP}_{c,s}[q,r]$ consists of all languages L, for each of which there is an (r,q)-restricted verifier V which satisfies the following conditions:

- **Completeness.** If $x \in L$, then there is some proof π such that V accepts x with probability at least $c(|x|)$. In other words,
$$\text{ACC}[V(x)] \geq c(|x|), \forall x \in L.$$
- **Soundness.** If $x \notin L$, then V accepts x with probability less than $s(|x|)$, no matter what the proof is. In other words,
$$\text{ACC}[V(x)] < s(|x|), \forall x \notin L.$$

For the sake of brevity, we write $\mathbf{PCP}[r,q]$ instead of $\mathbf{PCP}_{1,\frac{1}{2}}[r,q]$. There are important parameters of a PCP system other than the randomness r and the query complexity q. We will define them later as needed.

Exercise 1. Let $\text{poly}(n) = \bigcup_{k\geq 0} O(n^k)$. Prove that

$$\mathbf{NP} = \mathbf{PCP}[0, \text{poly}(n)] \quad (1)$$
$$\mathbf{NP} = \mathbf{PCP}[O(\log n), \text{poly}(n)] \quad (2)$$

Exercise 2. Prove that

$$\mathbf{PCP}[O(\log n), O(1)] = \mathbf{PCP}[O(\log n), \text{poly}(n)]. \quad (3)$$

Exercise 3. The class $\mathbf{RP}$ consists of all languages L which have a polynomial time randomized algorithm A satisfying the following properties:

- if $x \in L$, then A accepts x with probability at least $1/2$,
- if $x \notin L$, then A does not accept x.

The class co-$\mathbf{RP}$ consists of all languages L for which $\overline{L} \in \mathbf{RP}$. Prove that

$$\mathbf{PCP}[\text{poly}(n), 0] = \text{co-}\mathbf{RP}. \tag{4}$$

One of the major results in theoretical computer science of the past two decades is the PCP theorem, which gives a robust characterization of the class $\mathbf{NP}$.

Theorem 3.1 (PCP Theorem). $\mathbf{NP} = \mathbf{PCP}[O(\log n), O(1)]$.

This theorem helps us devise gap-producing reductions as alluded to in Section 1. We illustrate this point via several inapproximability results presented in the next section.

4 Direct reductions from PCP

4.1 Satisfiability problems

Our first interesting result is that MAX-E3SAT is a MAXSNP problem. (MAXSNP is the class of problems which have a constant ratio approximation algorithm, but no approximation scheme unless P=NP.) In an earlier lecture, we have shown that there is a $8/7$-approximation algorithm for MAX-E3SAT. Thus, it remains to show that the problem cannot be approximated to within some constant $\rho > 1$, unless $\mathbf{P} = \mathbf{NP}$.

Theorem 4.1. *There is some constant $\rho_1 > 1$ such that it is $\mathbf{NP}$-hard to approximate* MAX-E3SAT *to within ρ_1.*

Proof. The general strategy was outlined in Section 1. Consider any $\mathbf{NP}$-complete language L. The reduction works by constructing in polynomial time a CNF formula φ_x with m clauses, given an input x. The formula φ_x satisfies the following properties.

$$\begin{aligned} x \in L &\Rightarrow \text{MAX-E3SAT}(\varphi_x) = m, \\ x \notin L &\Rightarrow \text{MAX-E3SAT}(\varphi_x) < m/\rho_1. \end{aligned} \tag{5}$$

Obviously, if we could find such a reduction then MAX-E3SAT is not approximable to within ρ_1, establishing MAX-E3SAT membership in the class MaxSNP.

By the PCP theorem, is some (r, q)-restricted verifier V recognizing L, where $r = O(\lg n)$ and q is a fixed constant. We will use V to construct φ_x for each input string x. Note that, when V is adaptive the length of the proof does not need to be more than $2^r 2^q$. When V is non-adaptive, the corresponding bound is $q2^r$. In both cases, V only needs polynomial-size proofs. Let $p = 2^{r+q} \geq q2^r$ be the upperbound on proof sizes.

We construct φ_x as follows. Create p variables $x_1, \dots, x_p$, so that each truth assignment to these variables corresponds to a proof presented to V. For each random string R of length r, there are some combinations of the answers to V's queries that make V accept. We can model this fact by a CNF formula φ_R on $\{x_1, \dots, x_p\}$ such that $\varphi_R(\mathbf{x}) = \text{TRUE}$ iff V accepts the proof $\mathbf{x}$. The formula ψ_R can be constructed in polynomial time by simulating V on the random string R and generating all possible combinations of answers. Since q is a constant, there are only 2^q answer combinations.

For example, suppose the queries ask for bits $1,4,5,9$ of the proofs, and V rejects x if the answers are either $(1,0,0,0)$, $(0,1,0,1)$, or $(1,1,1,0)$, then we can assign

$$\varphi_R(\mathbf{x}) = (\bar{x}_1 \vee x_4 \vee x_5 \vee x_9) \wedge (x_1 \vee \bar{x}_4 \vee x_5 \vee \bar{x}_9) \wedge (\bar{x}_1 \vee \bar{x}_4 \vee \bar{x}_5 \vee x_9).$$

Each clause says that the corresponding combination does **not** hold. Thus, the conjunction of these clauses says that V accepts x.

By adding a few auxiliary variables, we can convert φ_R into E3-CNF form. This conversion is standard. For example, $(x_1+x_2+x_3+x_4+x_5)$ is equivalent to

$$(x_1+x_2+y_1)(\bar{y}_1+x_3+y_2)(\bar{y}_2+x_4+x_5).$$

We create at most q clauses of size 3 for each of the original 2^q clauses. The resulting 3-CNF formula φ_R has at most $q2^q$ clauses.

Finally, let $\varphi_x = \prod_R \varphi_R$, then φ_x itself can be constructed in polynomial time since there are only polynomially many random strings R. (This is why the randomness of $O(\log n)$ is crucial.) Let m be the total number of 3-CNF clauses of φ_x, then $m = r(|x|)q2^q = O(\log n)q2^q$.

- When $x \in L$, there is a proof π (a truth assignment) such that V always accepts. Hence, under this assignment ϕ_x is satisfiable.
- When $x \notin L$, set $\pi_i = x_i$ for all i and feed π as a proof to V. In this case, V only accepts with probability $< 1/2$. Hence, at least half of the φ_R are not satisfiable by any truth assignment. For each φ_R that is not satisfied, there is at least one clause that is not satisfied. The number of non-satisfied clauses is thus at least $\frac{1}{2}r(|x|)$. Consequently,

$$\text{MAX-E3SAT}(\varphi_x) < m - \frac{1}{2}r(|x|) = m(1-\frac{1}{2q2^q}).$$

Let $\rho_1 = \frac{q2^{q+1}}{q2^{q+1}-1}$ and the proof is completed. □

Exercise 4. Show that $\mathbf{PCP}[O(\log n), O(1)] \subseteq \mathbf{NP}$.

It is interesting to know that the converse of the above theorem is also true, i.e. one can theoretically prove the PCP theorem by showing a constant-ratio inapproximability result for MAX-E3SAT.

Theorem 4.2. *If there is a reduction satisfying* (5) *from an* $\mathbf{NP}$*-complete language* L*, then* $\mathbf{NP} = \mathbf{PCP}[O(\log n), O(1)]$.

Proof. By Exercise 4, $\mathbf{PCP}[O(\log n), O(1)] \subseteq \mathbf{NP}$. Conversely, to show that $\mathbf{NP} \subseteq \mathbf{PCP}[O(\log n), O(1)]$ we design an (r,q)-verifier V for L. This can be used as an (r,q)-verifier for any other language $L' \in \mathbf{NP}$, since L' reduces to L.

Consider any input string x. Use the assumed reduction to construct φ_x. The strategy for V is to pick a constant number k of clauses of φ_x at random, ask the prover for the values of (at most $3k$) variables in these clauses, and accept iff all the clauses are satisfied. Clearly V has perfect completeness. When $x \notin L$, more than $(1-1/\rho_1)m$ clauses are satisfied. The probability that V accepts is at most

$$\begin{aligned}
\text{ACC}[V(x)] &< \frac{\binom{(1-1/\rho_1)m}{k}}{\binom{m}{k}} \\
&= \frac{(m-m/\rho_1)(m-m/\rho_1-1)\dots(m-m/\rho_1-k+1)}{m(m-1)\dots(m-k+1)} \\
&< \left(\frac{m-m/\rho_1}{m}\right)^k \\
&= (1-1/\rho_1)^k.
\end{aligned}$$

When $k \geq \lg(\rho_1/(\rho_1 - 1))$, ACC$[V(x)] < (1 - 1/\rho_1)^k \leq 1/2$. Since $m = \text{poly}(|x|)$, the number of random bits V used is $O(\lg m) = O(\lg |x|)$, and the number of query bits needed is about $3 \lg(\rho_1/(\rho_1 - 1))$, which is a constant. $\square$

4.2 MAX CLIQUE and the FGLSS reduction

The *PCP connection* refers to the use of a PCP characterization of $\mathbf{NP}$ to show hardness results for optimization problems. This connection was first noticed via a reduction from interactive proofs to MAX-CLIQUE in the pioneering work of Feige, Goldwasser, Lovász, Safra, and Szegedy [15]. Since then, the reduction is referred to as the FGLSS reduction.

Consider an (r, q)-restricted verifier V for a language $L \in \mathbf{PCP}_{c,s}[q, r]$. On input x a *transcript* is a tuple $T = \langle R, Q_1, a_1, \dots, Q_q, a_q \rangle$ such that $|R| = r$ is a random string, the Q_i and a_i are the queries and corresponding answers that V made and received, in that order, given the random string. T is an *accepting transcript* if V accepts x after seeing the answers.

Two transcripts $T = \langle R, Q_1, a_1, \dots, Q_q, a_q \rangle$ and $T' = \langle R', Q'_1, a'_1, \dots, Q'_q, a'_q \rangle$ are *consistent* with each other if $Q_i = Q'_j \Rightarrow a_i = a'_j \; \forall i, j$, i.e. if for the same questions we get the same answers.

On an input x which V tries to verify if x is in L or not, we will construct a graph G_x in polynomial time such that

$$\begin{aligned} x \in L \quad &\Rightarrow \quad \text{MAX-CLIQUE}(G_x) \geq \frac{c}{2^q}|V_x| \\ x \notin L \quad &\Rightarrow \quad \text{MAX-CLIQUE}(G_x) < \frac{s}{2^q}|V_x|. \end{aligned}$$

Let $G_x = (V_x, E_x)$, where V_x represents all accepting transcripts of V on x and E_x consists of edges connecting consistent pairs of transcripts. It follows that $|V_x| \leq 2^{r+q}$. We can add dummy vertices so that $|V_x| = 2^{r+q}$.

Note that the first question V asks is deterministic, knowing w and R. Then, knowing the first answer the second question is known, etc. Thus, the questions in a transcript are in fact redundant for the encoding of transcripts. Consequently, the vertices of G_x with the same random string R form a cluster of independent vertices.

If $x \in L$, then there is some proof π such that ACC$[V^\pi(x)] \geq c$. Consider the set of all transcripts whose answers come from π, then all these transcripts are consistent with each other. In other words, they form a clique. The fact that ACC$[V^\pi(x)] \geq c$ implies that the clique size is at least $c2^r$. Hence,

$$\text{MAX-CLIQUE}(G_x) \geq c2^r = \frac{c}{2^q}|V_x|.$$

Conversely, from a clique of G_x of size k, say, we can construct a proof π for which V^π accepts with probability $k/2^r$. The proof is constructed by taking the union of the answers of the transcripts from the clique, adding dummy answers if they were not part of any transcript in the clique. Consequently, when $x \notin L$ there cannot be a clique of size $s2^r$ or more, otherwise there would be a proof π for which V^π accepts with probability at least s. Hence, in this case

$$\text{MAX-CLIQUE}(G_x) < s2^r = \frac{s}{2^q}|V_x|.$$

Theorem 4.3. *It is $\mathbf{NP}$-hard to approximate* MAX CLIQUE *to within any constant ratio.*

Proof. By the PCP theorem, we have a $(O(\lg n), O(1))$-restricted verifier V for any chosen $\mathbf{NP}$-hard language L. The reduction above implies that MAX CLIQUE cannot be approximated to within $\rho = c/s = 1/(1/2) = 2$. By repeating the verifier independently k times and accepting iff all k rounds accept the input, we can reduce V's soundness exponentially down to $1/2^k$. Given any constant ρ, set k to be slightly more than $\lg \rho$. Now the approximation ratio lowerbound is at least ρ. $\square$

4.3 Reductions using better verifiers

Håstad [25, 26] showed the following remarkable result.

Theorem 4.4 (Håstad). *For every $\epsilon > 0$,*

$$\mathbf{NP} = \mathbf{PCP}_{1-\epsilon, \frac{1}{2}+\epsilon}[O(\log n), 3].$$

Furthermore, there is a verifier for SAT which works as follows. It computes three query positions i, j, k and a bit b. Then, it accepts the input iff

$$\pi_i + \pi_j + \pi_k = b \pmod 2.$$

One can easily use his result to prove quite a few strong inapproximation results.

Exercise 5. The MAX EkLINEQ-2 problem is the problem of finding a solution which satisfies the most number of equations among the given m linear equations over $\mathbb{F}_2$, where each equation has exactly k variables. From Håstad's theorem, show that MAX E3LINEQ-2 cannot be approximated to within any constant less than 2, unless $\mathbf{P} = \mathbf{NP}$.

5 Hardness reductions among problems

5.1 Gap-preserving reductions

As we have seen from the previous section, to show that it is $\mathbf{NP}$-hard to approximate some maximization problem Π to within some ratio $\rho(n)$ we do the following. Start from some $\mathbf{NP}$-complete language L, find a polynomial time reduction from L to Π such that: given an input x the reduction constructs in polynomial time an instance I_x of Π satisfying

- $x \in L$ implies $\Pi(I_x) \geq t(I_x)$, and
- $x \notin L$ implies $\Pi(I_x) < t(I_x)/\rho(|I_x|)$,

where $t(I_x)$ is some threshold function on the instance I_x. Put it another way, *it is $\mathbf{NP}$-hard to distinguish between instances of Π whose optimal value is at least $t(I)$ and instances of Π whose optimal value is less than $t(I)/\rho(|I|)$.* For example, for the MAX CLIQUE problem $I_x = G_x$, $t(G_x) = |V_x|/2^q$, and $\rho(|G_x|) = 2$.

Similarly, if Π is a minimization problem the reduction should satisfy

- $x \in L$ implies $\Pi(I_x) \leq t(I_x)$, and
- $x \notin L$ implies $\Pi(I_x) > \rho(|I_x|)t(I_x)$.

Just the usual reductions among $\mathbf{NP}$-complete problems, we do not need to do reductions directly from some $\mathbf{PCP}$. The notion of *gap preserving* reductions plays the role of Karp reductions among $\mathbf{NP}$-complete problems. To define a gap preserving reduction from Π_1 to Π_2, we need to know if each of them is a maximization or minimization problem.

There are four cases which are completely similar. We will consider only one case here: Π_1 is a maximization and Π_2 is a minimization problem. A gap preserving reduction from Π_1 to Π_2 comes with four functions t_1, ρ_1, t_2, ρ_2. Given an instance I_1 of Π_1, the reduction constructs in polynomial time an instance I_2 of Π_2 such that

- if $\Pi_1(I_1) \geq t_1(I_1)$, then $\Pi_2(I_2) \leq t_2(I_2)$

- if $\Pi_1(I_1) < \frac{1}{\rho(|I_1|)} t_1(I_1)$, then $\Pi_2(I_2) > \rho(|I_2|) t_2(I_2)$

The following theorem is straightforward.

Theorem 5.1. *Suppose there is a reduction from Π_1 to Π_2 as described above. If it is $\mathbf{NP}$-hard to distinguish between instances of Π_1 with optimal value at least t_1 and instances of Π_1 with optimal value less than t_1/ρ_1, then it is $\mathbf{NP}$-hard to distinguish between instances of Π_2 with optimal value at most t_2 and instances of Π_2 with optimal value more than $\rho_2 t_2$; in particular, it is $\mathbf{NP}$-hard to approximate Π_2 to within ρ_2.*

Exercise 6. Suppose we know that MAX E3LINEQ-2 cannot be approximated to within $2-\epsilon$ for any $\epsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$. Give a gap-preserving reduction to show that MAX E3SAT cannot be approximated to within $8/7 - \epsilon$, for any $\epsilon > 0$.

Hint: the equation $x + y + z = 0 \pmod 2$ is equivalent to the conjunction of four 3CNF clauses

$$(\bar{x} \vee y \vee z) \wedge (x \vee \bar{y} \vee z) \wedge (x \vee y \vee \bar{z}) \wedge (\bar{x} \vee \bar{y} \vee \bar{z}).$$

5.2 MAX-3SAT with bounded occurrences

Definition 5.2 (Expanders). A *λ-expander* is a k-regular graph G for which k is a constant, and $|[S, \bar{S}]| \geq \lambda |S|$ for all subsets $S \subset V(G)$ where $|S| \leq |V(G)|/2$.

For our purposes, we will need the following fact: *there is an n_0 such that, for each $n \geq n_0$ a 1-expander with n vertices can be constructed in polynomial time.* (There is some technical discrepancies between the above statement and the actual result; however, the difference is neglectable.) The reader is referred to the recent survey [28] for more information on expander graphs and their various applications.

The problem MAX-3SAT-d is MAX-3SAT restricted to formulas where each variable appear (as itself or as its negation) in at most d clauses. For reasons to be clear later, we will need the result that MAX-3SAT-d cannot be approximated to a certain constant ratio.

Theorem 5.3. *There is a constant $\rho_2 > 1$ and a constant d such that it is $\mathbf{NP}$-hard to approximate* MAX-3SAT-d *to within ρ_2. More specifically, given an E3-CNF formula φ with m clauses, there is a polynomial time reduction which constructs a 3-CNF-d formula φ' with m' clauses such that*

- *If* MAX-E3SAT$(\varphi) = m$, *then* MAX-3SAT-$d(\varphi') = m'$*; and,*
- *if* MAX-E3SAT$(\varphi) < m/\rho_1$, *then* MAX-3SAT-$d(\varphi') < m'/\rho_2$.

Proof. Let $x_1, \ldots, x_n$ be the variables of φ. First, repeat each clause of φ n_0 times so that each variable appears in at least n_0 clauses. Call the resulting formula ψ, which has mn_0 clauses $C_1, C_2, \ldots, C_{mn_0}$.

Create one variable y_{ij} if x_i appears (as itself or as its negation) in clause C_j. Let ψ' be the formula obtained by replacing x_i in C_j by y_{ij}. For example, if $C_j = (x_1 + \bar{x}_3 + x_7)$, then the clause becomes $(y_{1j} + \bar{y}_{3j} + y_{7j})$ in ψ'. Thus, ψ' has mn_0 clauses and each variable y_{ij} appears only once. The clauses of ψ' are called *satisfiability clauses*.

Define $S_i = \{j \mid x_i \in C_j\}$. For each variable x_i that appears $n_i \geq n_0$ times in ψ, let G_i be a k-regular 1-expander with the set of vertices S_i. By the aforementioned fact, G_i can be constructed in polynomial time.

To this end, let ψ_i be the formula which is the conjunction of clauses of the form $(y_{ij} + \bar{y}_{ij'})(\bar{y}_{ij} + y_{ij'})$, for each edge (j, j') in the graph G_i. The key point to notice is that both $(y_{ij} + \bar{y}_{ij'})$ and $(\bar{y}_{ij} + y_{ij'})$ if and only if $y_{ij} = y_{ij'}$. Note that, each variable y_{ij} appears $2k$ times in ψ_i; and that there are $n_i k$ clauses in ψ_i. These clauses are called *consistency clauses*.

Finally, let

$$\varphi' = \psi' \wedge \bigwedge_{i=1}^{n} \psi_i.$$

Then, φ' has $m' = mn_0 + k\sum_i n_i = m(n_0 + 3k)$ clauses.
Claim 1. If MAX-E3SAT$(\varphi) = m$, then MAX-3SAT-$d(\varphi') = m'$.

The proof is immediate. Consider any truth assignment that satisfies φ. Assign $y_{ij} = x_i$ for all $j \in S_i$
Claim 2. If MAX-E3SAT$(\varphi) < m/\rho_1$, then MAX-3SAT-$d(\varphi') < m'/\rho_2$, where $\rho_2 = \frac{n_0+3k}{n_0/\rho_1+3k}$.

A truth assignment for φ' is *consistent* if, for each i, all y_{ij} are assigned with the same boolean value. To prove that claim, we first show the following crucial fact: there is an optimal truth assignment for φ' which is consistent.

Consider any optimal truth assignment for φ'. Suppose there is some i for which not all y_{ij} are assigned with the same boolean value. Let $S_t = \{j \mid y_{ij} = \text{TRUE}\}$ and $S_f = \{j \mid y_{ij} = \text{FALSE}\}$. Then $S_t \cup S_f$ is a partition of the vertices of G_i. WLOG, assume $|S_t| \leq |S_f|$. The number of edges (j, j') with $j \in S_f$ and $j' \in S_t$ is at least $|S_t|$. For each such edge (j, j'), exactly one of the two clauses $(y_{ij} + \bar{y}_{ij'})$ and $(\bar{y}_{ij} + y_{ij'})$ is satisfied by the current truth assignment. Now, invert the values of all y_{ij} for $j \in S_t$. Then, in the conjunction $\bigwedge_{i=1}^{n} \psi_i$ we gain at least $|S_t|$ more satisfied clauses. We might turn some clauses in ψ' from being satisfied to being unsatisfied, however. Fortunately, the number of such clauses is at most $|S_t|$. Consequently, the new truth assignment satisfies as many clauses as the previous one. Repeat this process for any i for which not all y_{ij} are assigned with the same boolean value. We finally end up with a consistent truth assignment for φ'.

Now, consider any optimal and consistent truth assignment for φ'. Set $x_i = y_{ij}$ for any $j \in S_i$. Then, the fraction of satisfied clauses of φ is equal to the fraction of satisfied clauses of ψ'. In other words, if an optimal truth assignment for φ cannot satisfy m/ρ_1 clauses, then an optimal truth assignment for φ' cannot satisfy

$$mn_0/\rho_1 + 3km = m'\frac{n_0/\rho_1 + 3k}{n_0 + 3k}$$

clauses of φ'. □

Exercise 7. Show that there are constants $\rho > 1$ and d such that it is **NP**-hard to approximate MAX-E3SAT-d to within ρ.

5.3 INDEPENDENT SET and VERTEX COVER on bounded degree graphs

The problem VERTEX COVER-d is the VERTEX COVER problem restricted to graphs with maximum degree at most d. Similarly we define INDEPENDENT SET-d.

Consider any 3-CNF formula φ with m clauses. By repeating some literal in any clause of size < 3, we can assume that the formula is E3-CNF. Consider the graph G_φ constructed as follows. For each clause $(l_a + l_b + l_c)$ in φ, there corresponds a triangle in the graph. Thus, the graph has $3m$ vertices. The edges of the triangles are called *clause edges*. Next, connect x_i and $\bar{x}_i$ for all occurrences of x_i and $\bar{x}_i$. These edges are called *consistency edges*. Since each variable appears in at most d clauses, for each vertex of G_φ there are 2 incident clause edges and at most $d - 1$ incident consistency edges. Thus, G_φ has maximum degree $d + 1$.

It is straightforward to see that

$$\begin{aligned}\text{MAX-3SAT-}d(\varphi) &= \text{INDEPENDENT SET-}(d+1)(G_\varphi)\\ \text{MAX-3SAT-}d(\varphi) &= 3m - \text{VERTEX COVER-}(d+1)(G_\varphi)\end{aligned}$$

Consequently,

- if MAX-3SAT-$d(\varphi) = m$, then INDEPENDENT SET-$(d+1)(G_\varphi) = m$ and VERTEX COVER-$(d+1)(G_\varphi) = 2m = \frac{2}{3}(3m)$.
- if MAX-3SAT-$d(\varphi) < m/\rho_2$, then INDEPENDENT SET-$(d+1)(G_\varphi) < m/\rho_2$ and VERTEX COVER-$(d+1)(G_\varphi) > (3 - 1/\rho_2)m = (9/2 - 3/(2\rho_2))\frac{2}{3}(3m)$.

The following theorems are immediate.

Theorem 5.4. *There is a constant d such that it is* $\mathbf{NP}$*-hard to approximate* INDEPENDENT SET-d *to within* ρ_2.

Theorem 5.5. *Let* $\rho_3 = (9/2 - 3/(2\rho_2))$*, then there is a constant d such that it is* $\mathbf{NP}$*-hard to approximate* VERTEX COVER-d *to within* ρ_3.

Exercise 8. Show that there are constants $\rho > 1$ and d such that it is $\mathbf{NP}$-hard to approximate the VERTEX COVER problem for *connected* graphs with maximum degree at most d.

5.4 STEINER TREE and METRIC STEINER TREE

In the STEINER TREE problem, we are given an edge-weighted graph G and a subset $S \subseteq V(G)$ of "terminals." The problem is to find a minimum weight subtree of G which spans the terminals. Note that the tree may contain non-terminal vertices, which are also commonly referred to as Steiner vertices. In the METRIC STEINER TREE problem, G is a complete graph and the edge weights satisfy the triangle inequality.

Exercise 9. Prove that STEINER TREE can be approximated to within ρ iff METRIC STEINER TREE can be approximated to within ρ.

Now, we show an inapproximability result for METRIC STEINER TREE with a reduction from bounded degree VERTEX COVER. Let $G = (V, E)$ be a *connected* graph with maximum degree at most d, where $|V| = n$ and $|E| = m$. Let H be the complete graph on $n + m$ vertices labeled by $V \cup E$. To avoid confusion, we use $[v]$ and $[u, v]$ to denote vertices of H, for all $v \in V$ and $(u, v) \in E$. The edge weights of H are defined as follows. For any two vertices $u, v \in V$, $([u], [v])$ has weight 1. For any edge $(u, v) \in E$, $([u], [u, v])$ and $([v], [u, v])$ has weight 1. All other edges have weights 2. Elements of E are the terminals. Clearly, H is an instance of METRIC STEINER TREE.

Lemma 5.6. *G has a vertex cover of size k iff H has a Steiner tree of cost $m + k - 1$.*

Proof. Necessity is obvious. For sufficiency, suppose H has a Steiner tree T of cost $m + k - 1$. We first show that there is another Steiner tree of the same cost which uses only edges of weight 1.

- If there is an edge $([u], [v, w])$ of weight 2, replace it by two edges $([u], [v])$ and $([v], [v, w])$.
- If there is an edge $([u, v], [v, w])$ of weight 2, replace it by $([u, v], [v])$ and $([v], [v, w])$.
- The last case is the most interesting, which is when there is an edge $([u, v], [x, y])$ of weight 2. If we remove this edge from T, then the set E of vertices of H is partitioned into two parts E_1 and E_2 which are contained in two connected components of T. Note that $[u, v]$ belongs to one connected component and $[x, y]$ belongs to the other. Since G is connected, there are two edges in E_1 and E_2 which share a vertex. Denote these two edges by (r, s) and (s, t). Now, add back to G two edges $([r, s], [s])$ and $([s], [s, t])$, we get a connected subgraph of G with the same cost as the original Steiner tree.

Repeat the above steps until T no longer contains a weight-2 edge. The Steiner vertices in T form a vertex cover of G with size at most k. □

Theorem 5.7. *There is a constant* $\rho_3 > 1$ *such that it is* $\mathbf{NP}$*-hard to approximate* METRIC STEINER TREE *to within* ρ_3.

Proof. By the previous lemma, if G has a vertex cover of size at most $2n/3$, then there is a Steiner tree for H of cost at most $m + 2n/3 - 1$.

Conversely, suppose G has no vertex cover of size $\rho_2 \cdot 2n/3$, then H has no Steiner tree of cost $m + \rho_2 \cdot 2n/3 - 1$. We want a ρ_3 for which

$$m + \rho_2 \cdot 2n/3 - 1 \geq \rho_3(m + 2n/3 - 1),$$

which is equivalent to

$$m \leq \frac{2(\rho_2 - \rho_3)}{3(\rho_3 - 1)} n + 1.$$

Since G has maximum degree at most d, we know $m \leq \frac{d}{2} n$. Thus, any ρ_3 for which

$$\frac{d}{2} \leq \frac{2(\rho_2 - \rho_3)}{3(\rho_3 - 1)}$$

would be sufficient. Elementary computations confirm that

$$\rho_3 = \frac{d + 4\rho_2/3}{d + 4/3}$$

works. Note that $\rho_3 > 1$ as desired. □

Historical Notes

The reader is referred to excellent recent surveys by Feige [14], Trevisan [36], and Johnson [] for the topic of hardness of approximation. See also a recent paper on Dinur's proof of the $\mathbf{PCP}$ theorem [] and Wigderson's report at ICM 2006 [].

The idea of checking proofs probabilistically dates back to the seminal papers on *interactive proofs* (IP) by Goldwasser, Micali and Rackoff [21], and Babai [6]. Later development led to the notion of *multi-prover interactive proof systems* (MIP) introduced by Ben-Or, Goldwasser, Kilian, and Wigderson [10]. MIPs in turn led to the idea of a probabilistic verifier with oracle access (PCP verifier) originated in the paper by Fortnow, Rompel and Sipser [17].

Restrictions on resources of the proof system, of the verification process in particular, led to other interesting concepts and ideas. The first of such papers was that by Babai, Fortnow, Levin and Szegedy [7], who gave attention to the computation time of the verifier and the proof size.

The idea that a prover can convince someone the validity of a statement without that someone reading the entire proof has numerous practical and theoretical implications and applications, especially in the area of cryptography and security. Zero-knowledge proofs [21], for instance, is a perfect example of this theme. The reader is referred to [20] for a sample discussion of these ideas. Lovász [31] gave an introduction with interesting every-day's examples on this topic.

In 1990, the landmark work by Feige, Goldwasser, Lovász, Safra, and Szegedy [15] connects probabilistic proof systems and the inapproximability of $\mathbf{NP}$-hard problems. This has become known as **the** PCP connection. A year later, the PCP theorem - a very strong characterization of $\mathbf{NP}$ - was proved with the works of Arora and Safra [4], Arora, Lund, Motwani, Sudan, and Szegedy [3]. A plethora of inapproximability results using the PCP connection follow, some of them are optimal [2, 12, 13, 16, 22, 24, 25, 32].

Following Madhu Sudan's lectures at ISA, we can summarize the results on PCP into several phases as follows.

Phase 0:

$$\mathbf{NP} = \mathbf{PCP}[0, \text{poly}]$$
$$\mathbf{NP} = \mathbf{PCP}[\log, \text{poly}]$$

Phase 1:

$$\begin{aligned} \text{NEXP} &= \mathbf{PCP}[\text{poly}, \text{poly}] \text{ [8, 9]} \\ \mathbf{NP} &\subseteq \mathbf{PCP}[\text{poly}(\log), \text{poly}(\log)] \text{ [7]} \end{aligned}$$

Phase 2:

$$\begin{aligned} \mathbf{NP} &= \mathbf{PCP}[O(\log), o(\log)] \text{ [4]} \\ \mathbf{NP} &= \mathbf{PCP}[O(\log), O(1)], \text{ [3]} - \textbf{The PCP Theorem} \end{aligned}$$

Phase 3:

$$\begin{aligned} \text{SAT} &\in \mathbf{PCP}[(1+\epsilon)\ \log, O(1)],\ \forall \epsilon > 0 \quad \text{[35]} \\ \mathbf{NP} &= \mathbf{PCP}_{1-\epsilon, 1/2+\epsilon}[O(\log), 3],\ \forall \epsilon > 0 \quad \text{[25]} \\ \mathbf{NP} &= \mathbf{PCP}_{1, 1/2+\epsilon}[O(\log), 3],\ \forall \epsilon > 0 \quad \text{[23]} \\ \text{P} &= \mathbf{PCP}_{1, 1/2}[O(\log), 3] \quad \text{[30]} \end{aligned}$$

The results in phase 3 are pretty amazing. In a sense, P and NP are only ϵ apart, and thus the NP characterizations in [23, 25] are optimal! It is also interesting to note that the last result by Karloff and Zwick [30] made use of *semi-definite programming* methods introduced by Goemans and Williamson [19] to design approximation algorithms for the MAX-CUT and satisfiability problems.

The class MAXSNP was formulated in Papadimitriou and Yannakakis [34]. The $8/7$-approximation algorithm for MAX-E3SAT follows the line of Yannakakis [38], who gave the first $4/3$-approximation for MAX-SAT. A 2-approximation for MAX-SAT was given in the seminal early work of Johnson [29]. Johnson's algorithm can also be interpreted as a derandomized algorithm, mostly the same as the one we presented. Details on derandomization can be found in the standard texts by Alon and Spencer [1], and Motwani and Raghavan [33]. Later, Karloff and Zwick [30] gave an $8/7$-approximation algorithm for MAX-3SAT based on semidefinite programming. This approximation ratio is optimal as shown by Håstad [25, 26]. In fact, by designing special purpose PCPs for different problems, Håstad was able to obtain optimal non-approximability results for MAX-EkSAT for all $k \geq 3$, MAX LINEQ, and SET SPLITTING.

References

[1] N. ALON AND J. H. SPENCER, *The probabilistic method*, Wiley-Interscience Series in Discrete Mathematics and Optimization, Wiley-Interscience [John Wiley & Sons], New York, second ed., 2000. With an appendix on the life and work of Paul Erdős.

[2] S. ARORA, L. BABAI, J. STERN, AND Z. SWEEDYK, *The hardness of approximate optima in lattices, codes, and systems of linear equations*, J. Comput. System Sci., 54 (1997), pp. 317–331. 34th Annual Symposium on Foundations of Computer Science (Palo Alto, CA, 1993).

[3] S. ARORA, C. LUND, R. MOTWANI, M. SUDAN, AND M. SZEGEDY, *Proof verification and the hardness of approximation problems*, J. ACM, 45 (1998), pp. 501–555. Prelim. version in FOCS'92.

[4] S. Arora and S. Safra, *Probabilistic checking of proofs: a new characterization of NP*, J. ACM, 45 (1998), pp. 70–122. Prelim. version in FOCS'92.

[5] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi, *Complexity and approximation*, Springer-Verlag, Berlin, 1999. Combinatorial optimization problems and their approximability properties, With 1 CD-ROM (Windows and UNIX).

[6] L. Babai, *Trading group theory for randomness*, in STOC '85 (Providence, Rhode Island), ACM, New York, 1985, pp. 421–429.

[7] L. Babai, L. Fortnow, L. A. Levin, and M. Szegedy, *Checking computations in polylogarithmic time*, in Proceedings of the twenty-third annual ACM symposium on Theory of computing, ACM Press, 1991, pp. 21–32.

[8] L. Babai, L. Fortnow, and C. Lund, *Nondeterministic exponential time has two-prover interactive protocols*, Comput. Complexity, 1 (1991), pp. 3–40.

[9] L. Babai, L. Fortnow, and C. Lund, *Addendum to: "Nondeterministic exponential time has two-prover interactive protocols" [Comput. Complexity* **1** *(1991), no. 1, 3–40]*, Comput. Complexity, 2 (1992), p. 374.

[10] M. Ben-Or, S. Goldwasser, J. Kilian, and A. Wigderson, *Multiprover interactive proofs: How to remove intractability assumptions*, in STOC '88 (Chicago, Illinois), ACM, New York, 1988, pp. 113–131.

[11] P. Crescenzi and V. K. (eds.), *A compendium of NP-optimization problems*. http://www.nada.kth.se/

[12] I. Dinur and S. Safra, *On the hardness of approximating label-cover*, Inform. Process. Lett., 89 (2004), pp. 247–254.

[13] U. Feige, *A threshold of* $\ln n$ *for approximating set cover (preliminary version)*, in Proceedings of the Twenty-eighth Annual ACM Symposium on the Theory of Computing (Philadelphia, PA, 1996), New York, 1996, ACM, pp. 314–318.

[14] ———, *Approximation thresholds for combinatorial optimization problems*, in Proceedings of the International Congress of Mathematicians, Vol. III (Beijing, 2002), Beijing, 2002, Higher Ed. Press, pp. 649–658.

[15] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy, *Interactive proofs and the hardness of approximating cliques*, J. ACM, 43 (1996), pp. 268–292. Prelim. version in FOCS'91.

[16] U. Feige and J. Kilian, *Zero knowledge and the chromatic number*, J. Comput. System Sci., 57 (1998), pp. 187–199. Complexity 96—The Eleventh Annual IEEE Conference on Computational Complexity (Philadelphia, PA).

[17] L. Fortnow, J. Rompel, and M. Sipser, *On the power of multi-prover interactive protocols*, Theoret. Comput. Sci., 134 (1994), pp. 545–557.

[18] M. R. Garey and D. S. Johnson, *Computers and intractability*, W. H. Freeman and Co., San Francisco, Calif., 1979. A guide to the theory of NP-completeness, A Series of Books in the Mathematical Sciences.

[19] M. X. Goemans and D. P. Williamson, *Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming*, J. Assoc. Comput. Mach., 42 (1995), pp. 1115–1145.

[20] O. Goldreich, *Modern cryptography, probabilistic proofs and pseudorandomness*, vol. 17 of Algorithms and Combinatorics, Springer-Verlag, Berlin, 1999.

[21] S. Goldwasser, S. Micali, and C. Rackoff, *The knowledge complexity of interactive proof systems*, SIAM J. Comput., 18 (1989), pp. 186–208.

[22] V. Guruswami, J. Håstad, and M. Sudan, *Hardness of approximate hypergraph coloring*, SIAM J. Comput., 31 (2002), pp. 1663–1686 (electronic).

[23] V. Guruswami, D. Lewin, M. Sudan, and L. Trevisan, *A tight characterization of NP with 3 query PCPs*, in IEEE Symposium on Foundations of Computer Science, 1998, pp. 8–17.

[24] J. Håstad, *Clique is hard to approximate within* $n^{1-\epsilon}$, Acta Math., 182 (1999), pp. 105–142.

[25] ———, *Some optimal inapproximability results*, in STOC '97 (El Paso, TX), ACM, New York, 1999, pp. 1–10 (electronic).

[26] ———, *Some optimal inapproximability results*, J. ACM, 48 (2001), pp. 798–859 (electronic).

[27] D. S. HOCHBAUM, ed., *Approximation Algorithms for NP Hard Problems*, PWS Publishing Company, Boston, MA, 1997.

[28] S. HOORY, N. LINIAL, AND A. WIGDERSON, *Expander graphs and their applications*, Bull. Amer. Math. Soc. (N.S.), 43 (2006), pp. 439–561 (electronic).

[29] D. S. JOHNSON, *Approximation algorithms for combinatorial problems*, J. Comput. System Sci., 9 (1974), pp. 256–278. Fifth Annual ACM Symposium on the Theory of Computing (Austin, Tex., 1973).

[30] H. KARLOFF AND U. ZWICK, *A 7/8-approximation algorithm for MAX 3SAT?*, in Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science, Miami Beach, FL, USA, IEEE Press, 1997.

[31] L. LOVÁSZ, *Interactive proofs: a new look at passwords, proofs, and refereeing*. http://research.microsoft.com/users/lovasz/popular.htm.

[32] C. LUND AND M. YANNAKAKIS, *On the hardness of approximating minimization problems*, J. Assoc. Comput. Mach., 41 (1994), pp. 960–981.

[33] R. MOTWANI AND P. RAGHAVAN, *Randomized algorithms*, Cambridge University Press, Cambridge, 1995.

[34] C. H. PAPADIMITRIOU AND M. YANNAKAKIS, *Optimization, approximation, and complexity classes*, J. Comput. System Sci., 43 (1991), pp. 425–440.

[35] A. POLISHCHUK AND D. A. SPIELMAN, *Nearly-linear size holographic proofs*, in STOC ’94 (Montral, Qubec, Canada), ACM, New York, 1994, pp. 194–203.

[36] L. TREVISAN, *Inapproximability of combinatorial optimization problems*, Tech. Rep. 65, The Electronic Colloquium in Computational Complexity, 2004.

[37] V. V. VAZIRANI, *Approximation algorithms*, Springer-Verlag, Berlin, 2001.

[38] M. YANNAKAKIS, *On the approximation of maximum satisfiability*, J. Algorithms, 17 (1994), pp. 475–502. Third Annual ACM-SIAM Symposium on Discrete Algorithms (Orlando, FL, 1992).