PCP & Hardness of Approximation

Vasilis Margonis

Advanced topics in Algorithms

April 25, 2018

Outline

1. Introduction
2. The **PCP** Theorem, a new characterization of **NP**
3. The Hardness of Approximation View
4. An optimal inapproximability result for MAX-3SAT
5. Inapproximability results for other known problems
6. Appendix: Derandomization via Conditional Expectations

Outline

1. Introduction
2. The **PCP** Theorem, a new characterization of **NP**
3. The Hardness of Approximation View
4. An optimal inapproximability result for $\textsc{Max-3SAT}$
5. Inapproximability results for other known problems
6. Appendix: Derandomization via Conditional Expectations

Introduction

- Suppose a mathematician circulates a proof of an important result, say Riemann Hypothesis, fitting 10 thousand pages.
- To verify it would take us several years, going through all of those pages.
- **Weird question**: Can we do better than that? (e.g. ignore most part of the proof)
- **Even weirder answer**: Yes, according to the **PCP** theorem.

The idea behind **PCP**

So, the mathematician can rewrite his proof in a certain format. the **PCP** format, so we can verify it by probabilistically selecting a constant number of bits to examine it. Furthermore, this verification has the following properties:

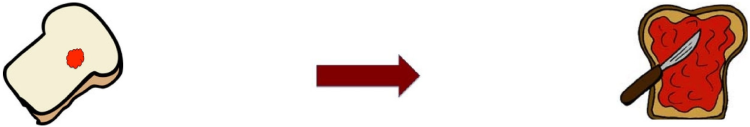
1. A correct proof will always convince us.
2. A false proof will convince us with only negligible probability (2^{-100} if we examine 300 bits).

The idea behind **PCP**

- In general, a mathematical proof is invalid if it has even a single error somewhere, which can be very difficult to detect.
- What the **PCP** theorem tells us is that there is a mechanical way to rewrite the proof so that the error is almost everywhere!

A nice analogue is the following:

Initial Proof **PCP transformation** **PCP format**



Outline

Standard definitions of **NP**

From now on, we shall refer to languages $L \subseteq \{0,1\}^*$.

Definition (Classic definition)

$$\mathbf{NP} = \bigcup_{c \in \mathbb{N}} \mathbf{NTIME}(n^c)$$

Definition (YES-certificate definition)

A language L is in **NP** if there exists a polynomial p and a polynomial-time TM V (called *verifier*) such that, given an input x, verifies certificates (proofs), denoted π:

$$x \in L \Rightarrow \exists \pi \in \{0,1\}^{p(|x|)} : V(x,\pi) = 1$$
$$x \notin L \Rightarrow \forall \pi \in \{0,1\}^{p(|x|)} : V(x,\pi) = 0$$

If $V(x,\pi) = 1$, then we call π a *correct proof* for x.

Towards a new characterization of **NP**

Definition (**PCP** verifier)

Let L be a language and $r, q : \mathbb{N} \to \mathbb{N}$. We say that L has an $[r(n), q(n)]$-**PCP** verifier if there is a polynomial-time TM V such that:

On input $x \in \{0,1\}^n$ and a string $\pi \in \{0,1\}^*$, V uses at most $r(n)$ random coins and makes at most $q(n)$ non-adaptive queries to locations of π, satisfying

- **Completeness**: $x \in L \Rightarrow \exists \pi \in \{0,1\}^* : \Pr[V(x,\pi) = 1] = 1$.
- **Soundness**: $x \notin L \Rightarrow \forall \pi \in \{0,1\}^* : \Pr[V(x,\pi) = 1] \leq \frac{1}{2}$.

We say that $L \in \mathbf{PCP}[r(n), q(n)]$, if L has a $[r(n), q(n)]$-**PCP** verifier.

Towards a new characterization of **NP**

Notes:

1. Proofs checkable by an $[r, q]$-**PCP** verifier are of length at most $q2^r$. The verifier looks at only q bits of the proof for any particular choice of its random coins, and there are only 2^r such choices.

2. The constant $1/2$ in the soundness condition is arbitrary, in the sense that we can execute the verifier multiple times to make the constant as small as we want.

The **PCP** theorem

By the definitions of **P** and **NP**:

- $\mathbf{P} = \mathbf{PCP}[0, 0]$
- $\mathbf{NP} = \mathbf{PCP}[0, poly(n)]$

Surprisingly...

Theorem (Arora, Safra, Lund, Motwani, Sudan, Szegedy)

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

Proof of the PCP theorem - easy direction

Lemma

$$\mathbf{PCP}[O(\log n), O(1)] \subseteq \mathbf{NP}$$

Proof.

An $[O(\log n), O(1)]$-**PCP** verifier can check proofs of length at most

$$2^{O(\log n)} O(1) = O(n^c).$$

Hence, a nondeterministic machine could "guess" the proof in $O(n^c)$ time, and verify it deterministically by running the verifier for all $2^{O(\log n)} = n^c$ possible outcomes of its random coin tosses. If the verifier accepts for all these possible coin tosses then the nondeterministic machine accepts.

Let $p(n)$ be the running time of the verifier. Then,

$$\mathbf{PCP}[O(\log n), O(1)] \subseteq \mathbf{NTIME}[O(n^c) + n^c \cdot p(n)] \subseteq \mathbf{NP}. \qquad \square$$

Proof of the **PCP** theorem - hard direction

Lemma

$$\mathbf{NP} \subseteq \mathbf{PCP}[O(\log n), O(1)]$$

The original proof is very extensive and outside the scope of this presentation. However, Irit Dinur gave a significantly simpler (but still hard) proof in 2007.

Outline

1. Introduction
2. The **PCP** Theorem, a new characterization of **NP**
3. The Hardness of Approximation View
4. An optimal inapproximability result for MAX-3SAT
5. Inapproximability results for other known problems
6. Appendix: Derandomization via Conditional Expectations

Motivation: Approximate solutions to **NP**-hard problems

- Since the discovery of **NP**-completeness in 1972, researchers tried to efficiently compute good approximate solutions to **NP**-hard optimization problems.
- After failing to design good approximation algorithms for some problems, they tried to give inapproximability results, but this effort also stalled.
- Researchers slowly began to realize that classic Cook/Karp style reductions do not suffice for proving limits on approximation algorithms. (apart from few isolated successes)
- The **PCP** Theorem, not only gave a new characterization of **NP**, but also provided a new type of reductions suitable for proving hardness of approximation, the *gap-producing* reductions.

Case Study: Max-3Sat

Input: A 3CNF formula ϕ, with n variables and m clauses.

$$\text{e.g.} \qquad \phi = (x_1 \vee \bar{x_2} \vee x_4) \wedge \ldots \wedge (x_2 \vee \bar{x_3} \vee \bar{x_n})$$

Goal: Find an assignment that satisfies as many clauses as possible.

Definition

- $\mathrm{val}(\phi)$ denotes the maximum *fraction* of clauses that can satisfied by any assignment. For example, ϕ is satisfiable iff $\mathrm{val}(\phi) = 1$.
- Let $\rho < 1$. An algorithm A is an ρ-approximation algorithm for MAX-3SAT if for every 3CNF formula ϕ with m clauses,

$$\mathrm{SOL}(A) \geq \rho \cdot \underbrace{\mathrm{val}(\phi) \cdot m}_{\mathrm{OPT}}.$$

A simple randomized algorithm for MAX-3SAT

Algorithm

For every variable x_i, set $x_i = 1$ with probability $\frac{1}{2}$, independently.

Claim

This is a $\frac{7}{8}$-approximation algorithm (in expectation).

Proof: We define the following random variable for every clause C_j

$$Y_j = \begin{cases} 1, & \text{clause } j \text{ is satisfied} \\ 0, & \text{otherwise} \end{cases}$$

Then, the number of clauses satisfied by the algorithm is

$$\mathrm{SOL} = \sum_{j=1}^{m} Y_j$$

A simple randomized algorithm for MAX-3SAT

- For every clause C_j

$$\mathbf{Pr}[Y_j = 1] = 1 - \left(\frac{1}{2}\right)^3 = \frac{7}{8}.$$

- Hence,

$$\mathbf{E}[\mathrm{SOL}] = \mathbf{E}\left[\sum_{j=1}^{m} Y_j\right] = \sum_{j=1}^{m} \mathbf{Pr}[Y_j = 1] = \sum_{j=1}^{m} \frac{7}{8} = \frac{7}{8} m \geq \frac{7}{8} \mathrm{OPT}. \quad \square$$

Remark

This algorithm can be derandomized via the method of conditional expectations. (See appendix section)

The hardness of approximation view

- Any hope for a PTAS or an FPTAS?
- The **PCP** Theorem implies that the answer is NO (unless $\mathbf{P} = \mathbf{NP}$). The reason is that it is *equivalent* to the following theorem.

Theorem (Gap-producing reduction)

There exists $\rho < 1$ such that $\forall L \in \mathbf{NP}$ there is a polynomial-time function f mapping strings to $3\mathrm{CNF}$ formulas such that:

$$x \in L \Rightarrow \mathrm{val}(f(x)) = 1 \tag{1}$$

$$x \notin L \Rightarrow \mathrm{val}(f(x)) < \rho \tag{2}$$

$\textsc{Max-3SAT}$ is **APX**-hard

Corollary

There exists some constant $\rho < 1$ such that there is no polynomial-time ρ-approximation algorithm for $\textsc{Max-3SAT}$, unless **P** $=$ **NP**.

- Indeed, we can convert a ρ-approximation algorithm A for $\textsc{Max-3SAT}$ into an algorithm deciding L.
- We apply the reduction f on x and then run the approximation algorithm to the resultant $3\mathrm{CNF}$ formula $f(x)$.
- (1) and (2) together imply that $x \in L$ iff $A(f(x))$ returns an assignment that satisfies at least a ρ fraction of $f(x)$'s clauses.

Outline

Recap

- There is a deterministic $\frac{7}{8}$-approximation algorithm.
- There exists a constant $\rho < 1$ such that there is no ρ-approximation unless $\mathbf{P} = \mathbf{NP}$.
- No hope for a PTAS.

Question: Can we do better than $7/8$?

More important question: what is the actual value of ρ?

Håstad answered both...

Only 3 bits ?

Theorem (Håstad, 1997)

$$\mathbf{NP} = \mathbf{PCP}_{1-\varepsilon,\frac{1}{2}+\varepsilon}[O(\log n), 3], \ \forall \varepsilon > 0$$

Moreover, the tests used by V *are linear: Given a proof* $\pi \in \{0,1\}^m$, V *chooses a random triple* (i,j,k) *and a bit* $b^{ijk} \in \{0,1\}$ *according to some distribution and accepts iff* $\pi_i \oplus \pi_j \oplus \pi_k = b^{ijk}$.

- ***Completeness***:

$$x \in L \Rightarrow \exists \pi \in \{0,1\}^m : \Pr_{(i,j,k)\in[m]^3}\left[\pi_i \oplus \pi_j \oplus \pi_k = b^{ijk}\right] \geq 1 - \varepsilon$$

- ***Soundness***:

$$x \notin L \Rightarrow \forall \pi \in \{0,1\}^m : \Pr_{(i,j,k)\in[m]^3}\left[\pi_i \oplus \pi_j \oplus \pi_k = b^{ijk}\right] \leq \frac{1}{2} + \varepsilon$$

3-bit **PCP** and $\textsc{Max-E3Lin}$

- We can convert the computation of Håstad's 3-bit **PCP** into an instance of a problem called $\textsc{Max-E3Lin}$, as follows:

$$\pi_1 \oplus \pi_1 \oplus \pi_1 = b^{111}$$
$$\pi_1 \oplus \pi_1 \oplus \pi_2 = b^{112}$$
$$\vdots$$
$$\pi_m \oplus \pi_m \oplus \pi_m = b^{mmm}$$

- If $x \in L$, $\exists \pi = (\pi_1, \ldots, \pi_m)$ that satisfies all constraints.
- If $x \notin L$, $\forall \pi = (\pi_1, \ldots, \pi_m)$ at most a $(1/2 + \varepsilon)$ fraction of constraints can be satisfied.

Corollary

Håstad's Theorem implies that there is no $(1/2 + \varepsilon)$-approximation for $\textsc{Max-E3Lin}$, for every $\varepsilon > 0$.

- This is a tight result! The problem has a simple $\frac{1}{2}$-approximation algorithm.

Hardness of approximating $\textsc{Max-3SAT}$

Corollary

*For every $\varepsilon > 0$, $(7/8 + \varepsilon)$-approximation to $\textsc{Max-3SAT}$ is **NP**-hard.*

Proof: Take an instance of $\textsc{Max-E3LIN}$ with n variables and m constraints, where we want to determine whether at least a $(1 - \nu)$ or at most a $(1/2 + \nu)$ fraction of constraints can be satisfied. We Construct an an instance of $\textsc{Max-3SAT}$ with $4m$ clauses and n variables:

$$\begin{array}{lll} E_1 : & x_1 \oplus x_2 \oplus x_3 = 0 & \\ E_2 : & x_4 \oplus x_1 \oplus x_7 = 1 & \\ & \vdots & \Rightarrow \\ E_m : & x_5 \oplus x_n \oplus x_9 = 1 & \end{array}
\quad
\begin{array}{l} E_1 : \left\{ \begin{array}{l} (\overline{x_1} \vee x_2 \vee x_3), (x_1 \vee \overline{x_2} \vee x_3), \\ (x_1 \vee x_2 \vee \overline{x_3}), (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}). \end{array} \right. \\ \quad\vdots \\ E_m : \left\{ \begin{array}{l} (\overline{x_5} \vee \overline{x_n} \vee x_9), (x_5 \vee \overline{x_n} \vee \overline{x_9}), \\ (\overline{x_5} \vee x_n \vee \overline{x_9}), (x_5 \vee x_n \vee x_9). \end{array} \right. \end{array}$$

Hardness of approximating $\textsc{Max-3SAT}$

- If x_i, x_j, x_k satisfy a linear constraint, then they satisfy all four corresponding clauses. Otherwise, they satisfy exactly three clauses.
- **Completeness:** If at least a $(1-\nu)$ fraction of constraints can be satisfied, then the fraction of clauses that can be satisfied is at least

$$(1-\nu)\cdot\frac{4}{4}+\nu\cdot\frac{3}{4}=\left(1-\frac{\nu}{4}\right)$$

- **Soundness:** If at most a $(1/2+\nu)$ fraction of constraints can be satisfied, then the fraction of clauses that can be satisfied is at most

$$\left(\frac{1}{2}+\nu\right)\cdot\frac{4}{4}+\left(\frac{1}{2}-\nu\right)\cdot\frac{3}{4}=\left(\frac{7}{8}+\frac{\nu}{4}\right)=\rho$$

- Therefore, it is **NP**-hard to approximate $\textsc{Max-3SAT}$ within a factor better than $\rho=(7/8+\nu/4)=(7/8+\varepsilon)$. □

Outline

Vertex Cover & Independent Set

Vertex Cover:

- Current best: $\left(2 - \Theta\left(1/\sqrt{\log |V|}\right)\right)$-apx. [Karakostas, 2009]
- **NP**-hard to approximate within a factor of 1.3606. [Dinur & Safra, 2005]
- If UGC is true, Vertex Cover cannot be approximated within any constant factor better than 2. [Khot & Regev, 2008]

Independent Set:

- Trivial $(1/n)$-approximation: return any vertex of the graph.
- For every $\varepsilon > 0$ there is no $(1/n^{1-\varepsilon})$-approximation algorithm. [Zuckerman, 2007]
- No $(2^{O(\sqrt{\log d})}/d)$-approximation algorithm exists, where d is the graph's maximum degree. [Trevisan, 2001]

Max Cut & Metric TSP

Max-Cut:

- **NP**-hard to approximate with a ratio better than $16/17 \approx 0.941$. [Håstad, 2001]
- Using semidefinite programming, there is an approximation algorithm with a ratio of $\alpha \approx 0.878$. [Goemans & Williamson, 1995]
- If UGC is true, this is the best possible approximation ratio for Max Cut. [Khot et al., 2007]

Metric TSP:

- The best known approximation ratio is $3/2$. [Christofides, 1976]
- 5500-apx for asymmetric distances. [Svensson, Tarnawski & Végn, 2017]
- There is no polynomial time algorithm for Metric TSP with performance ratio better that $123/122$ (and $75/74$ for asymmetric distances). [Karpinski, Lampis & Schmied, 2013]

Colorability

- For every $\varepsilon > 0$, there is no $n^{1-\varepsilon}$-approximation algorithm. [Zuckerman, 2007]

An interesting special case of the problem is to devise algorithms that color a 3-colorable graph with a minimum number of colors.

- There is a polynomial time algorithm that colors every 3-colorable graph with at most $\tilde{O}(n^{3/14 \approx 0.214})$ colors. [Karger & Blum, 1997]
- There is no polynomial time algorithm that colors every 3-colorable graph using at most 4 colors. [Khanna, Linial & Safra, 1993]

This is one of the largest gaps between known approximation algorithms and known inapproximability results.

Outline

Deterministic choices

We saw the following randomized algorithm for $\textsc{Max-3SAT}$

Algorithm

For every variable x_i, set $x_i = 1$ with probability $\frac{1}{2}$, independently.

and we proved that

$$\mathbf{E}[\mathrm{SOL}] = \mathbf{E}\left[\sum_{j=1}^{m} Y_j\right] = \sum_{j=1}^{m} \mathbf{Pr}[Y_j = 1] = \sum_{j=1}^{m} \frac{7}{8} = \frac{7}{8}m \geq \frac{7}{8}\mathrm{OPT}.$$

Idea: What if we set $x_1 = b_1 \in \{0, 1\}$ deterministically and all others 1 with probability $\frac{1}{2}$?

Then, the number of clauses satisfied would be $\mathbf{E}[\mathrm{SOL}|x_1 = b_1]$.

Maximization of conditional expectations

- We have

$$\begin{aligned}\mathbf{E}[\mathrm{SOL}] &= \mathbf{E}[\mathrm{SOL}|x_1 = 0] \cdot \mathbf{Pr}[x_1 = 0] + \mathbf{E}[\mathrm{SOL}|x_1 = 1] \cdot \mathbf{Pr}[x_1 = 1] \\ &= \frac{1}{2}\big(\underbrace{\mathbf{E}[\mathrm{SOL}|x_1 = 0]}_{E_0} + \underbrace{\mathbf{E}[\mathrm{SOL}|x_1 = 1]}_{E_0}\big).\end{aligned}$$

- So $\mathbf{E}[\mathrm{SOL}]$ is a convex combination of E_0 and E_1. Hence,

$$\max_{b_1 \in \{0,1\}} \mathbf{E}[\mathrm{SOL}|x_1 = b_1] \geq \mathbf{E}[\mathrm{SOL}] = \frac{7}{8}m.$$

- Suppose that we set $x_1 = b_1 \in \{0, 1\}$ so as to maximize the conditional expectation $\mathbf{E}[\mathrm{SOL}|x_1 = b_1]$, and for the rest variables we continue with probability $1/2$. Then, in expectation, we satisfy at least as many clauses as the full-randomized algorithm!

Derandomizing...

Do the same for all variables, sticking with every choice along the way.

Algorithm 1 DE-RANDOMIZED(ϕ, n, m)

for i=1:n **do**
  $E_0 \leftarrow \mathbf{E}[\mathrm{SOL} \mid x_1 = b_1, \ldots, x_{i-1} = b_{i-1}, x_i = 0]$;
  $E_1 \leftarrow \mathbf{E}[\mathrm{SOL} \mid x_1 = b_1, \ldots, x_{i-1} = b_{i-1}, x_i = 1]$;
  if $E_0 \geq E_1$ **then**
    $b_i \leftarrow 0$;
  else
    $b_i \leftarrow 1$;
  end if
end for
Output: Assign b_i to variable x_i.

Analysis

- In every iteration $i \in [n]$, we choose b_i so as to maximize the conditional expectation

$$\mathbf{E}[\text{SOL}|x_1 = b_1, \ldots, x_i = b_i].$$

- We already proved that

$$\mathbf{E}[\text{SOL}|x_1 = b_1] \geq \mathbf{E}[\text{SOL}] = \frac{7}{8}m.$$

- Using the same argument, for every $i \in \{2, \ldots, n\}$

$$\mathbf{E}[\text{SOL}|x_1 = b_1, \ldots, x_i = b_i] \geq \mathbf{E}[\text{SOL}|x_1 = b_1, \ldots, x_{i-1} = b_{i-1}].$$

- Then, by induction:

$$\begin{aligned}\mathbf{E}[\text{SOL}|x_1 = b_1, \ldots, x_n = b_n] &\geq \mathbf{E}[\text{SOL}|x_1 = b_1, \ldots, x_{n-1} = b_{n-1}]\\ &\;\;\vdots\\ &\geq \mathbf{E}[\text{SOL}|x_1 = b_1] \geq \mathbf{E}[\text{SOL}] = \frac{7}{8}m.\end{aligned}$$

Analysis

- $\mathbf{E}[\mathrm{SOL}|x_1 = b_1, \ldots, x_n = b_n]$ is the number of clauses satisfied by the algorithm. Thus, the derandomized algorithm is a deterministic $7/8$-approximation.
- We must show that in each iteration i the conditional expectation can be computed in polynomial time.
- We have that

$$\mathbf{E}[\mathrm{SOL}|x_1 = b_1, \ldots, x_i = b_i] = \sum_{j=1}^{m} \underbrace{\mathbf{Pr}[Y_j = 1|x_1 = b_1, \ldots, x_i = b_i]}_{P_j}.$$

- Let $C_j = (l_1 \vee l_2 \vee l_3)$, then

$$P_j = \begin{cases} 1, & \text{one literal already true} \\ 0, & \text{all literals already false} \\ 3/4, & \text{one literal already false} \\ 1/2, & \text{two literals already false} \end{cases}$$

- We compute P_j in $O(1)$ time. So, naively, the running time of the algorithm is $O(n \cdot m)$.

Bibliography

Chapter 11 and Section 22.4 of

Sanjeev Arora, Boaz Barak
Computational Complexity: A Modern Approach.
Cambridge University Press, 2009.

Sections 5.1-5.2 of

D. P. Williamson, D. B. Shmoys
The Design of Approximation Algorithms.
Cambridge University Press, 2011.