Probabilistically Checkable Proofs (PCP) and Hardness of Approximations

Nicolas Nisse

Inria, France

Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271, Sophia Antipolis, France

JCALM 2015

March 10th, 2015

Outline

1. Introduction on Approximation Algorithms
2. PCP & PCP Theorem ($NP = PCP(O(\log n), O(1))$)
3. PCP Theorem $\Leftrightarrow$ MAX-3SAT is hard to Approximate
4. From Label-Cover to MAX-Clique

Outline

1. Introduction on Approximation Algorithms
2. PCP & PCP Theorem ($NP = PCP(O(\log n), O(1))$)
3. PCP Theorem $\Leftrightarrow$ MAX-3SAT is hard to Approximate
4. From Label-Cover to MAX-Clique

Approximation Algorithms

Π a maximization Problem

c-Approximation for Π — $1 < c$ constant or depends on input length

- deterministic polynomial-time algorithm $\mathcal{A}$
- for any input I, $\mathcal{A}$ returns a solution with value at least $OPT(I)/c$.

Example: MAX-3SAT
input: a 3CNF formula Φ
output: maximum (over all assignments A) number of clauses of Φ satisfied by A

remark: MAX-3SAT is NP-hard

Algorithm: Random assignment $\Rightarrow \frac{8}{7}$-approximation for MAX-3SAT

each clause is satisfied with probability $\frac{7}{8}$

Application of PCP Theorem : above algorithm (derandomized) is optimal unless $P = NP$.

Approximation Algorithms

Π a maximization Problem

c-Approximation for Π $1 < c$ constant or depends on input length

- deterministic polynomial-time algorithm $\mathcal{A}$
- for any input I, $\mathcal{A}$ returns a solution with value at least $OPT(I)/c$.

Example: MAX-3SAT
input: a 3CNF formula Φ
output: maximum (over all assignments A) number of clauses of Φ satisfied by A

remark: MAX-3SAT is NP-hard

Algorithm: Random assignment $\Rightarrow \frac{8}{7}$-approximation for MAX-3SAT

each clause is satisfied with probability $\frac{7}{8}$

Application of PCP Theorem : above algorithm (derandomized) is optimal unless $P = NP$.

Approximation Algorithms

Π a maximization Problem

c-Approximation for Π $1 < c$ constant or depends on input length

- deterministic polynomial-time algorithm $\mathcal{A}$
- for any input I, $\mathcal{A}$ returns a solution with value at least $OPT(I)/c$.

Example: MAX-3SAT
input: a 3CNF formula Φ
output: maximum (over all assignments A) number of clauses of Φ satisfied by A

remark: MAX-3SAT is NP-hard

Algorithm: Random assignment $\Rightarrow \frac{8}{7}$-approximation for MAX-3SAT

each clause is satisfied with probability $\frac{7}{8}$

Application of PCP Theorem : above algorithm (derandomized) is optimal unless $P = NP$.

Hierarchy of NP-hard Problems

Fully Polynomial Time Approximation Scheme (FPTAS)

For any ϵ, $\exists$ $(1+\epsilon)$-approximation algorithm, polynomial in both n and $1/\epsilon$

e.g., Knapsack

Polynomial Time Approximation Scheme (PTAS)

For any ϵ, $\exists$ $(1+\epsilon)$-approximation algorithm, polynomial in n (typically $n^{f(\epsilon)}$)
but no FPTAS (unless P=NP)

e.g., Euclidean TSP

$\Theta(1)$-approximation but no PTAS (unless P=NP)

e.g., Vertex Cover, MAX-3SAT

$O(\log n)$-approximation but no constant approximation (unless P=NP)

e.g., Set Cover

only $n^{\Theta(1)}$-approximation, no $n^{1-\epsilon}$-approximation for any $\epsilon > 0$ (unless P=NP)

e.g., MAX-Clique

Hierarchy of NP-hard Problems

Fully Polynomial Time Approximation Scheme (FPTAS)

For any ϵ, $\exists$ $(1+\epsilon)$-approximation algorithm, polynomial in both n and $1/\epsilon$

e.g., Knapsack

Polynomial Time Approximation Scheme (PTAS)

For any ϵ, $\exists$ $(1+\epsilon)$-approximation algorithm, polynomial in n (typically $n^{f(\epsilon)}$)
but no FPTAS (unless P=NP)

e.g., Euclidean TSP

$\Theta(1)$-approximation but no PTAS (unless P=NP)

e.g., Vertex Cover, MAX-3SAT

$O(\log n)$-approximation but no constant approximation (unless P=NP)

e.g., Set Cover

only $n^{\Theta(1)}$-approximation, no $n^{1-\epsilon}$-approximation for any $\epsilon > 0$ (unless P=NP)

e.g., MAX-Clique

Hierarchy of NP-hard Problems

Fully Polynomial Time Approximation Scheme (FPTAS)

For any ϵ, $\exists$ $(1+\epsilon)$-approximation algorithm, polynomial in both n and $1/\epsilon$

e.g., Knapsack

Polynomial Time Approximation Scheme (PTAS)

For any ϵ, $\exists$ $(1+\epsilon)$-approximation algorithm, polynomial in n (typically $n^{f(\epsilon)}$)
but no FPTAS (unless P=NP)

e.g., Euclidean TSP

$\Theta(1)$-approximation but no PTAS (unless P=NP)

e.g., Vertex Cover, MAX-3SAT

$O(\log n)$-approximation but no constant approximation (unless P=NP)

e.g., Set Cover

only $n^{\Theta(1)}$-approximation, no $n^{1-\epsilon}$-approximation for any $\epsilon > 0$ (unless P=NP)

e.g., MAX-Clique

Hierarchy of NP-hard Problems

Fully Polynomial Time Approximation Scheme (FPTAS)

For any ϵ, $\exists$ $(1+\epsilon)$-approximation algorithm, polynomial in both n and $1/\epsilon$

e.g., Knapsack

Polynomial Time Approximation Scheme (PTAS)

For any ϵ, $\exists$ $(1+\epsilon)$-approximation algorithm, polynomial in n (typically $n^{f(\epsilon)}$)
but no FPTAS (unless P=NP)

e.g., Euclidean TSP

$\Theta(1)$-approximation but no PTAS (unless P=NP)

e.g., Vertex Cover, MAX-3SAT

$O(\log n)$-approximation but no constant approximation (unless P=NP)

e.g., Set Cover

only $n^{\Theta(1)}$-approximation, no $n^{1-\epsilon}$-approximation for any $\epsilon > 0$ (unless P=NP)

e.g., MAX-Clique

Hierarchy of NP-hard Problems

Fully Polynomial Time Approximation Scheme (FPTAS)

For any ϵ, $\exists$ $(1+\epsilon)$-approximation algorithm, polynomial in both n and $1/\epsilon$

e.g., Knapsack

Polynomial Time Approximation Scheme (PTAS)

For any ϵ, $\exists$ $(1+\epsilon)$-approximation algorithm, polynomial in n (typically $n^{f(\epsilon)}$)
but no FPTAS (unless P=NP)

e.g., Euclidean TSP

$\Theta(1)$-approximation but no PTAS (unless P=NP)

e.g., Vertex Cover, MAX-3SAT

$O(\log n)$-approximation but no constant approximation (unless P=NP)

e.g., Set Cover

only $n^{\Theta(1)}$-approximation, no $n^{1-\epsilon}$-approximation for any $\epsilon > 0$ (unless P=NP)

e.g., MAX-Clique

Outline

Reminder: Definition of NP

$L \in NP$ iff $\exists$ a deterministic Verifier V s.t.

for any input (word) w

- V polynomial-time in $|w|$
- and
  - if $w \in L$ then $\exists$ proof (certificate) x with $V(w,x) = 1$
  - if $w \notin L$ then $\forall$ proof x, $V(w,x) = 0$

examples:
3-SAT: proof= assignment of variables
3-Coloriage: proof= Labelling of vertices

Verifier checks (1) that certificate is consistent (2) that the answer is correct

Definition of the class PCP(r,q)

$L \in PCP(r,q)$ iff $\exists$ a randomized Verifier V s.t.

for any input w

- V polynomial-time in $|w|$, uses a string ρ of $O(r(|w|))$ bits of randomness and only checks $O(q(|w|))$ bits of the proof, and
  - if $w \in L$ then $\exists$ proof x with $Pr_\rho(V(w,x,\rho)=1)=1$ (**completeness**).
  - if $w \notin L$ then $\forall$ proof x, $Pr_\rho(V(w,x,\rho)=1) \leq 1/2$ (**soundness**).

proof x has size $\leq q(|w|) \cdot 2^{r(|w|)}$ and only $O(q(|w|))$ bits of x are checked.

If the proof x is correct, then w is accepted with proba 1 (**completeness**).
Otherwise (if proof x is a fake), w is rejected with probability at least $1/2$ (**soundness**).

tradeoff between randomness (r) and number of bits (q) checked

Relationships between PCP and NP

$NP \subseteq PCP(0, poly(n))$

idea: NP-proof has polynomial size and the PCP-Verifier can check it completely

$PCP(O(\log n), O(1)) \subseteq NP$

idea: NP-Verifier: apply PCP-Verifier for each of the $2^{O(\log n)}$ possible random strings
i.e., "check the full proof" in polynomial-time

$NP \subseteq PCP(n^{O(1)}, O(1))$

idea: Increase the size of NP-proof s.t. if it is a fake, the mistakes are "everywhere"

PCP Theorem: $NP \subseteq PCP(O(\log n), O(1))$ Corollary: $NP = PCP(O(\log n), O(1))$

Initial proof: [Arora,Safra FOCS'92] [Arora,Lund,Motwani,Sudan,Szegedy FOCS'92]
[Arora FOCS'95]

improvements: [Raz STOC'95] [Hastad JACM'01]
"easier" proof: [Dinur STOC'06, JACM'07]

Relationships between PCP and NP

$NP \subseteq PCP(0, poly(n))$

idea: NP-proof has polynomial size and the PCP-Verifier can check it completely

$PCP(O(\log n), O(1)) \subseteq NP$

idea: NP-Verifier: apply PCP-Verifier for each of the $2^{O(\log n)}$ possible random strings
i.e., "check the full proof" in polynomial-time

$NP \subseteq PCP(n^{O(1)}, O(1))$

idea: Increase the size of NP-proof s.t. if it is a fake, the mistakes are "everywhere"

PCP Theorem: $NP \subseteq PCP(O(\log n), O(1))$ Corollary: $NP = PCP(O(\log n), O(1))$

Initial proof: [Arora,Safra FOCS'92] [Arora,Lund,Motwani,Sudan,Szegedy FOCS'92] [Arora FOCS'95]

improvements: [Raz STOC'95] [Hastad JACM'01]
"easier" proof: [Dinur STOC'06, JACM'07]

Relationships between PCP and NP

$NP \subseteq PCP(0, poly(n))$

idea: NP-proof has polynomial size and the PCP-Verifier can check it completely

$PCP(O(\log n), O(1)) \subseteq NP$

idea: NP-Verifier: apply PCP-Verifier for each of the $2^{O(\log n)}$ possible random strings i.e., "check the full proof" in polynomial-time

$NP \subseteq PCP(n^{O(1)}, O(1))$

idea: Increase the size of NP-proof s.t. if it is a fake, the mistakes are "everywhere"

PCP Theorem: $NP \subseteq PCP(O(\log n), O(1))$ Corollary: $NP = PCP(O(\log n), O(1))$

Initial proof: [Arora,Safra FOCS'92] [Arora,Lund,Motwani,Sudan,Szegedy FOCS'92] [Arora FOCS'95]

improvements: [Raz STOC'95] [Hastad JACM'01]
"easier" proof: [Dinur STOC'06, JACM'07]

Relationships between PCP and NP

$NP \subseteq PCP(0, poly(n))$

idea: NP-proof has polynomial size and the PCP-Verifier can check it completely

$PCP(O(\log n), O(1)) \subseteq NP$

idea: NP-Verifier: apply PCP-Verifier for each of the $2^{O(\log n)}$ possible random strings i.e., "check the full proof" in polynomial-time

$NP \subseteq PCP(n^{O(1)}, O(1))$

idea: Increase the size of NP-proof s.t. if it is a fake, the mistakes are "everywhere"

PCP Theorem: $NP \subseteq PCP(O(\log n), O(1))$ Corollary: $NP = PCP(O(\log n), O(1))$

Initial proof: [Arora,Safra FOCS'92] [Arora,Lund,Motwani,Sudan,Szegedy FOCS'92] [Arora FOCS'95]

improvements: [Raz STOC'95] [Hastad JACM'01]

"easier" proof: [Dinur STOC'06, JACM'07]

PCP Theorem

$NP = PCP(O(\log n), O(1))$ [Arora *et al.*'92]

Consequence: for any $L \in NP$, $\exists$ a Verifier V s.t.

for any input w of size n

- V polynomial-time in $|w|$, uses a string ρ of $O(\log n)$ bits of randomness and only checks $O(1)$ bits of the proof, and
  - if $w \in L$ then $\exists$ proof x with $Pr_\rho(V(w,x,\rho)=1)=1$ (**completeness**).
  - if $w \notin L$ then $\forall$ proof x, $Pr_\rho(V(w,x,\rho)=1) \leq 1/2$ (**soundness**).

Remark 1: $PCP(O(\log n), O(1))$ is stable by polynomial-time reductions.

i.e., if $L' \prec_{poly} L$ and $L \in PCP(O(\log n), O(1))$ then $L' \in PCP(O(\log n), O(1))$

$L' \prec_{poly} L$ means $\exists$ a poly-time reduction from L to L' (i.e., L not "easier" than L').

Remark 2: V may have extra/restricted properties [Arora *et al.*'92, Raz'95,Hastadt'01]

Outline

MAX-3SAT is hard to Approximate (1/2)

Goal: prove that $\exists c_0$ s.t. $\forall \epsilon$, there is no $(c_0 - \epsilon)$-approx of MAX-3SAT unless $P = NP$

MAX-3SAT

input: a 3CNF formula Φ
output: maximum (over all assignments A) number of clauses of Φ satisfied by A

GAP-3SAT$_s$ $0 < s < 1$

input: a 3CNF formula Φ with m clauses
output:

- YES if Φ is satisfiable
- NO if at most $s \cdot m$ clauses are satisfiable (for all assignments)
- anything otherwise

Lemma

If $\exists s < 1$ s.t. GAP-3SAT$_s$ is NP-hard then there is no s-approx of MAX-3SAT unless $P = NP$

MAX-3SAT is hard to Approximate (1/2)

Goal: prove that $\exists c_0$ s.t. $\forall \epsilon$, there is no $(c_0 - \epsilon)$-approx of MAX-3SAT unless $P = NP$

MAX-3SAT

input: a 3CNF formula Φ
output: maximum (over all assignments A) number of clauses of Φ satisfied by A

GAP-3SAT$_s$ $0 < s < 1$

input: a 3CNF formula Φ with m clauses
output:

- YES if Φ is satisfiable
- NO if at most $s \cdot m$ clauses are satisfiable (for all assignments)
- anything otherwise

Lemma

If $\exists s < 1$ s.t. GAP-3SAT$_s$ is NP-hard
then there is no s-approx of MAX-3SAT unless $P = NP$

MAX-3SAT is hard to Approximate (1/2)

Goal: prove that $\exists c_0$ s.t. $\forall \epsilon$, there is no $(c_0 - \epsilon)$-approx of MAX-3SAT unless $P = NP$

MAX-3SAT

input: a 3CNF formula Φ
output: maximum (over all assignments A) number of clauses of Φ satisfied by A

GAP-3SAT$_s$ $0 < s < 1$

input: a 3CNF formula Φ with m clauses
output:

- YES if Φ is satisfiable
- NO if at most $s \cdot m$ clauses are satisfiable (for all assignments)
- anything otherwise

Lemma

If $\exists s < 1$ s.t. GAP-3SAT$_s$ is NP-hard
then there is no s-approx of MAX-3SAT unless $P = NP$

MAX-3SAT is hard to Approximate (2/2)

PCP $\Leftrightarrow$ Hardness of Approximation

$NP \subseteq PCP(O(\log n), O(1)) \Leftrightarrow \exists s < 1$ s.t. GAP-3SAT$_s$ is NP-hard

MAX-3SAT is hard to Approximate (2/2)

PCP $\Leftrightarrow$ Hardness of Approximation

$NP \subseteq PCP(O(\log n), O(1)) \Leftrightarrow \exists s < 1$ s.t. GAP-3SAT$_s$ is NP-hard

$\Leftarrow$ roughly, prove that GAP-3SAT$_s \in PCP(O(\log n), O(1))$

For any $L \in NP$ there is Φ polynomial-time s.t.

- for any word w, $\Phi(w)$ is an instance of GAP-3SAT$_s$
- if $w \in L$ then $\Phi(w)$ satisfiable; else at most a fraction s of the clauses can be satisfied

Verifier:

1. Compute $\Phi(w)$, m its number of clauses
2. use $\lceil \log m \rceil$ random bits to choose a clause
3. read the value of the litterals in this clause
4. answer 1 if the clause is satisfied, 0 otherwise

If $w \notin L$ (i.e., at most $s.m$ clauses can be satisfied), it is detected with proba $\geq 1 - s$.
Repeat constant number of times to achieve the desired probability

MAX-3SAT is hard to Approximate (2/2)

PCP $\Leftrightarrow$ Hardness of Approximation

$NP \subseteq PCP(O(\log n), O(1)) \Leftrightarrow \exists s < 1$ s.t. GAP-3SAT$_s$ is NP-hard

$\Rightarrow$ reduce $L \in NP$ to GAP-kSAT$_{1-\frac{1}{2^{k+1}}}$

L has a Verifier V using $O(\log n)$ random bits and read $k = O(1)$ bits in the proof.
given a word w and a random string ρ, $V(w, \rho, x) \in \{0,1\}$ reads k bits of the proof x

$f_{w,\rho} : x \rightarrow V(w, \rho, x)$ expressed as k-SAT formula with $\leq 2^k$ clauses.

$$Let\ \Phi_w = \wedge_{\rho \in \{0,1\}^{O(\log n)}} f_{w,\rho}.$$

if $w \in L$ then Φ_w satisfied by x
otherwise at most a fraction $1 - \frac{1}{2^{k+1}}$ of clauses can be satisfied.

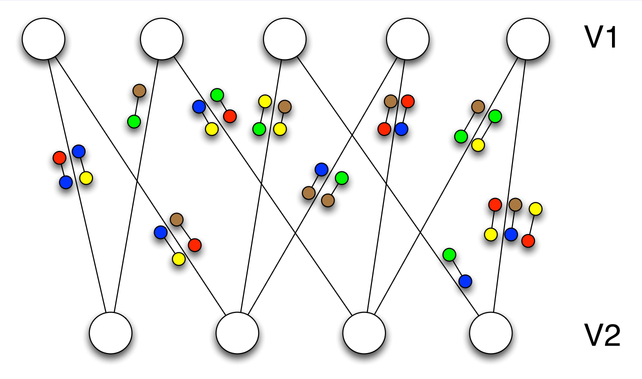
Outline

1. Introduction on Approximation Algorithms
2. PCP & PCP Theorem ($NP = PCP(O(\log n), O(1))$)
3. PCP Theorem $\Leftrightarrow$ MAX-3SAT is hard to Approximate
4. **From Label-Cover to MAX-Clique**

LABEL COVER (Maximization version)

regular bipartite graph $G = (V_1 \cup V_2, E)$ and, $\forall e \in E$, partial function

$$\Pi_e : [1, N] \to [1, N]$$



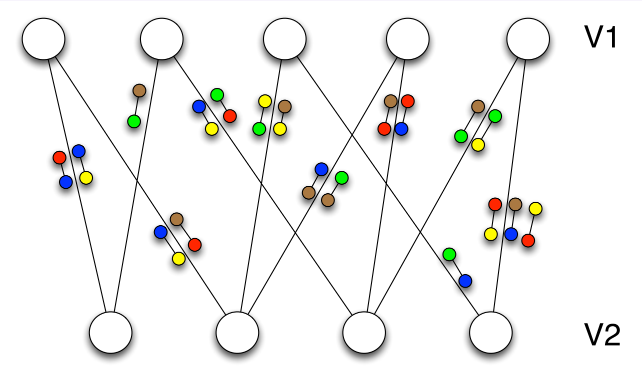
Let $c : V \to [1, N]$ be a coloring of V.
an edge $e = \{u, v\} \in E$, $u \in V_1, v \in V_2$, is covered if $\Pi_e(c(u)) = c(v)$.

Goal: Find a coloring maximizing the fraction of covered edges.

LABEL COVER (Maximization version)

regular bipartite graph $G = (V_1 \cup V_2, E)$ and, $\forall e \in E$, partial function

$$\Pi_e : [1, N] \to [1, N]$$



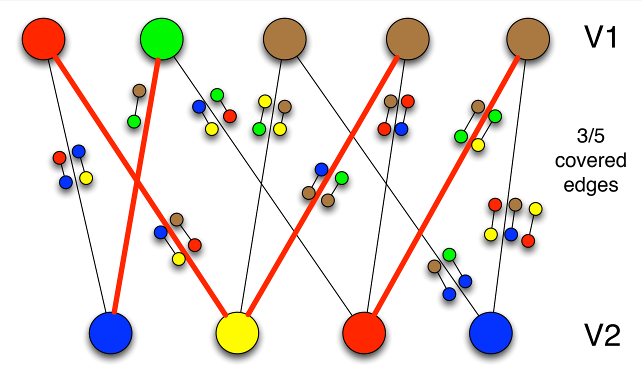
Let $c : V \to [1, N]$ be a coloring of V.
an edge $e = \{u, v\} \in E$, $u \in V_1, v \in V_2$, is covered if $\Pi_e(c(u)) = c(v)$.

Goal: Find a coloring maximizing the fraction of covered edges.

LABEL COVER (Maximization version)

regular bipartite graph $G = (V_1 \cup V_2, E)$ and, $\forall e \in E$, partial function

$$\Pi_e : [1, N] \rightarrow [1, N]$$



Let $c : V \rightarrow [1, N]$ be a coloring of V.
an edge $e = \{u, v\} \in E$, $u \in V_1, v \in V_2$, is covered if $\Pi_e(c(u)) = c(v)$.

Goal: Find a coloring maximizing the fraction of covered edges.

LABEL COVER hard to approximate

$\exists$ reduction f from 3SAT to LABEL COVER s.t.

for any instance I of 3SAT and instance $f(I)$ of LABEL COVER

if X clauses of I can be satisfied, $\geq X$ edges of $f(I)$ can be covered
if Y edges of $f(I)$ can be covered, $\geq Y/3$ clauses of I can be satisfied

Gap-Preserving Reduction

Corollary

$\exists \epsilon$ s.t. finding an $(1+\epsilon)$-approximation to Label Cover is NP-hard.

LABEL COVER very hard to approximate

Theorem

$\forall 0 < \epsilon < 1/2$ and $c(n) \geq 2^{\log^{0,5-\epsilon} n}$, and there is a reduction f in time $n^{poly(\log n)}$, from any NP problem Π to Label Cover such that

- for any $w \in \Pi$, $f(w)$ has value 1
- for any $w \notin \Pi$, $f(w)$ has value at most $1/c(n)$.

Corollary: if $\exists 0 < \epsilon < 1/2$ and a $2^{\log^{0,5-\epsilon} n}$-approximation algorithm for Label Cover, then any problem in NP can be solved in time $n^{poly(\log n)}$.

Conjecture: Exponential Time Hypothesis (ETH)

There is no sub-exponential-time algorithm to solve 3-SAT unless $P = NP$

Corollary 1: Assuming ETH

$\forall \epsilon$, there is no $2^{\log^{0,5-\epsilon} n}$-approximation algorithm for Label Cover unless $P = NP$

Corollary 2: Assuming ETH

$\forall \epsilon$, there is no $2^{\log^{0,5-\epsilon} n}$-approximation algorithm for Max Clique unless $P = NP$

LABEL COVER very hard to approximate

Theorem

$\forall 0 < \epsilon < 1/2$ and $c(n) \geq 2^{\log^{0,5-\epsilon} n}$, and there is a reduction f in time $n^{poly(\log n)}$, from any NP problem Π to Label Cover such that

- for any $w \in \Pi$, $f(w)$ has value 1
- for any $w \notin \Pi$, $f(w)$ has value at most $1/c(n)$.

Corollary: if $\exists 0 < \epsilon < 1/2$ and a $2^{\log^{0,5-\epsilon} n}$-approximation algorithm for Label Cover, then any problem in NP can be solved in time $n^{poly(\log n)}$.

Conjecture: Exponential Time Hypothesis (ETH)

There is no sub-exponential-time algorithm to solve 3-SAT unless $P = NP$

Corollary 1: Assuming ETH

$\forall \epsilon$, there is no $2^{\log^{0,5-\epsilon} n}$-approximation algorithm for Label Cover unless $P = NP$

Corollary 2: Assuming ETH

$\forall \epsilon$, there is no $2^{\log^{0,5-\epsilon} n}$-approximation algorithm for Max Clique unless $P = NP$

LABEL COVER very hard to approximate

Theorem

$\forall 0 < \epsilon < 1/2$ and $c(n) \geq 2^{\log^{0,5-\epsilon} n}$, and there is a reduction f in time $n^{poly(\log n)}$, from any NP problem Π to Label Cover such that

- for any $w \in \Pi$, $f(w)$ has value 1
- for any $w \notin \Pi$, $f(w)$ has value at most $1/c(n)$.

Corollary: if $\exists 0 < \epsilon < 1/2$ and a $2^{\log^{0,5-\epsilon} n}$-approximation algorithm for Label Cover, then any problem in NP can be solved in time $n^{poly(\log n)}$.

Conjecture: Exponential Time Hypothesis (ETH)

There is no sub-exponential-time algorithm to solve 3-SAT unless $P = NP$

Corollary 1: Assuming ETH

$\forall \epsilon$, there is no $2^{\log^{0,5-\epsilon} n}$-approximation algorithm for Label Cover unless $P = NP$

Corollary 2: Assuming ETH

$\forall \epsilon$, there is no $2^{\log^{0,5-\epsilon} n}$-approximation algorithm for Max Clique unless $P = NP$

Proof of Cor. 2: "Max Clique hard to approx."

Gap preserving reduction from Label Cover
Reduce any instance of Label Cover to instance of Max Clique s.t. both optima coincide

Let $G = (V_1 \cup V_2, E)$ and, $\forall e \in E$, $\Pi_e : [1, N] \to [1, N]$

- for any $e \in E$ and any $a, b \in [1, N]$ s.t., $\Pi_e(a) = b$
  add a vertex (e, a, b)
- 2 nodes are adjacent if they are consistent
  e.g., (uv, a, b) and (uw, a', b') are adjacent iff $a = a'$

Proof of Theorem

$L \in RPCP(r, s, p)$ iff $\exists$ a restricted Verifier V s.t.

- V polynomial-time in $|w|$, uses a string ρ of $O(r(|w|))$ bits of randomness
  - the proof uses an alphabet with $2^{O(s(|w|))}$ symbols
  - the proof consists of 2 Tables T_1, T_2
  - V chooses 1 location (chosen uniformly at random) in each table, and read only the corresponding 2 symbols a_1 and a_2
  - V confirms that T_1 coherent with T_2: for any a_1 at most one symbol a_2 makes V accepting
    - if $w \in L$ then $\exists$ proof x with $Pr_\rho(V(w, x, \rho) = 1) = 1$ (**completeness**).
    - if $w \notin L$ then $\forall$ proof x, $Pr_\rho(V(w, x, \rho) = 1) \leq 2^{-p(|w|)}$ (**soundness**).

Theorem [Feige,Lovász'92] improved by Raz'95

For any integer $k \geq 2$,

$$NP \subseteq RPCP(\log^{2k+2} n, \log^{k+2} n, \log^k n)$$

For any $L \in NP$, use a restricted PCP-Verifier of it to build an instance of Label Cover.

References

- Sanjeev Arora, Probabilistic Checking of Proofs and Hardness of Approximation Problems, Ph.D. thesis, 1994
- Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, Mario Szegedy: Proof Verification and the Hardness of Approximation Problems. J. ACM 45(3): 501-555 (1998)
- Irit Dinur: The PCP theorem by gap amplification. J. ACM 54(3): 12 (2007)
- Johan Hastad: Some optimal inapproximability results. J. ACM 48(4): 798-859 (2001)
- Ran Raz: A Parallel Repetition Theorem. SIAM J. Comput. 27(3): 763-803 (1998)

lecture of Nicolas Schabanel on proof PCP theorem (video online)

lecture of Luca Trevisan on PCP (video online)