

HARDNESS OF APPROXIMATIONS - A SURVEY *

Authors: Arora, Lund

Presenter: Rave Harpaz †

December 11, 2001

*Sanjeev Arora. The Approximability of NP-hard problems. *ACM STOC'98*.

Sanjeev Arora, Carsten Lund. Hardness of Approximations In *Approximation Algorithms for NP-hard Problems*, Dorit Hochbaum, Ed. PWS Publishing ,pages 399-446, 1996.

Sanjeev Arora. Probabilistic checking of proofs and the hardness of approximation problems. *Ph.D. Thesis, UC Berkeley, 1994*.

†Computer Science Department, The Graduate Center, City University of New York.

What will be covered:

1. Introduction and motivation
2. History
3. Classification of inapproximability
4. The **PCP** theorem
5. How to prove inapproximability
6. The Hardness of Approximating MAX-3SAT
7. Inapproximability results of other NP-hard problems
8. Open problems

Introduction

- Computing good approximations to many **NP-hard** optimization problems is **NP-hard**, i.e. **no easier** than computing exact solutions.
- **Proving** NP-hardness of approximations involves a reduction from a known NP-hard problem to the new problem.
- The **Cook-Karp-Levin** techniques for reductions do not suffice; the reduction must produce a **gap** in the value of the optimum.
- A new technique for constructing **gap-producing** reductions, relies upon new probabilistic characterizations of the class-**NP**.
This is known as the **PCP Theorem**.

Motivation

- Which problems can or cannot be approximated and to what extent.
- Insight to how to design an approximation algorithm for a certain problem.
- Further classification of NP-hard problems according to their approximability.

History

- Early 70's, after the discovery of NP-completeness, Garey, Graham, Ullman and then Johnson formalized the notion of an approximation algorithm.
- 1976, Sahni and Gozalez showed that achieving any constant approximation for TSP is NP-hard.
- Early 80's, many approximation algorithms were designed for various NP-hard problems.
- Late 80's, Papadimitriou and Yannakakis laid the study of approximability on a sound ground by defining a class of optimization problems called MAX-SNP. Using a certain approximability reduction they defined completeness for MAX-SNP and showed that MAX-3SAT is MAX-SNP complete.
- This result implies that an inapproximability result for MAX-3SAT would generalize to a large class of problems, which motivated the discovery of the PCP Theorem.
- The concept of PCPs evolved out of interactive proofs, defined in the mid 80's by Goldwasser, Micali, Rackoff and Babai.
- 1991, Feige et al. proved the first inapproximability result to come out of the PCP area. They showed that if any poly-time algorithm can achieve a constant approximation ratio for MAX-CLIQUE then all NP problems are solvable in $n^{O(\log \log n)}$.
- 1992, Arora and Safra formalized and named the class PCP and used it to give the class NP a new probabilistic definition.
- 1992, Arora, Lund, Motwani, Sudan and Szegedy proved the PCP Theorem and showed the MAX-SNP-hard problems do not have a PTAS if $P \neq NP$.

Classification of inapproximability:

- Not all NP-hard optimization problems are equivalent in terms of their approximability behavior.
- Inapproximability results divide problems into four classes*, based on the approximation ratio that is provably hard to achieve.
- Inapproximability results within a class share common ideas.

Class	Approximation ratio hard to achieve	Representative problem
1	$1 + \epsilon$	MAX-3SAT
2	$O(\log n)$	SETCOVER
3	$2^{\log^{1-\gamma} n}$	LABELCOVER
4	n^ϵ	CLIQUE

The four classes and their representative problems.

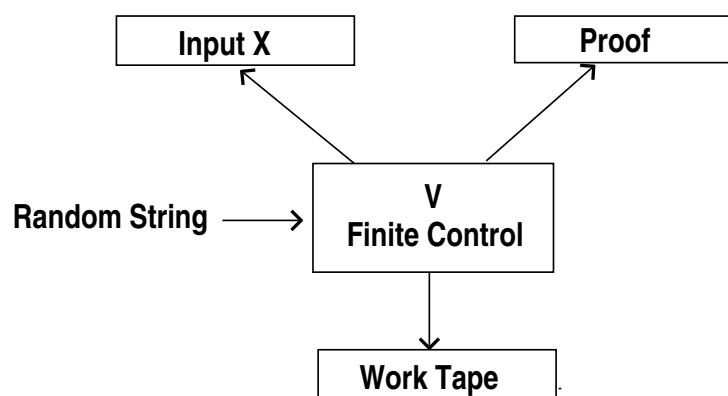
*known so far, new results may reveal other classes or collapse two classes.

The PCP theorem

Definition 1. A **verifier** is a polynomial-time probabilistic Turing Machine- M , used to verify membership proofs, containing:

- *Input tape*
- *Work tape*
- *A source of random bits, called the **random string**, denoted- τ*
- *read only tape called the **proof string**, denoted- Π*

M has random access to Π . The work tape contains a special addressing portion on which M can write the address of a location in Π , and then read just the bit in that location. This operation is called a **query**.



Definition 2. A verifier is $(r(n), q(n))$ – restricted if on each input of size n it uses at most $O(r(n))$ random bits for its computation, and queries at most $O(q(n))$ bits of the proof.

M operates as follows: on input x , where $|x| = n$, it reads τ of length $cr(n)$, computes a sequence of $kq(n)$ locations and queries those locations in Π . depending on what these bits were, it accepts or rejects.

Definition 3. $M^\Pi(x, \tau) = 1$ if M accepts input x , with access to proof Π , using τ , and 0 otherwise.

Definition 4. A verifier M can probabilistically check membership proofs for language L if:

- $x \in L \Rightarrow \exists \Pi$ s.t. $Pr[M^\Pi(x, \tau) = 1] = 1$
(i.e. M accepts for every random string with probability 1).
- $x \notin L \Rightarrow \forall \Pi$ $Pr[M^\Pi(x, \tau) = 1] < \frac{1}{2}$
(i.e. M rejects all proofs with probability at least $\frac{1}{2}$).

Definition 5. The complexity class $PCP(r(n), q(n))$ is a class consisting of all languages L for which membership proofs can be checked by a $(r(n), q(n))$ – restricted verifier.

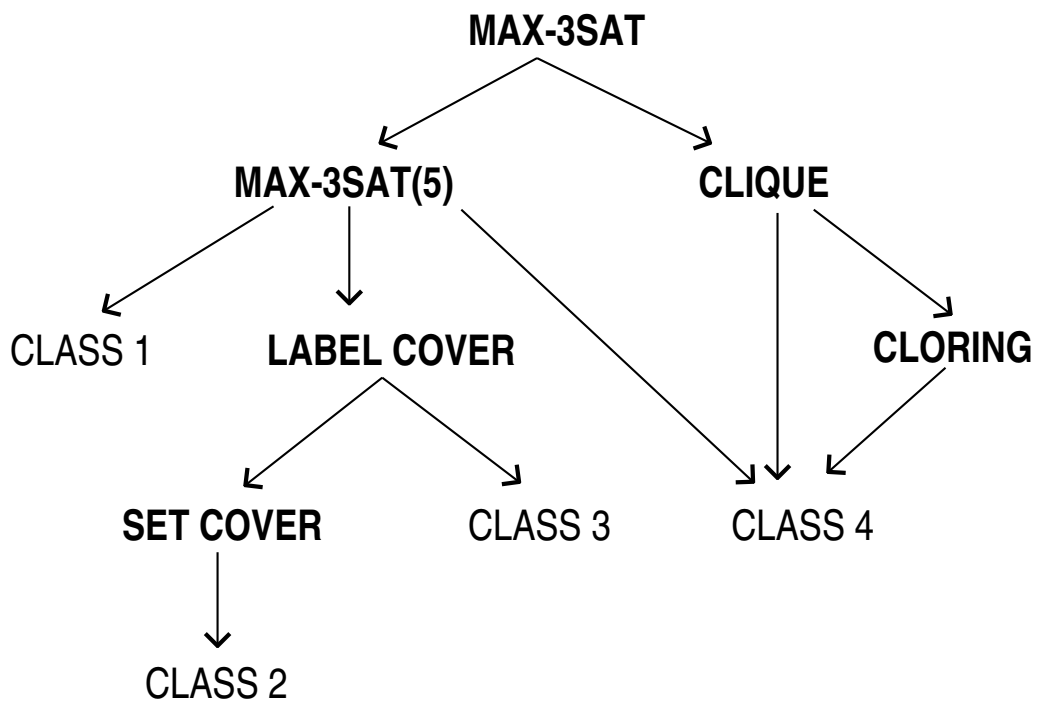
Note that $NP = PCP(0, poly(n))$, since $PCP(0, poly(n))$ is the set of languages for which membership proofs can be checked in deterministic polynomial-time, which is exactly the set NP .

Theorem 1 (PCP Theorem). $NP = PCP(\log n, 1)$

Since the verifier uses $O(\log n)$ random bits, it will have $2^{c \log n} = n^c$ possible runs, and in each run it reads only $O(1)$ bits in the proof-string.

How to prove inapproximability

- As with proving NP-hardness, we start with a known inapproximable problem and reduce it, using a gap preserving reduction to the new problem.
- The known problems, which are referred to as the six "canonical" inapproximable problems, serve the purpose of deriving new inapproximability results. they play the same role in inapproximability as the canonical problems used in proving NP-completeness.
- In particular, as 3SAT is the most basic canonical problem in proving NP-completeness, its optimization version- MAX-3SAT is the most basic canonical problem in proving inapproximability. i.e. reductions from MAX-3SAT can prove the hardness of all canonical problems.
- the other canonical problems are: MAX-3SAT(5), R-CLIQUE, LABELCOVER, SETCOVER and COLORING (they will be defined later).



sequence of transformations used to prove the inapproximability of the six canonical problems.

The Hardness of Approximating MAX-3SAT

We start by proving, using the PCP Theorem, the hardness result for MAX-3SAT. Then using this result we prove the hardness of approximating other NP-hard problems.

Definition 6. MAX-3SAT

input A 3CNF formula φ

output An assignment that maximizes the number of satisfied clauses in φ

Let $\text{MAX-3SAT}(\varphi)$ denote the maximum fraction of clauses that can be satisfied. i.e $\text{MAX-3SAT}(\varphi) \leq 1$

Definition 7. β -approximation to MAX-3SAT

input A 3CNF formula φ

output An assignment that satisfies at least $(\frac{1}{\beta})\text{MAX-3SAT}(\varphi)$ of the clauses in φ

This is the problem that will be proved to be NP-Hard

Theorem 2. $\text{NP} = \text{PCP}(\log n, 1)$ if and only if the following polynomial-time reduction $\psi : \text{SAT} \rightarrow \text{MAX-3SAT}$, together with some $\varepsilon > 0$, exists:

$$\begin{aligned}\varphi \in \text{SAT} &\Rightarrow \text{MAX-3SAT}(\psi(\varphi)) = 1, \\ \varphi \notin \text{SAT} &\Rightarrow \text{MAX-3SAT}(\psi(\varphi)) < \frac{1}{1 + \varepsilon}\end{aligned}$$

Proof: (\Leftarrow) Given $L \in NP$ and the reduction ψ , show that $L \in PCP(\log n, 1)$.

Let $x \in L$, M (verifier) reduces it to SAT and then using ψ to MAX-3SAT.

M expects as a membership proof- Π for x a satisfying assignment for the MAX-3SAT instance.

M then checks this membership proof probabilistically by choosing a clause at random and querying Π for the clause's values, it accepts iff the clause is satisfied.

If $x \in L$ then there is a proof for which M accepts with probability 1. if $x \notin L$ then all proofs are rejected with probability at least $1 - \frac{1}{1+\varepsilon}$.

Repeating the verification $O(\frac{1}{\varepsilon})$ times makes the rejection probability at least $\frac{1}{2}$.

(\Rightarrow) Given $NP = PCP(\log n, 1)$, and hence $SAT \in PCP(\log n, 1)$, show that there exists $\psi : SAT \rightarrow MAX-3SAT$ for some $\varepsilon > 0$.

Let φ be an instance of SAT, given a random string- τ , M computes $q_1(\varphi), \dots, q_k(\varphi)$ the positions in Π that will be queried and then accepts or rejects.

For each τ_i where $i = 1, \dots, n^c$ construct (using a truth table), a 3CNF formula φ_{τ_i} that represents the relation between the queried bits in the proof and the acceptance behavior of M . i.e. $\varphi_{\tau_i} = 1 \Leftrightarrow M^\pi(\varphi, \tau_i) = 1$.

Take the conjunction $\varphi_{\tau_1} \wedge \dots \wedge \varphi_{\tau_{n^c}}$ as the result of the desired reduction ψ on input φ .

If $\varphi \in SAT$ then $MAX-3SAT(\varphi_{\tau_1} \wedge \dots \wedge \varphi_{\tau_{n^c}}) = 1$, if $\varphi \notin SAT$ then at least half of φ_{τ_i} will be false, that is at least one clause in each φ_{τ_i} is false. hence $MAX-3SAT(\varphi_{\tau_1} \wedge \dots \wedge \varphi_{\tau_{n^c}}) < \frac{1}{1+\varepsilon}$ for some ε .

Therefore by the existence reduction we have proved, we have the following,

Corollary 1. *There is some $\beta > 1$ such that MAX-3SAT is hard to approximate within a factor of β .*

Inapproximability results of other NP-hard problems

Now we show that using **Gap-Preserving** reductions further inapproximability results can be achieved.

Definition 8. Let Π and Π' be two maximization problems. A **gap-preserving** reduction from Π to Π' with parameters $(c, \rho), (c', \rho')$ is a polynomial-time algorithm f . For each instance I of Π , algorithm f produces an instance $I' = f(I)$ of Π' . The optima of I and I' , say $OPT(I)$ and $OPT(I')$ respectively, satisfy the following property:

$$\begin{aligned} OPT(I) \geq c &\Rightarrow OPT(I') \geq c', \\ OPT(I) < \frac{c}{\rho} &\Rightarrow OPT(I') < \frac{c'}{\rho'}. \end{aligned}$$

Where c and ρ are functions of $|I|$, c', ρ' are functions of $|I'|$ and $\rho, \rho' \geq 1$.

Idea: suppose we have a reduction τ from SAT to Π that insures:

$$\varphi \in SAT \Rightarrow OPT(\tau(\varphi)) \geq c$$

$$\varphi \notin SAT \Rightarrow OPT(\tau(\varphi)) \leq \frac{c}{\rho}$$

then composing this reduction with a gap preserving reduction gives a reduction $f \circ \tau$ from SAT to Π' that ensures:

$$\varphi \in SAT \Rightarrow OPT(f(\tau(\varphi))) \geq c'$$

$$\varphi \notin SAT \Rightarrow OPT(f(\tau(\varphi))) \leq \frac{c'}{\rho'}$$

i.e. $f \circ \tau$ shows that achieving an approximation ratio ρ' for Π' is NP-hard.

Example: A gap-preserving reduction from **MAX-3SAT** to **CLIQUE** with parameters $(c, 1 + \varepsilon), (cN/3, 1 + \varepsilon)$, where N is the number of vertices in the graph.

let φ be a 3CNF formula with m clauses. Construct a new graph $\tau(\varphi)$ with $3m$ vertices where each clause is represented with a triple of vertices, one per literal. put an edge between two vertices iff they appear in two different triples (clauses) and are not the negation of each other.

A clique in the graph can contain only one vertex per triple and by looking at the literals in a clique we can find an **assignment that satisfies as many clauses as there are vertices in the clique**. Thus,

$$MAX - 3SAT = c \Rightarrow CLIQUE(\tau(\varphi)) = cm,$$

$$MAX - 3SAT < \frac{c}{1 + \varepsilon} \Rightarrow CLIQUE(\tau(\varphi)) < \frac{cm}{1 + \varepsilon}$$

\therefore Approximating CLIQUE within a factor $(1 + \varepsilon)$ is NP-hard.
(we will later see a stronger inapproximability result)

Theorem 3. *The following reductions for the canonical problems exist.*

Definition 9. MAX-3SAT(5) *subcase case of MAX-3SAT where every variable appears in at most 5 clauses.*

1. A polynomial-time reduction $\tau_1 : SAT \rightarrow MAX-3SAT(5)$ that ensures:

$$\begin{aligned} I \in SAT &\Rightarrow MAX - 3SAT(\tau_1(I)) = 1, \\ I \notin SAT &\Rightarrow MAX - 3SAT(\tau_1(I)) < \frac{1}{1 + \varepsilon} \end{aligned}$$

\therefore Approximating MAX-3SAT(5) within a factor $(1 + \varepsilon)$ is NP-hard.

Definition 10. R-CLIQUE *The input consists of an integer r , an r -partite graph G and its r -partition. The goal is to find the largest clique in G , i.e. the largest complete r -partite subgraph of G . Let the objective function $R\text{-CLIQUE}(G) = \frac{k}{r} \leq 1$, where k is the size of the largest clique.*

2. A polynomial-time reduction $\tau_2 : SAT \rightarrow R\text{-CLIQUE}$ that ensures for some $\delta > 0$:

$$I \in SAT \Rightarrow R\text{-CLIQUE}(\tau_2(I)) = 1,$$

$$I \notin SAT \Rightarrow R\text{-CLIQUE}(\tau_2(I)) < \frac{1}{n^\delta}$$

To prove this reduction we first use the SAT to MAX-3SAT reduction and then reduce MAX-3SAT to R-CLIQUE.

\therefore Approximating R-CLIQUE and thus CLIQUE within a factor n^δ is NP-hard.

Definition 11. LABELCOVER_{max} Given a regular bipartite graph $G = (V_1, V_2, E)$, an integer N that defines the set of labels which are $\{1, 2, \dots, N\}$ and a partial function $\Pi_e : \{1, 2, \dots, N\} \rightarrow \{1, 2, \dots, N\}$ for every $e \in E$. A labelling has to associate a non-empty set of labels with every vertex in G . It is said to cover an edge $e = (u, v)$ if for every label a_2 assigned to v , there is some label a_1 assigned to u such that $\Pi_e(a_1) = a_2$. The collection of such partial functions is denoted Π . The goal is a labelling that assigns one label per vertex and maximizes the fraction of covered edges.

Definition 12. Quasi-NP-hard a problem is Quasi-NP hard if any polynomial-time algorithm for it can be used to solve all NP-problems in quasi-polynomial time, i.e. $2^{\text{poly}(\log n)}$.

3. A polynomial-time reduction $\tau_3 : SAT \rightarrow LABELCOVER_{max}$ that ensures:

$$I \in SAT \Rightarrow LABELCOVER_{max}(\tau_3(I)) = 1,$$

$$I \notin SAT \Rightarrow LABELCOVER_{max}(\tau_3(I)) < \frac{1}{2^{\log^{1-\gamma} n}}$$

where γ is a small positive constant and n is the size of $\tau_3(I)$.

The proof for this reduction uses a reduction from SAT to MAX-3SAT and then to $LABELCOVER_{max}$.

\therefore Approximating $LABELCOVER_{max}$ within a factor $2^{\log^{1-\gamma} n}$ is Quasi-NP-hard for any $\gamma > 0$.

Definition 13. SETCOVER Given a set U and a collection of its subsets S_1, S_2, \dots, S_m satisfying $\bigcup_{i=1}^m S_i = U$, find the minimum subcollection that covers U , where the objective function is $|I|$, the number of sets in the subcollection.

4. A polynomial-time reduction $\tau_4 : SAT \rightarrow SETCOVER$ that ensures:

$$I \in SAT \Rightarrow SETCOVER(\tau_4(I)) = K(|I|),$$

$$I \notin SAT \Rightarrow SETCOVER(\tau_4(I)) > K(|I|) \cdot \frac{\log n}{48}$$

where $K(|I|)$ is a polynomial-time computable function and n is the size of $\tau_4(I)$.

To prove this reduction we reduce $LABELCOVER_{max}$ to SETCOVER and use the inapproximability result for $LABELCOVER_{max}$.

\therefore Approximating SETCOVER within a factor $\log n/48$ is Quasi-NP-hard.

Definition 14. COLORING Given a Graph G , assign a color to each vertex such that no two adjacent vertices have the same color. The goal is to minimize the total number of colors used. The objective function is the number of colors in a coloring. The minimum of the objective function is called the chromatic number of a graph, denoted $\chi(G)$.

5. A polynomial-time reduction $\tau_5 : SAT \rightarrow COLORING$ that ensures for some $\delta > 0$:

$$I \in SAT \Rightarrow \chi(\tau_5(I)) = K(|I|),$$

$$I \notin SAT \Rightarrow \chi(\tau_5(I)) > K(|I|)n^\delta$$

This reduction uses a reduction from R-CLIQUE to COLORING using the property that A k -coloring in G is a clique cover of size k in \overline{G} .

\therefore Approximating COLORING within a factor n^δ is NP-hard.

Open problems

- For many problems such as CLIQUE, COLORING and MAX-3SAT, there is a big gap between the approximation ratio that is provably hard to achieve and the ratio that we can achieve in polynomial time. Can this gap be closed?
- Identify inapproximability results that current techniques cannot prove.
- Prove inapproximability results for some of the counting problems ($\#P$), such as number of perfect matchings.