Approximation Algorithms (ADM III)
7- Hardness of Approximation

Guillaume Sagnol

Technische Universität Berlin

Outline

Reduction from an NP-complete problem

Assume we can reduce an NP-complete problem Π into a set of instances of a minimization problem, such that

$$\Pi \text{ Yes-Instance} \iff OPT \leq a$$
$$\Pi \text{ No-Instance} \iff OPT \geq b$$

Reduction from an NP-complete problem

Assume we can reduce an NP-complete problem Π into a set of instances of a minimization problem, such that

$$\Pi \text{ Yes-Instance} \iff OPT \leq a$$
$$\Pi \text{ No-Instance} \iff OPT \geq b$$

Then, the existence of an approximation algorithm with performance guarantee strictly better than $\frac{b}{a}$ implies $P = NP$.

Reduction from an NP-complete problem

Assume we can reduce an NP-complete problem Π into a set of instances of a minimization problem, such that

$$\Pi \text{ Yes-Instance} \iff OPT \leq a$$
$$\Pi \text{ No-Instance} \iff OPT \geq b$$

Then, the existence of an approximation algorithm with performance guarantee strictly better than $\frac{b}{a}$ implies $P = NP$.

We already encountered this idea to show hardness-of-approximation results: Unless, P=NP, the best approximation ratio is bounded by

- 2 for k-`center` (reduction from `Dominating set`)
- $3/2$ for `Bin-Packing` (reduction from `Partition`)
- $O(2^n)$ for the (non-metric) `TSP` (reduction from `Hamiltonian Cycle`)
- $4/3$ for edge-coloring (reduction from `3-coloring the edges of a graph with node degrees at most 3`)

Hardness of approximation of $R||C_{\max}$

Remember that in the unrelated parallel machines environment, denoted by “R”, the processing time of job j depends on the machine i on which it is executed.

Hardness of approximation of $R||C_{\max}$

Remember that in the unrelated parallel machines environment, denoted by "R", the processing time of job j depends on the machine i on which it is executed.

Definition 7.1 (Scheduling on unrelated machines)

Given some $p_{ij} \geq 0, \forall j \in [n], \forall i \in [m]$, the problem $R||C_{\max}$ asks to assign each job j to a machine $i \in [n]$, in order to minimize the quantity $C_{\max} = \max_{i \in [m]} \sum_{j \in J_i} p_{ij}$, where $J_i \subseteq [n]$ is the subset of jobs assigned to i.

Hardness of approximation of $R||C_{max}$

Definition 7.2 (3-dimensional matching)

Given: A, B, C, 3 disjoint sets of n elements, along with a family of m triples of the form $T_k = (a_{i_k}, b_{j_k}, c_{\ell_k}) \in A \times B \times C$ with one element from each of A, B, and C.
The `3-dimensional matching` problem asks whether there exists a subset of n triples covering all $3n$ elements of $A \cup B \cup C$.

Hardness of approximation of $R||C_{\max}$

Definition 7.2 (3-dimensional matching)

Given: A, B, C, 3 disjoint sets of n elements, along with a family of m triples of the form $T_k = (a_{i_k}, b_{j_k}, c_{\ell_k}) \in A \times B \times C$ with one element from each of A, B, and C.
The `3-dimensional matching` problem asks whether there exists a subset of n triples covering all $3n$ elements of $A \cup B \cup C$.

Theorem 7.3

It is NP-complete to decide whether there exists a schedule of length at most 3, given an input of $R||C_{\max}$ where each $p_{ij} \in \{1, 3\}$.

Corollary 7.4

There is no α-approximation algorithm with $\alpha < 4/3$ for $R||C_{\max}$, unless $P = NP$.

Hardness of approximation of $R||C_{\max}$

Definition 7.2 (3-dimensional matching)

Given: A, B, C, 3 disjoint sets of n elements, along with a family of m triples of the form $T_k = (a_{i_k}, b_{j_k}, c_{\ell_k}) \in A \times B \times C$ with one element from each of A, B, and C.
The `3-dimensional matching` problem asks whether there exists a subset of n triples covering all $3n$ elements of $A \cup B \cup C$.

Theorem 7.3 (Stronger version)

It is NP-complete to decide whether there exists a schedule of length at most 2, given an input of $R||C_{\max}$ where each $p_{ij} \in \{1, 2, 3\}$.

Corollary 7.4 (Stronger version)

There is no α-approximation algorithm with $\alpha < 3/2$ for $R||C_{\max}$, unless $P = NP$.

Hardness of approximation for edge-disjoint paths

Given: directed graph $G = (V, E)$ with k source-sink pairs $s_i, t_i \in V$.

Goal: find a subset of $S \subseteq \{1, \ldots, k\}$ of maximum cardinality, together with a path P_i for each $i \in S$, and for any $i, j \in S$, $i \neq j$, $P_i \cap P_j = \emptyset$.

Hardness of approximation for edge-disjoint paths

Given: directed graph $G = (V, E)$ with k source-sink pairs $s_i, t_i \in V$.

Goal: find a subset of $S \subseteq \{1, \ldots, k\}$ of maximum cardinality, together with a path P_i for each $i \in S$, and for any $i, j \in S$, $i \neq j$, $P_i \cap P_j = \emptyset$.

We will use the following claim, without proving it:
When $k = 2$, it is NP-complete to decide whether there exists 2 edge-disjoint paths from s_1 to t_1 and s_2 to t_2.

Hardness of approximation for edge-disjoint paths

Given: directed graph $G = (V, E)$ with k source-sink pairs $s_i, t_i \in V$.

Goal: find a subset of $S \subseteq \{1, \ldots, k\}$ of maximum cardinality, together with a path P_i for each $i \in S$, and for any $i, j \in S$, $i \neq j$, $P_i \cap P_j = \emptyset$.

We will use the following claim, without proving it:
When $k = 2$, it is NP-complete to decide whether there exists 2 edge-disjoint paths from s_1 to t_1 and s_2 to t_2.

Corollary 7.5

There is no $\alpha-$approximation algorithm with $\alpha > \frac{1}{2}$ for the edge disjoint paths problem, unless $P = NP$.

Hardness of approximation for edge-disjoint paths

Given: directed graph $G = (V, E)$ with k source-sink pairs $s_i, t_i \in V$.

Goal: find a subset of $S \subseteq \{1, \ldots, k\}$ of maximum cardinality, together with a path P_i for each $i \in S$, and for any $i, j \in S$, $i \neq j$, $P_i \cap P_j = \emptyset$.

We will use the following claim, without proving it:
When $k = 2$, it is NP-complete to decide whether there exists 2 edge-disjoint paths from s_1 to t_1 and s_2 to t_2.

Corollary 7.5

There is no $\alpha-$approximation algorithm with $\alpha > \dfrac{1}{2}$ for the edge disjoint paths problem, unless $P = NP$.

Corollary 7.6

For any $\epsilon > 0$, there is no $\Omega(m^{-\frac{1}{2}+\epsilon})-$approximation for the edge disjoint paths problem, unless $P = NP$.

Outline

Approximation-Preserving Reductions

It is sometimes possible to construct a reduction showing that if there exists an α-approximation-algorithm for $\mathcal{P}'$, then an $f(\alpha)-$approximation algorithm for $\mathcal{P}$ can be constructed.

Approximation-Preserving Reductions

It is sometimes possible to construct a reduction showing that if there exists an α-approximation-algorithm for $\mathcal{P}'$, then an $f(\alpha)-$approximation algorithm for $\mathcal{P}$ can be constructed.

Then, if we know it is hard to approximate $\mathcal{P}$ within a factor $f(\alpha)$, we deduce it is hard to approximate $\mathcal{P}'$ within α.

Reduction from MAX E 3SAT to MAX 2SAT

Recall that MAX 2SAT is the maximum satisfiability problem reduced to clauses with at most 2 literals, while MAX E 3SAT has clauses with exactly 3 literals.

Reduction from MAX E 3SAT to MAX 2SAT

Recall that MAX 2SAT is the maximum satisfiability problem reduced to clauses with at most 2 literals, while MAX E 3SAT has clauses with exactly 3 literals.

- Consider an instance I of MAX E 3SAT, and assume that the jth clause is of the form $x_1 \vee x_2 \vee x_3$

Reduction from MAX E 3SAT to MAX 2SAT

Recall that MAX 2SAT is the maximum satisfiability problem reduced to clauses with at most 2 literals, while MAX E 3SAT has clauses with exactly 3 literals.

- Consider an instance I of MAX E 3SAT, and assume that the jth clause is of the form $x_1 \vee x_2 \vee x_3$
- We create an instance I' of MAX 2SAT by replacing C_j with the following 8 clauses, involving the new variable y_j:

$$x_1 \vee x_3 \quad \bar{x_1} \vee \bar{x_3} \quad x_1 \vee \bar{y_j} \quad \bar{x_1} \vee y_j \quad x_3 \vee \bar{y_j} \quad \bar{x_3} \vee y_j \quad x_2 \vee y_j \quad x_2 \vee y_j$$

Reduction from MAX E 3SAT to MAX 2SAT

Recall that MAX 2SAT is the maximum satisfiability problem reduced to clauses with at most 2 literals, while MAX E 3SAT has clauses with exactly 3 literals.

- Consider an instance I of MAX E 3SAT, and assume that the jth clause is of the form $x_1 \vee x_2 \vee x_3$
- We create an instance I' of MAX 2SAT by replacing C_j with the following 8 clauses, involving the new variable y_j:

$$x_1 \vee x_3 \quad \bar{x_1} \vee \bar{x_3} \quad x_1 \vee \bar{y_j} \quad \bar{x_1} \vee y_j \quad x_3 \vee \bar{y_j} \quad \bar{x_3} \vee y_j \quad x_2 \vee y_j \quad x_2 \vee y_j$$

Number of satisfied clauses, for each assignment of x_1, x_2, x_3, y_j:

x_1	x_2	x_3	$y_j = 0$	$y_j = 1$
0	0	0	5	5
0	0	1	5	7
0	1	0	7	5
0	1	1	7	7
1	0	0	5	7
1	0	1	3	7
1	1	0	7	7
1	1	1	5	7

Reduction from MAX E 3SAT to MAX 2SAT

Observation

- For any assignment of the variables x_1, x_2, x_3

$$C_j \text{ satisfied } \iff \exists y_j : \text{7 clauses of this group satisfied in } I'$$
$$\neg C_j \text{ satisfied } \iff \forall y_j, \text{ 5 clauses of this group satisfied in } I'$$

Reduction from MAX E 3SAT to MAX 2SAT

Observation

- For any assignment of the variables x_1, x_2, x_3

$$C_j \text{ satisfied } \iff \exists y_j : 7 \text{ clauses of this group satisfied in } I'$$

$$\neg C_j \text{ satisfied } \iff \forall y_j, \; 5 \text{ clauses of this group satisfied in } I'$$

- Moreover, we know that $\texttt{OPT}(I) \geq 7/8 \cdot m$.

Reduction from MAX E 3SAT to MAX 2SAT

Observation

- For any assignment of the variables x_1, x_2, x_3

$$C_j \text{ satisfied } \iff \exists y_j : \text{7 clauses of this group satisfied in } I'$$
$$\neg C_j \text{ satisfied } \iff \forall y_j, \text{ 5 clauses of this group satisfied in } I'$$

- Moreover, we know that $\texttt{OPT}(I) \geq 7/8 \cdot m$.

Lemma 7.7

If there is an α approximation algorithm for MAX 2SAT, then there is a $1 - \frac{27}{7}(1-\alpha)$-approximation algorithm for MAX E 3SAT

Reduction from MAX E 3SAT to MAX 2SAT

Observation

- For any assignment of the variables x_1, x_2, x_3

$$C_j \text{ satisfied} \iff \exists y_j: 7 \text{ clauses of this group satisfied in } I'$$
$$\neg C_j \text{ satisfied} \iff \forall y_j,\ 5 \text{ clauses of this group satisfied in } I'$$

- Moreover, we know that $\texttt{OPT}(I) \geq 7/8 \cdot m$.

Lemma 7.7

If there is an α approximation algorithm for MAX 2SAT, then there is a $1 - \frac{27}{7}(1-\alpha)$-approximation algorithm for MAX E 3SAT

We will see in the next section that there is no α-approximation algorithm for MAX E 3SAT with $\alpha > 7/8$, unless P=NP. Therefore, we get:

Theorem 7.8

There is no $\alpha-$approximation algorithm for MAX 2SAT for constant $\alpha > 209/216 \simeq 0.968$ unless P=NP.

L-Reductions

Consider two optimization problems $\mathcal{P}$ and $\mathcal{P}'$ with corresponding sets of instances $X_{\mathcal{P}}$ and $X_{\mathcal{P}'}$, respectively.

Definition 7.9 (L-Reduction)

An L-reduction from $\mathcal{P}$ to $\mathcal{P}'$ with parameters $a, b > 0$ is a map $f : X_{\mathcal{P}} \to X_{\mathcal{P}'}$ such that for all $I \in X_{\mathcal{P}}$:

- i $I' := f(I)$ can be computed in time polynomial in the size of I;
- ii $\mathtt{OPT}(I') \leq a \cdot \mathtt{OPT}(I)$;
- iii given a solution of value V' to I', one can compute in polynomial time a solution of value V to I such that

$$|\mathtt{OPT}(I) - V| \leq b \cdot |\mathtt{OPT}(I') - V'| \ .$$

L-Reductions

Consider two optimization problems $\mathcal{P}$ and $\mathcal{P}'$ with corresponding sets of instances $X_{\mathcal{P}}$ and $X_{\mathcal{P}'}$, respectively.

Definition 7.9 (L-Reduction)

An L-reduction from $\mathcal{P}$ to $\mathcal{P}'$ with parameters $a, b > 0$ is a map $f : X_{\mathcal{P}} \to X_{\mathcal{P}'}$ such that for all $I \in X_{\mathcal{P}}$:

i. $I' := f(I)$ can be computed in time polynomial in the size of I;
ii. $\mathtt{OPT}(I') \leq a \cdot \mathtt{OPT}(I)$;
iii. given a solution of value V' to I', one can compute in polynomial time a solution of value V to I such that
$$|\mathtt{OPT}(I) - V| \leq b \cdot |\mathtt{OPT}(I') - V'| \ .$$

Example: The reduction from MAX E 3SAT to MAX 2SAT in the previous proof is an L-reduction with parameters $a = \frac{54}{7}$ and $b = \frac{1}{2}$.

Approximation-Preserving Reductions

Theorem 7.10

For maximization problems $\mathcal{P}$ and $\mathcal{P}'$, if there is an L-reduction from $\mathcal{P}$ to $\mathcal{P}'$, and there is an α-approximation algorithm for $\mathcal{P}'$, then there is an $\big(1 - ab(1-\alpha)\big)$-approximation algorithm for $\mathcal{P}$.

Approximation-Preserving Reductions

Theorem 7.10

For maximization problems $\mathcal{P}$ and $\mathcal{P}'$, if there is an L-reduction from $\mathcal{P}$ to $\mathcal{P}'$, and there is an α-approximation algorithm for $\mathcal{P}'$, then there is an $\big(1 - ab(1-\alpha)\big)$-approximation algorithm for $\mathcal{P}$.

Proof:… □

Approximation-Preserving Reductions

Theorem 7.10

For maximization problems $\mathcal{P}$ and $\mathcal{P}'$, if there is an L-reduction from $\mathcal{P}$ to $\mathcal{P}'$, and there is an α-approximation algorithm for $\mathcal{P}'$, then there is an $\big(1 - ab(1-\alpha)\big)$-approximation algorithm for $\mathcal{P}$.

Proof:... □

Theorem 7.11

For minimization problems $\mathcal{P}$ and $\mathcal{P}'$, if there is an L-reduction from $\mathcal{P}$ to $\mathcal{P}'$, and there is an α-approximation algorithm for $\mathcal{P}'$, then there is an $\big(ab(\alpha-1)+1\big)$-approximation algorithm for $\mathcal{P}$. □

MaxClique Problem and Maximum Independent Set

MaxClique

Given: Undirected graph $G = (V, E)$.

Task: Find $V' \subseteq V$ maximizing $|V'|$ with all nodes in V' pairwise adjacent.

MaxClique Problem and Maximum Independent Set

MaxClique

Given: Undirected graph $G = (V, E)$.

Task: Find $V' \subseteq V$ maximizing $|V'|$ with all nodes in V' pairwise adjacent.

Notation: The size of a largest clique $V' \subseteq V$ in G is denoted by $\omega(G)$.

MaxClique Problem and Maximum Independent Set

MaxClique

Given: Undirected graph $G = (V, E)$.

Task: Find $V' \subseteq V$ maximizing $|V'|$ with all nodes in V' pairwise adjacent.

Notation: The size of a largest clique $V' \subseteq V$ in G is denoted by $\omega(G)$.

Maximum Independent Set

Given: Undirected graph $G = (V, E)$.

Task: Find $V' \subseteq V$ maximizing $|V'|$ with V' an independent set (or a stable), i.e., all nodes of V' are pairwise non-adjacent.
Notation: The size of a largest stable $V' \subseteq V$ in G is denoted by $\alpha(G)$.

MaxClique Problem and Maximum Independent Set

MaxClique

Given: Undirected graph $G = (V, E)$.

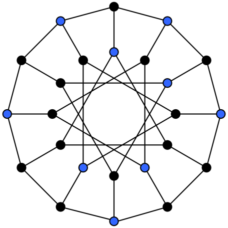
Task: Find $V' \subseteq V$ maximizing $|V'|$ with all nodes in V' pairwise adjacent.

Notation: The size of a largest clique $V' \subseteq V$ in G is denoted by $\omega(G)$.

Maximum Independent Set

Given: Undirected graph $G = (V, E)$.

Task: Find $V' \subseteq V$ maximizing $|V'|$ with V' an independent set (or a stable), i.e., all nodes of V' are pairwise non-adjacent.
Notation: The size of a largest stable $V' \subseteq V$ in G is denoted by $\alpha(G)$.



MaxClique Problem and Maximum Independent Set

MaxClique

Given: Undirected graph $G = (V, E)$.

Task: Find $V' \subseteq V$ maximizing $|V'|$ with all nodes in V' pairwise adjacent.

Notation: The size of a largest clique $V' \subseteq V$ in G is denoted by $\omega(G)$.

Maximum Independent Set

Given: Undirected graph $G = (V, E)$.

Task: Find $V' \subseteq V$ maximizing $|V'|$ with V' an independent set (or a stable), i.e., all nodes of V' are pairwise non-adjacent.
Notation: The size of a largest stable $V' \subseteq V$ in G is denoted by $\alpha(G)$.

Observation: MaxClique and Maximum Independent Set are fundamentally equivalent, as $\omega(G) = \alpha(\bar{G})$.

Examples of L-Reductions

Lemma 7.12

There is an L-reduction with parameters $a = 1$ and $b = 1$ from MAX E 3SAT to MaxClique.

Examples of L-Reductions

Lemma 7.12

There is an L-reduction with parameters $a = 1$ and $b = 1$ from MAX E 3SAT to MaxClique.

Proof: … □

Examples of L-Reductions

Lemma 7.12

There is an L-reduction with parameters $a = 1$ and $b = 1$ from MAX E 3SAT to MaxClique.

Proof: … □

Lemma 7.13

There is an L-reduction with parameters $a = 2\Delta$ and $b = 1$ from Vertex Cover in bounded degree graphs to the Steiner Tree Problem.

Examples of L-Reductions

Lemma 7.12

There is an L-reduction with parameters $a = 1$ and $b = 1$ from MAX E 3SAT to MaxClique.

Proof: … □

Lemma 7.13

There is an L-reduction with parameters $a = 2\Delta$ and $b = 1$ from Vertex Cover in bounded degree graphs to the Steiner Tree Problem.

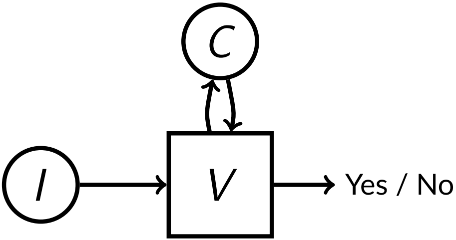
Moreover, it is known that for all Δ large enough, there exists $\epsilon > 0$ s.t. the existence of a $(1+\epsilon)$-approximation algorithm for vertex cover in bounded degree graphs ($\leq \Delta$) would imply P=NP.

Corollary 7.14

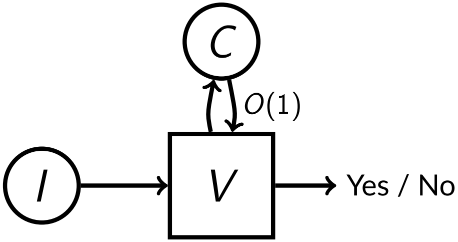
There is no PTAS for the Steiner tree problem, unless P=NP.

Outline

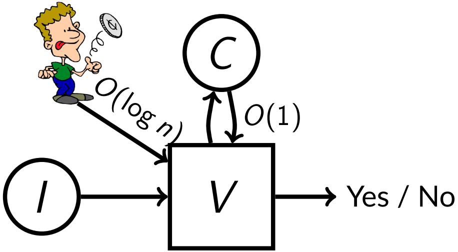
Another Characterization of NP



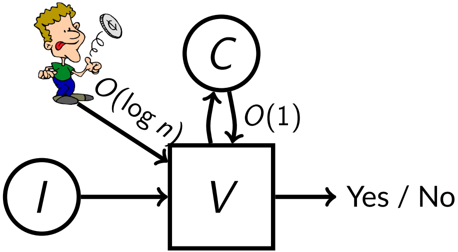
Another Characterization of NP



Another Characterization of NP

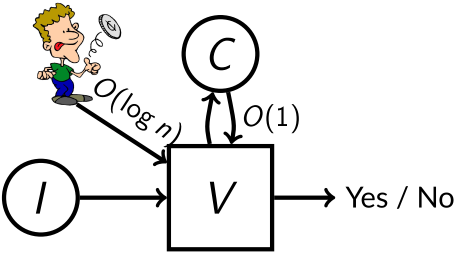


Another Characterization of NP



correct answer is “Yes” $\implies$ $\exists$ certificate C: Pr(V outputs “Yes”)$= 1$.

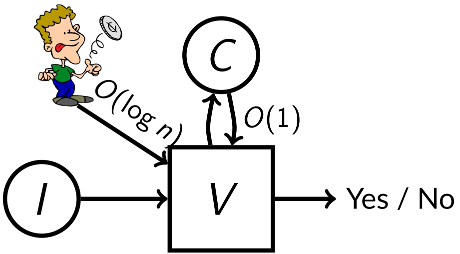
Another Characterization of NP



correct answer is "Yes" $\implies$ $\exists$ certificate C: Pr(V outputs "Yes")$= 1$.

correct answer is "No" $\implies$ $\forall$ certificates C: Pr(V outputs "Yes")$\leq \frac{1}{2}$.

Another Characterization of NP



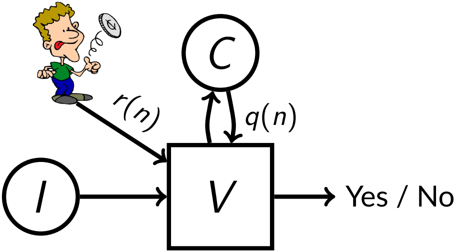
correct answer is "Yes" $\Longrightarrow$ $\exists$ certificate C: Pr(V outputs "Yes")$= 1$.

correct answer is "No" $\Longrightarrow$ $\forall$ certificates C: Pr(V outputs "Yes")$\leq \frac{1}{2}$.

Definition 7.15

The class of decision problems that admit such probabilitically checkable proofs is called PCP $\equiv \mathrm{PCP}_{1,\frac{1}{2}}[O(\log(n)), O(1)]$.

Another Characterization of NP



correct answer is "Yes" $\Longrightarrow$ $\exists$ certificate C: Pr(V outputs "Yes")$\geq c$.
correct answer is "No" $\Longrightarrow$ $\forall$ certificates C: Pr(V outputs "Yes")$\leq s$.

Definition 7.15

The class of decision problems that admit such probabilitically checkable proofs is called PCP $\equiv \mathrm{PCP}_{1,\frac{1}{2}}[O(\log(n)), O(1)]$.

More generally, we can define the class $\mathrm{PCP}_{c,s}[r(n), q(n)]$, so that the standard definition of NP reads: $\mathrm{NP}=\mathrm{PCP}_{1,0}[0, \mathrm{poly}(n)]$.

Hardness of Approximation

Theorem 7.16 (PCP Theorem) [Arora, Lund, Motwani, Sudan & Szegedy 92]

NP=PCP.

□

Hardness of Approximation

Theorem 7.16 (PCP Theorem) [Arora, Lund, Motwani, Sudan & Szegedy 92]

NP=PCP. In words, this means that every decision problem in NP has a probabilistically checkable proof of constant query complexity and logarithmic randomness complexity.

□

Hardness of Approximation

Theorem 7.16 (PCP Theorem) [Arora, Lund, Motwani, Sudan & Szegedy 92]

NP=PCP. In words, this means that every decision problem in NP has a probabilistically checkable proof of constant query complexity and logarithmic randomness complexity.

Proving PCP $\subseteq$ NP is easy. The converse inclusion is much more involved.

□

Hardness of Approximation

Theorem 7.16 (PCP Theorem) [Arora, Lund, Motwani, Sudan & Szegedy 92]

NP=PCP. In words, this means that every decision problem in NP has a probabilistically checkable proof of constant query complexity and logarithmic randomness complexity.

Proving PCP $\subseteq$ NP is easy. The converse inclusion is much more involved. The following theorem shows it can also be viewed as a result of hardness of approximation:

Theorem 7.17

(NP$\subseteq$ PCP) if and only if there exists $\epsilon > 0$ such that the problem of distinguishing between MAX E 3SAT instances for which there is a variable assignment satisfying all clauses, from instances in which at most a (1-ϵ) fraction of all clauses can be satisfied simultaneously, is NP-hard.

Proof: . . . $\square$

Product Graph

Definition 7.18 (product graph)

For an undirected graph $G = (V, E)$ let $G^k = (V^k, E_k)$ where $V^k = V \times V \times \cdots \times V$ is the set of all k-tuples of V and E_k is defined by

$$E_k := \{(u_1, \ldots, u_k)(v_1, \ldots, v_k) \mid u_i = v_i \text{ or } u_i v_i \in E \text{ for all } i\} \quad .$$

Product Graph

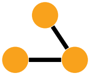
Definition 7.18 (product graph)

For an undirected graph $G = (V, E)$ let $G^k = (V^k, E_k)$ where $V^k = V \times V \times \cdots \times V$ is the set of all k-tuples of V and E_k is defined by

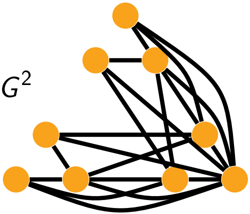
$$E_k := \{(u_1, \ldots, u_k)(v_1, \ldots, v_k) \mid u_i = v_i \text{ or } u_i v_i \in E \text{ for all } i\} \ .$$

Example:

G



G^2



Lemma 7.19

It holds $\omega(G^k) = \omega(G)^k$. Moreover, given a clique C of G^k, one can efficiently compute a clique C' of G, of size $|C'| \geq |C|^{1/k}$.

MaxClique: Self improvement of approx. ratio

Proposition 7.20

If there is an α-approximation algorithm for MaxClique for some fixed $\alpha < 1$, then there is a PTAS.

MaxClique: Self improvement of approx. ratio

Proposition 7.20

If there is an α-approximation algorithm for MaxClique for some fixed $\alpha < 1$, then there is a PTAS.

Proof:… □

MaxClique: Self improvement of approx. ratio

Proposition 7.20

If there is an α-approximation algorithm for MaxClique for some fixed $\alpha < 1$, then there is a PTAS.

Proof:... □

But by Lemma 7.12, if there is a PTAS for MaxClique, there is also a PTAS for MAX E 3SAT, and by the PCP theorem, this would imply P=NP !

MaxClique: Self improvement of approx. ratio

Proposition 7.20

If there is an α-approximation algorithm for MaxClique for some fixed $\alpha < 1$, then there is a PTAS.

Proof:... □

But by Lemma 7.12, if there is a PTAS for MaxClique, there is also a PTAS for MAX E 3SAT, and by the PCP theorem, this would imply P=NP !

Theorem 7.21

If there is a constant factor approximation algorithm for MaxClique, then P=NP.

MaxClique: Self improvement of approx. ratio

Proposition 7.20

If there is an α-approximation algorithm for MaxClique for some fixed $\alpha < 1$, then there is a PTAS.

Proof:… □

But by Lemma 7.12, if there is a PTAS for MaxClique, there is also a PTAS for MAX E 3SAT, and by the PCP theorem, this would imply P=NP !

Theorem 7.21

If there is a constant factor approximation algorithm for MaxClique, then P=NP.

Theorem 7.22 (Zuckerman 2007)

There is no $n^{-1+\varepsilon}$-approximation algorithm for MaxClique, for any $\varepsilon > 0$, unless $P = NP$.

MaxClique: Self improvement of approx. ratio

Proposition 7.20

If there is an α-approximation algorithm for MaxClique for some fixed $\alpha < 1$, then there is a PTAS.

Proof:… □

But by Lemma 7.12, if there is a PTAS for MaxClique, there is also a PTAS for MAX E 3SAT, and by the PCP theorem, this would imply P=NP !

Theorem 7.21

If there is a constant factor approximation algorithm for MaxClique, then P=NP.

Theorem 7.22 (Zuckerman 2007)

There is no $n^{-1+\varepsilon}$-approximation algorithm for MaxClique, for any $\varepsilon > 0$, unless $P = NP$.

Approximability of MAX E 3SAT

Theorem 7.23

For all $\epsilon, \delta > 0$, NP $\subseteq \mathrm{PCP}^{\text{parity}}_{1-\epsilon, \frac{1}{2}+\delta}[O(\log(n)), 3]$, where *parity* indicates that the verifyer can only evaluate the parity $(x_{i_1} + x_{i_2} + x_{i_3} \mod 2)$ of the three bits it checks.

Approximability of MAX E 3SAT

Theorem 7.23

For all $\epsilon, \delta > 0$, NP $\subseteq \mathrm{PCP}^{\text{parity}}_{1-\epsilon, \frac{1}{2}+\delta}[O(\log(n)), 3]$, where *parity* indicates that the verifyer can only evaluate the parity $(x_{i_1} + x_{i_2} + x_{i_3} \mod 2)$ of the three bits it checks.

Corollary 7.24

There is no $\alpha-$approximation algorithm for MAX E 3SAT for some fixed $\alpha > 7/8$, unless $P = NP$.