Lecture 18: PCP Theorem and Hardness of Approximation I

Arijit Bishnu

26.04.2010

Outline

Outline

Approximation Algorithm

Approximation Ratio: Minimization Problems

An algorithm is r-approximate for a minimization problem if, for every input, the algorithm finds a solution whose cost is at most r times the optimum. $r \geq 1$ is called the approximation ratio of the algorithm.

Approximation Ratio: Maximization Problems

An algorithm is r-approximate for a maximization problem if, for every input, the algorithm finds a solution whose cost is at most $1/r$ times the optimum. $r \geq 1$ is called the approximation ratio of the algorithm.

Approximation Ratio: PTAS

A PTAS is an r-approximate algorithm for every $r > 1$.

Approximation Algorithm for Max-3SAT

The Problem: Max-3SAT

Max-3SAT is the problem of finding, given a 3CNF Boolean formula φ as input, an assignment that maximizes the number of satisfied clauses.

Approximation Algorithm for Max-3SAT

The Problem: Max-3SAT

Max-3SAT is the problem of finding, given a 3CNF Boolean formula φ as input, an assignment that maximizes the number of satisfied clauses.

Approximation Algorithm

Approximation Algorithm for Max-3SAT

The Problem: Max-3SAT

Max-3SAT is the problem of finding, given a 3CNF Boolean formula φ as input, an assignment that maximizes the number of satisfied clauses.

Approximation Algorithm

- Assign to the i-th variable, starting from the first, a boolean value that satisfies at least half of the clauses in which it appears. Remove the satisfied clauses and continue.

Approximation Algorithm for Max-3SAT

The Problem: Max-3SAT

Max-3SAT is the problem of finding, given a 3CNF Boolean formula φ as input, an assignment that maximizes the number of satisfied clauses.

Approximation Algorithm

- Assign to the i-th variable, starting from the first, a boolean value that satisfies at least half of the clauses in which it appears. Remove the satisfied clauses and continue.
- Easy to see that it is a 2-factor approximation.

A Relook at Cook-Levin Reduction

3SAT

- What is the problem of 3SAT?

A Relook at Cook-Levin Reduction

3SAT

- What is the problem of 3SAT?
- Given any CNF expression φ, we want to determine whether $\exists$ a satisfying assignment to φ, i.e., whether $\varphi \in$ 3SAT.

A Relook at Cook-Levin Reduction

3SAT

- What is the problem of 3SAT?
- Given any CNF expression φ, we want to determine whether $\exists$ a satisfying assignment to φ, i.e., whether $\varphi \in$ 3SAT.
- Another interpretation is that we want to distinguish between satisfiable and unsatisfiable instances.

A Relook at Cook-Levin Reduction

Another Problem: Vertex Cover (VC)

Given an undirected graph $G = (V, E)$ and an integer $k > 0$, is there a subset $V' \subseteq V$ of size at most k such that each edge in E is incident to at least one vertex in V'?

A Relook at Cook-Levin Reduction

Polynomial Time (Cook) Reduction: 3SAT $\leq_P$ VC.

A Relook at Cook-Levin Reduction

Polynomial Time (Cook) Reduction: 3SAT $\leq_P$ VC.

- Start from a 3SAT instance φ and construct a graph G and an integer $k > 0$ such that if φ is satisfiable then G has a vertex cover of size k.

A Relook at Cook-Levin Reduction

Polynomial Time (Cook) Reduction: 3SAT $\leq_P$ VC.

- Start from a 3SAT instance φ and construct a graph G and an integer $k > 0$ such that if φ is satisfiable then G has a vertex cover of size k.
- If φ is unsatisfiable then all vertex covers in G have size at least $k + 1$.

A Relook at Cook-Levin Reduction

Polynomial Time (Cook) Reduction: 3SAT $\leq_P$ VC.

- Start from a 3SAT instance φ and construct a graph G and an integer $k > 0$ such that if φ is satisfiable then G has a vertex cover of size k.
- If φ is unsatisfiable then all vertex covers in G have size at least $k + 1$.
- Consider the question of approximability of VC and the Cook Reduction .

A Relook at Cook-Levin Reduction

Polynomial Time (Cook) Reduction: 3SAT $\leq_P$ VC.

- Start from a 3SAT instance φ and construct a graph G and an integer $k > 0$ such that if φ is satisfiable then G has a vertex cover of size k.
- If φ is unsatisfiable then all vertex covers in G have size at least $k + 1$.
- Consider the question of approximability of VC and the Cook Reduction .
- We observe that if φ has an assignment that satisfies all the clauses excepting one (or c), then G has a vertex cover of size $k - 2$ $(k - 2c)$.

A Relook at Cook-Levin Reduction

Polynomial Time (Cook) Reduction: 3SAT $\leq_P$ VC.

- Start from a 3SAT instance φ and construct a graph G and an integer $k > 0$ such that if φ is satisfiable then G has a vertex cover of size k.
- If φ is unsatisfiable then all vertex covers in G have size at least $k + 1$.
- Consider the question of approximability of VC and the Cook Reduction .
- We observe that if φ has an assignment that satisfies all the clauses excepting one (or c), then G has a vertex cover of size $k - 2$ $(k - 2c)$.
- The instances that we get by such reductions are easy to approximate.

A Relook at Cook-Levin Reduction

Polynomial Time (Cook) Reduction: 3SAT $\leq_P$ VC.

- Start from a 3SAT instance φ and construct a graph G and an integer $k > 0$ such that if φ is satisfiable then G has a vertex cover of size k.
- If φ is unsatisfiable then all vertex covers in G have size at least $k + 1$.
- Consider the question of approximability of VC and the Cook Reduction .
- We observe that if φ has an assignment that satisfies all the clauses excepting one (or c), then G has a vertex cover of size $k - 2$ $(k - 2c)$.
- The instances that we get by such reductions are easy to approximate.
- The "gap" between satisfiability and unsatisfiability is "small".

Inapproximability: Hardness of Aproximation

- For finding hardness of approximation ratio of VC, the PCP theorem is used to show the following poly-time reduction. It maps an instance φ of SAT to a graph $G = (V, E)$ such that

Inapproximability: Hardness of Aproximation

- For finding hardness of approximation ratio of VC, the PCP theorem is used to show the following poly-time reduction. It maps an instance φ of SAT to a graph $G = (V, E)$ such that
  - if φ is satisfiable, G has a vertex cover of size $\leq \frac{2}{3}|V|$.

Inapproximability: Hardness of Aproximation

- For finding hardness of approximation ratio of VC, the PCP theorem is used to show the following poly-time reduction. It maps an instance φ of SAT to a graph $G = (V, E)$ such that
  - if φ is satisfiable, G has a vertex cover of size $\leq \frac{2}{3}|V|$.
  - if φ is not satisfiable, G's smallest vertex cover is of size $> \alpha \cdot \frac{2}{3}|V|$, where $\alpha > 1$ is a fixed constant.

Inapproximability: Hardness of Aproximation

- For finding hardness of approximation ratio of VC, the PCP theorem is used to show the following poly-time reduction. It maps an instance φ of SAT to a graph $G = (V, E)$ such that
  - if φ is satisfiable, G has a vertex cover of size $\leq \frac{2}{3}|V|$.
  - if φ is not satisfiable, G's smallest vertex cover is of size $> \alpha \cdot \frac{2}{3}|V|$, where $\alpha > 1$ is a fixed constant.

Claim

If the above reduction is doable, then there is no polynomial time algorithm for vertex cover that achieves an approximation guarantee of α, assuming P $\neq$ NP.

Inapproximability: Hardness of Aproximation

Proof

Inapproximability: Hardness of Aproximation

Proof

- An approximation algorithm for VC, having an approximation factor of α or better, will find a cover of size $\leq \alpha \cdot \frac{2}{3} |V|$ when given a graph from the first class.

Inapproximability: Hardness of Aproximation

Proof

- An approximation algorithm for VC, having an approximation factor of α or better, will find a cover of size $\leq \alpha \cdot \frac{2}{3} |V|$ when given a graph from the first class.
- Thus, the approximation algorithm will be able to distinguish between the two classes of graphs, leading to a contradiction.

Inapproximability: Hardness of Aproximation

Proof

- An approximation algorithm for VC, having an approximation factor of α or better, will find a cover of size $\leq \alpha \cdot \frac{2}{3} |V|$ when given a graph from the first class.
- Thus, the approximation algorithm will be able to distinguish between the two classes of graphs, leading to a contradiction.

- The above reduction introduces a gap of factor α in the optimal objective function value achieved by the two classes of graphs. If $\alpha = 1$, then it is the original vertex cover problem.

Inapproximability: Hardness of Aproximation

Proof

- An approximation algorithm for VC, having an approximation factor of α or better, will find a cover of size $\leq \alpha \cdot \frac{2}{3} |V|$ when given a graph from the first class.
- Thus, the approximation algorithm will be able to distinguish between the two classes of graphs, leading to a contradiction.

- The above reduction introduces a gap of factor α in the optimal objective function value achieved by the two classes of graphs. If $\alpha = 1$, then it is the original vertex cover problem.
- To prove inapproximability results, we need to find a machine model for NP in which accepting computations would be "far" from rejecting computations.

Outline

An Alternate View of NP

- The PCP theorem signifies a new proof system.

An Alternate View of NP

- The PCP theorem signifies a new proof system.
- For problems in NP, once a certificate is given, the entire certificate is read and checked for its membership.

An Alternate View of NP

- The PCP theorem signifies a new proof system.
- For problems in NP, once a certificate is given, the entire certificate is read and checked for its membership.
- PCP theorem shows that the certificate can be rewritten so that it can be verified by probabilistically selecting a constant number of locations to examine it.

An Alternate View of NP

- The PCP theorem signifies a new proof system.
- For problems in NP, once a certificate is given, the entire certificate is read and checked for its membership.
- PCP theorem shows that the certificate can be rewritten so that it can be verified by probabilistically selecting a constant number of locations to examine it.
- This verification satisfies the following condition:

An Alternate View of NP

- The PCP theorem signifies a new proof system.
- For problems in NP, once a certificate is given, the entire certificate is read and checked for its membership.
- PCP theorem shows that the certificate can be rewritten so that it can be verified by probabilistically selecting a constant number of locations to examine it.
- This verification satisfies the following condition:
  - a correct certificate is always accepted for membership whatever be the random choices of bits of the certificate to be verified.

An Alternate View of NP

- The PCP theorem signifies a new proof system.
- For problems in NP, once a certificate is given, the entire certificate is read and checked for its membership.
- PCP theorem shows that the certificate can be rewritten so that it can be verified by probabilistically selecting a constant number of locations to examine it.
- This verification satisfies the following condition:
  - a correct certificate is always accepted for membership whatever be the random choices of bits of the certificate to be verified.
  - every certificate that is a wrong one is rejected with a high probability.

A Relook at NP

Definition: The Class NP

A language $L \subseteq \{0,1\}^*$ is in NP if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM V s.t. for every $x \in \{0,1\}^*$,

$$x \in L \iff \exists \pi \in \{0,1\}^{p(|x|)} \text{ such that } V(x,\pi) = 1$$

A Relook at NP

Definition: The Class NP

A language $L \subseteq \{0,1\}^*$ is in NP if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM V s.t. for every $x \in \{0,1\}^*$,

$$x \in L \iff \exists \pi \in \{0,1\}^{p(|x|)} \text{ such that } V(x,\pi) = 1$$

Recasting the Definition of NP

A language $L \subseteq \{0,1\}^*$ is in NP if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and a polynomial-time TM V s.t. for every $x \in \{0,1\}^*$,

$$\begin{aligned} x \in L &\Rightarrow \exists \pi \in \{0,1\}^{p(|x|)} \text{ such that } V^{\pi}(x) = 1 \\ x \notin L &\Rightarrow \forall u \in \{0,1\}^{p(|x|)} \text{ such that } V^{\pi}(x) = 0 \end{aligned}$$

Here, V^{π} denotes a verifier with access to certificate π which is the proof string.

A Relook at NP

Changes to the Verifier

A Relook at NP

Changes to the Verifier

- Make the verifier probabilistic.

A Relook at NP

Changes to the Verifier

- Make the verifier probabilistic.
- The verifier has random access to the proof string π.

A Relook at NP

Changes to the Verifier

- Make the verifier probabilistic.
- The verifier has random access to the proof string π.
- Each bit of the proof string can be independently queried by the verifier via a special address tape. The verifier writes i on the address tape to get the bit $\pi[i]$.

A Relook at NP

Changes to the Verifier

- Make the verifier probabilistic.
- The verifier has random access to the proof string π.
- Each bit of the proof string can be independently queried by the verifier via a special address tape. The verifier writes i on the address tape to get the bit $\pi[i]$.
- How big can the proof string π be?

A Relook at NP

Changes to the Verifier

- Make the verifier probabilistic.
- The verifier has random access to the proof string π.
- Each bit of the proof string can be independently queried by the verifier via a special address tape. The verifier writes i on the address tape to get the bit $\pi[i]$.
- How big can the proof string π be?
- As the address size is logarithmic in the proof size, a poly-time verifier can check membership proofs of exponential size.

PCP Verifier

Definition: PCP verifer

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

- Efficiency: On an input string $x \in \{0,1\}^n$ and given random access to a string $\pi \in \{0,1\}^*$ of length at most $q(n)2^{r(n)}$ (the proof), V uses at most $r(n)$ random coins and makes at most $q(n)$ queries, that do not depend upon the responses to any prior queries, to locations of π. Then, it outputs 1 or 0. $V^{\pi}(x)$ denotes the r.v. representing V's output on input x and with random access to π.

PCP Verifier

Definition: PCP verifer

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

- Efficiency: On an input string $x \in \{0,1\}^n$ and given random access to a string $\pi \in \{0,1\}^*$ of length at most $q(n)2^{r(n)}$ (the proof), V uses at most $r(n)$ random coins and makes at most $q(n)$ queries, that do not depend upon the responses to any prior queries, to locations of π. Then, it outputs 1 or 0. $V^\pi(x)$ denotes the r.v. representing V's output on input x and with random access to π.
- Completeness: If $x \in L$, then there exists a proof $\pi \in \{0,1\}^*$ such that $\Pr[V^\pi(x) = 1] = 1$. This string π is the correct proof for x.

PCP Verifier

Definition: PCP verifer

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

- Efficiency: On an input string $x \in \{0,1\}^n$ and given random access to a string $\pi \in \{0,1\}^*$ of length at most $q(n)2^{r(n)}$ (the proof), V uses at most $r(n)$ random coins and makes at most $q(n)$ queries, that do not depend upon the responses to any prior queries, to locations of π. Then, it outputs 1 or 0. $V^\pi(x)$ denotes the r.v. representing V's output on input x and with random access to π.
- Completeness: If $x \in L$, then there exists a proof $\pi \in \{0,1\}^*$ such that $\Pr[V^\pi(x) = 1] = 1$. This string π is the correct proof for x.
- Soundness: If $x \notin L$, then for every proof $\pi \in \{0,1\}^*$, $\Pr[V^\pi(x) = 1] \leq \frac{1}{2}$.

  A language $L \in \mathrm{PCP}(r(n), q(n))$ if there are some constants $c, d > 0$ such that L has a $(c \cdot r(n), d \cdot q(n)) - \mathrm{PCP}$ verifier.

Some Relations

Lemma

$\mathsf{PCP}(r(n), q(n)) \subseteq \mathsf{NTIME}(2^{O(r(n))} q(n))$.

Some Relations

Lemma

$\mathrm{PCP}(r(n), q(n)) \subseteq \mathrm{NTIME}(2^{O(r(n))} q(n))$.

Proof

Some Relations

Lemma

$\mathrm{PCP}(r(n), q(n)) \subseteq \mathrm{NTIME}(2^{O(r(n))} q(n))$.

Proof

- A NDTM can guess the proof in $2^{O(r(n))} q(n)$ time.

Some Relations

Lemma

$\mathrm{PCP}(r(n), q(n)) \subseteq \mathrm{NTIME}(2^{O(r(n))} q(n))$.

Proof

- A NDTM can guess the proof in $2^{O(r(n))} q(n)$ time.
- For verification, deterministically run the verifier for all $2^{O(r(n))}$ possible choices of its random coin tosses.

Some Relations

Lemma

$\mathrm{PCP}(r(n), q(n)) \subseteq \mathrm{NTIME}(2^{O(r(n))} q(n))$.

Proof

- A NDTM can guess the proof in $2^{O(r(n))} q(n)$ time.
- For verification, deterministically run the verifier for all $2^{O(r(n))}$ possible choices of its random coin tosses.
- If the verifier accepts for all these possible coin tosses, then the NDTM accepts.

Some Relations

Lemma

$\mathrm{PCP}(r(n), q(n)) \subseteq \mathrm{NTIME}(2^{O(r(n))} q(n))$.

Proof

- A NDTM can guess the proof in $2^{O(r(n))} q(n)$ time.
- For verification, deterministically run the verifier for all $2^{O(r(n))}$ possible choices of its random coin tosses.
- If the verifier accepts for all these possible coin tosses, then the NDTM accepts.

Corollary

$\mathrm{PCP}(\log n, 1) \subseteq \mathrm{NTIME}(2^{O(\log n)}) = \mathrm{NP}$.

The PCP Theorem

PCP Theorem

$\mathrm{PCP}(\log n, 1) = \mathrm{NP}$

The PCP Theorem

PCP Theorem

$\mathrm{PCP}(\log n, 1) = \mathrm{NP}$

The other direction

The crux of the proof of PCP theorem is to show $\mathrm{NP} \subseteq \mathrm{PCP}(\log n, 1)$.